Codes with Availability and Multiple Local Erasure Correction for Distributed Data Storage

Ujwal Deep Kadiyam

Codes with Availability and Multiple Local Erasure Correction for Distributed Data Storage

A

Thesis Submitted

in Partial Fulfillment of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

By

UJWAL DEEP KADIYAM

DEPARTMENT OF ELECTRONICS AND ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

GUWAHATI - 781 039, ASSAM, INDIA

Declaration

This is to certify that the thesis entitled "**Codes with Availability and Multiple Local Erasure Correction for Distributed Data Storage**", submitted by me to the *Indian Institute of Technology Guwahati*, for the award of the degree of Doctor of Philosophy, is a bonafide work carried out by me under the supervision of Dr. Smarajit Das. The content of this thesis, in full or in parts, have not been submitted to any other University or Institute for the award of any degree or diploma. I also wish to state that to the best of my knowledge and understanding nothing in this report amounts to plagiarism.

Signed:

Ujwal Deep Kadiyam
Department of Electronics and Electrical Engineering,
Indian Institute of Technology Guwahati,
Guwahati-781039, Assam, India.

Date: 05-05-2021.

Certificate

This is to certify that the thesis entitled "**Codes with Availability and Multiple Local Erasure Correction for Distributed Data Storage**", submitted by **Ujwal Deep Kadiyam** (136102027), a *research scholar* in the *Department of Electronics and Electrical Engineering*, *Indian Institute of Technology Guwahati*, for the award of the degree of **Doctor of Philosophy**, is a record of an original research work carried out by him under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the institute and in my opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Signed: Smarajit

Supervisor: Dr. Smarajit Das
Department of Electronics and Electrical Engineering,
Indian Institute of Technology Guwahati,
Guwahati-781039, Assam, India.

Date: 05-05-2021.

I would like to dedicate this thesis to my children,

Ryan and Menora

Acknowledgements

This has been an incredible journey for me. I would like to express my gratitude to my supervisor Dr. Smarajit Das. It has been a privilege to work under his supervision. His clarity in analyzing a topic and his ability to see beyond the horizon is impeccable. He has been an inspiration to me and I will try to inculcate these skills in myself. I sincerely thank him for his kind and gracious support to me, in all of my ups and downs in this journey. He is very patient and understanding, and it is evident to anyone who get to know him. I had many health issues during my PhD and he was very supportive in all of those times. He has been a very good teacher. He has clarified all my doubts with patience, multiple times. He has always been respectful to me, even to the rest of his students as well. I cannot explain how good an impact his behavior has made on me when treating others. I would like to extend my gratitude to his family, who were nothing but warm and loving towards me and my family.

I would like to thank Dr. A. Rajesh, chairman of my doctoral committee for his support and guidance throughout my time in IIT Guwahati. He was always concerned about my health and it always showed when he interacted with me. I also thank him for his suggestions and feedback.

I would like to thank Dr. Tony Jacob, a member in my doctoral committee for his guidance and support in my thesis work. He always delves deep into my topic and his valuable suggestions helped me improve my work.

I would like to thank Dr. Benny George, another member in my doctoral committee for his guidance and support in my thesis work. He was very supportive with his suggestions and feedback, which helped me improve my work.

All the members of my doctoral committee took out time in helping me to improve myself and my work. I am extremely grateful to them.

I would like to specially thank Dr. Vidyacharan Bhaskar, my mentor during my M.Tech, as he was my motivation to pursue my PhD.

This journey would not have been possible with the support of my family members. I would like to thank my wonderful and lovely wife, Mary Silvya, for her unconditional support and her innumerable sacrifices throughout these years. She even quit her job to support me here. My love is with my two children, Ryan and Menora. I am deeply indebted to my parents, Babu Rao and Raja Mallika. My

mother has made many sacrifices, so that i would finish my PhD. They have supported me mentally and financially when needed. Without their help it is impossible for me to complete my PhD. I would thank my brother, Vishal Roshan, who has helped me in every way possible during this PhD. I would also like to thank my uncle Bharath kumar, for his unconditional support during my PhD.

Many friends in IIT Guwahati helped me through all these years. I would like to the thank k. Dharmalingam, Ravi.B, Rajasekhar Ravula, Hari krishna Veldandi, Sivarama prasad Tera, Bhagath.P and their families for their support to me and my family through out our stay in this campus. I have received a lot of encouragement and valuable suggestions from these people. I had a wonderful time interacting with these people sharing not just PhD related matters, but also personal and social issues. I would like to thank Kranti, Vivian and Teja for their help and affection.

I would like to thank Mr. Sam Israel, Mr. Josiah, Mr. Joseph, Dr. Atul kumar, Dr. Bithya grace, Dr. Prakash, Mr. Haokip, Dr. Roy paily and Josephine madam for their love and support through out my stay in IIT Guwahati. I will always remember their hospitality.

I would like to thank my lab mates deepak agrawal, sivarama prasad, prateek, ramanand, kaushik, vijith, rahul, pawan, vikas, vikranth, himanchal for their friendship and guidance throughout this journey.

I would like to thank the administrative staff in EEE office and in the institute administration for helping me when needed. I would like to thank technical officers in the department for addressing my concerns promptly.

Lastly, I would like to thank *our creator* for giving me an opportunity to study in this prestigious and wonderful institute. Without this opportunity, I would not have met all these wonderful people and would not have grown as a person. I have learnt to be humble, dedicated, caring and loving towards others and help others to the best of my ability. This opportunity has made me a better person than I was, when I joined this institute. Since *our creator* loves us, I want all of us to extend the same love and courtesy to the people around us and make their lives better.

Ujwal Deep Kadiyam

Abstract

In any data storage system, the system should provide the data even in the presence of disk failures. Coding theory is used to provide these features in Distributed Data storage systems. Locally Recoverable codes (LRCs) are used to recover data from a failed node in distributed data storage systems. LRCs with availability are proposed to provide accessibility to a disk by multiple users at the same time.

In a locally recoverable code (LRC), any code symbol can be recovered by accessing at most r other symbols (called a recovery set). In an LRC with *availability*, any information symbol has t disjoint recovery sets. As a first result in this thesis, we consider a new class of codes with availability, where the $l^{th}, 1 \leq l \leq t$ disjoint recovery set for any information symbol has locality r_l and it is protected by a local code of minimum Hamming distance at least δ_l. These codes are referred to as codes with $\{(r_1, \delta_1), \ldots, (r_t, \delta_t)\}$ availability. We derive an upper-bound on the minimum Hamming distance of these codes. A family of systematic codes with information availability are constructed achieving the bound with equality. We show that these optimal codes provide availability for any information symbol even when some symbols are erased from its disjoint recovery sets.

Codes with availability t allow us to access any code symbol simultaneously t times using t disjoint recovery sets. Codes with availability that are studied in the literature provide availability only for individual code symbols in the code. As a second result in this thesis, we consider a new class of codes with availability. These codes provide availability for a subset of code symbols. These are referred to as codes with multi-symbol availability. For this code, an upper-bound on minimum Hamming distance is derived and optimal codes with information multi-symbol availability have been constructed attaining the bounds with equality.

For codes with availability t that are studied in literature, the disjoint recovery sets are fixed. This restriction imposes limitations on the construction of optimal codes. Hence optimal codes with

availability do not exist for many parameters. As a third result in this thesis, we define codes with availability, where the disjoint recovery sets for a code symbol are not fixed. One can choose t subsets, each consisting of any r code symbols to provide availability for a code symbol. We derive an upper bound on the minimum Hamming distance of this code and provide optimal code constructions.

We show that usage of these codes in distributed storage blockchain with Local secret sharing scheme can increase the ability of the system to recover the transaction even when a large number of nodes fail in the system. It is shown that this node failures is almost equal to the case in traditional blockchain.

Keywords: locality, availability, multiple local recovery, minimum Hamming distance, Distributed storage systems, locally recoverable codes, multiple local erasure correction, blockchain.

Contents

List of Figures

List of Tables

1

Introduction

Contents

1.1 Introduction

Distributed Storage systems of large scale are used to store the data on the cloud. They are found in practice [1, 2]. Recovery from several failures is now common in these storage systems rather than an exception. Along with the recovery, the storage system should provide high data availability to ensure that many users can access the same data simultaneously. These features are desirable for users who want to store and retrieve their data in a distributed data storage system.

To design a system providing these features, redundant information must be introduced into the system. This is achieved by encoding the data using codes. These codes are designed and studied in communication theory for recovery of errors and erasures that arise during transmission and storage of data. Since a node failure in a distributed data storage system can be viewed as an erasure, coding theory has interesting applications in this area.

Data replication i.e., where a node is replicated is a popular technique for data recovery. For each node, there will be $n-1$ other copies of it. This is called as nX replication. It is obvious that a large amount of storage is needed to store the data using replication. For example, if there are 10 nodes and if 3X replication is used, then each node has two other distinct copies of it in the system. The total number of nodes is 30 and the storage overhead is 200%.

It is desirable to use a scheme with smaller storage overhead and at the same time providing similar resiliency against failures as a replication based system. This can be achieved using coding techniques developed in coding theory. Facebook uses a Reed-Solomon code [3] to achieve this. A Reed-Solomon code is an erasure code, where it recovers the data stored in the presence of erasures. Reed-Solomon code is a class of MDS code, where any k out of the n code symbols can be used to recover the k message symbols. A $(14, 10)$ Reed-Solomon code is used in Facebook, where it has 10 data nodes and they are encoded into 14 nodes. The storage over head here is 40%. Redundancy and reliability can be improved with erasure codes in distributed data storage.

When a failed node occurs in a distribute storage system and it needs to recovered, during the recovery process, the following questions need to be answered:

(1) How many nodes should be contacted in the system to recover the failed node ?

(2) How much data should to be downloaded from these nodes ?

Using a $[n, k, d]$ MDS code, to recover a failed node, the system needs to access k other nodes and recover all the k information nodes and choose the data that is stored in the failed node. This is a wastage of the system resources, as the system needs to access more nodes and the system needs to read and write more data to recover one failed node.

There are two classes of erasure codes providing recovery techniques for a node failure in the literature.

Regenerating Codes: These codes are introduced in [4]. We request the reader to refer [5] for a detailed survey. Regenerating codes use the concept of sub-packetization. Each packet is composed of few sub-packets. When a node storing a packet fails some or all of the other nodes send in some of their sub-packets for recovery. This system is evaluated by the total amount of data that is downloaded to repair a failed node. This is called as repair bandwidth. Regenerating codes can decrease the repair bandwidth significantly. However, based on the experiences of data centers, accessing less number of nodes is more desirable in case of repairing a failed node.

Locally Recoverable Codes (LRCs):These codes allow the system to recover the failed node, by accessing fewer other nodes (at most r) [6]. Since data is read from fewer disks, these reduce the disk I/O. It is extremely useful for the information symbols to have small locality. Windows Azure, the cloud storage service provided by Microsoft uses a locally recoverable code [7]. Let's say a code symbol has locality r in an $[n, k, d]$ LRC denoted by $\mathcal{C}$. This code symbol along with its recovery set will be in a sub code of $\mathcal{C}$. This sub code is called as local code. The LRCs where each local code has minimum Hamming distance at least δ and locality r are studied in [8]. These codes are called codes with (r, δ) locality.

Locally Recoverable codes with availability: Codes with availability are defined in [9]. In these codes, any code symbol has t disjoint recovery sets. This allows us to recover any code symbol simultaneously t times. This is needed because, there will be some data which is very important and has to be accessed very frequently (called as *hot data*).

1.2 Problem statements

We address three problems in this thesis, which are described below. We specifically consider codes with locality and availability.

(1) Codes with availability allow us to recover more erasures locally [9]. However, it is shown that

the number of disk reads is high. So an interesting problem is to construct codes with availability, that can recover multiple erasures locally, with fewer disk reads.

The construction of optimal codes given in [9] are constructed with exactly one parity symbol in each local code. These codes are optimal with respect to an upper bound on the minimum Hamming distance ($d \leq n - k + 1 - \lceil \frac{kt}{r} \rceil + t$) for codes with availability which is derived in [9]. In these codes, for any information symbol, if there is an erasure in one of its recovery sets, then availability property for that information symbol is reduced as this recovery set cannot be used for accessing the information symbol. If there is an erasure in each of its recovery sets, then the availability property for that symbol is lost. There will not be availability for this information symbol. We investigate if availability property can be provided even when some symbols are erased in a local code.

(2) Codes with availability are studied in literature, where availability is provided for individual code symbols of the codeword. If availability is provided for a subset of code symbols, then one can access the subset of code symbols simultaneously t times, with the same locality. We investigate if availability can be provided to a subset of code symbols in the local code.

(3) Codes with availability are studied in literature, where availability is provided for a code symbol using a fixed set of symbols i.e., it has a fixed recovery set. A code symbol can be recovered by only that set. These codes need *membership matrices*, which are special matrices that describe the disjoint recovery sets for the set of information symbols. Membership matrices are difficult to construct. Due to this, optimal codes do not exist for a lot of parameters. We investigate if a code symbol has dynamic recovery sets i.e., it can be recovered from any r other symbols in a local code.

1.3 Results presented in this thesis

The solutions for the problems described in section 1.2 are proposed in this thesis. A brief description of the results are given here.

1.3.1 Availability with different localities and multiple local erasure correction.

We consider a new class of codes with availability. These codes are referred to as codes with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ availability. These codes provide availability t for the information symbols. The

$l^{th}, 1 \leq l \leq t$ disjoint recovery set for any information symbol is protected by a local code with locality r_l and minimum Hamming distance at least δ_l. We show that using these codes, the disk reads can be reduced for recovering more erasures locally. An upper bound on its minimum Hamming distance is derived. Optimal codes are constructed achieving this bound with equality. This construction is based on pyramid code construction.

The optimal codes have an interesting property. We show that these optimal codes provide availability t for any information symbol even in the presence of any $\delta_l - 2$ code symbol erasures in each of the disjoint recovery sets of that information symbol.

These results are published in IEEE Communication letters and in conference proceedings SPCOM, 2018 and 2020.

1.3.2 Availability for multiple code symbols.

In this class of codes, availability is provided for a subset of information symbols called as *multi symbol sets*. The size of each multi symbol set is $\rho, \rho \leq r$. Each multi symbol set can be recovered by t disjoint recovery sets. Each local code in this code is protected by a local code of minimum Hamming distance at least δ, $\delta \geq \rho + 1$. An upper bound on its minimum Hamming distance is derived. Setting $\rho = 1$, we get the bound for the case when any information symbol has availability t and any local code is protected by a local code of minimum Hamming distance at least δ. Setting $\rho = 1$ and $\delta = 2$ will give us a bound derived in [9]. This result is accepted for publication in IEEE Communication Letters.

1.3.3 Availability with dynamic recovery sets.

We study codes where the disjoint recovery sets are dynamic in nature for any code symbol i.e., any subset of r code symbols in the local code will be able to recover the code symbol. We derive an upper bound on the minimum Hamming distance of these codes and show that (r, δ) codes can be used to construct optimal codes.

Distributed storage blockchain systems are studied in [10–12]. We show that these codes (availability with dynamic recovery sets) increase the ability of the system to recover the transaction even when a large number of nodes fail in the system. The traditional blockchain distributed system also has this property. The traditional blockchain distributed system can recover the transaction, when a certain number of nodes fail (Say X) [12]. We show that when the parameters are chosen appropriately, the

distributed storage blockchain system can also recover the transaction, when the number of nodes failed are almost equal to the traditional blockchain system (i.e. X number of node failures).

1.4 Thesis Organization

The thesis is organized into six chapters. The summary of each chapter is briefly outlined as follows:

Chapter 1: In this chapter, an introduction of distributed data storage system is provided. The problems that are studied in this thesis are introduced and a summary of the solutions is provided.

Chapter 2: In this chapter, a detailed review of literature is provided. Coding techniques for distributed data storage system are introduced in detail.

Chapter 3: In this chapter, we have introduced a class of codes with locality and multiple local erasure correction property. We have derived an upper bound on the minimum Hamming distance for these codes. A construction for optimal codes satisfying the bound with equality is given. We explain the properties of these codes using examples.

Chapter 4: In this chapter, we define a new class of codes which provide availability to subset of information symbols rather than a single code symbol. We derive an upper bound on their minimum Hamming distance. A construction for optimal codes is given.

Chapter 5: In this chapter, we study codes with availability where the t disjoint recovery sets for any code symbol are dynamic in nature. i.e., the code symbol can be recovered by any set of at most r other code symbols in the local code. We derive an upper bound on the minimum Hamming distance of these codes. We provide the applications of these codes to distributed storage blockchain system.

Chapter 6: In this chapter, a summary of the results presented in this thesis is provided and future work is discussed.

2

Literature Review

Contents

2.1 Introduction

A large number of hard drives are used to store data in a distributed data storage system. The components in the storage system are always not reliable. It is shown in [13] that the reliability is very low. There are two common failure scenarios [7]:

(1) Degraded Read : A machine is down or slow or is in an upgrade. It is temporary unavailability of data.

(2) Disk failure : The hard drive failed. It needs to be replaced with a new hard drive and the data (on the failed drive) should be recreated on a new drive.

A single node failure is very common in a distributed storage system. One of the most used technique for data reliability in storage systems is replication. In replication, the system stores many copies of the data. For each data node, there will be $n-1$ other copies of it. This is called as nX replication. Replication schemes require more storage space than the original data (the data that needs to be stored) itself, thus increasing the storage overhead. This increases the cost of hardware and also the cost of maintenance of the data center.

To reduce the storage overhead, erasure codes are used in data centers. In these codes, the data is split into k fragments. These k fragments are encoded into n fragments and are stored on n disks (or nodes). Generally, maximum distance separable (MDS) codes are used. With MDS codes, all the k data fragments can be recovered, even when any $n-k$ nodes (fragments) fail. A comparison of these two schemes are explained using an example, which is given below.

Example: Consider a data file that needs to be stored on the cloud. The data file is split into two fragments that are stored in two data nodes **X** and **Y**. These two data nodes are stored using a 2X replication scheme and a $(4, 2)$ MDS erasure code. So, the storage overhead is same. The schemes are illustrated in the Fig. 2.1.

Replication: In this scheme, two copies of each of the two data nodes are stored. So, the total number of nodes is four. The four nodes are ordered from top to bottom. In this system, if node 1 and node 3 or node 2 and node 4 are lost, the data can still be recoverable as one can still access **X** as well as **Y**.

But if node 1 and node 2 are lost, then the data node **X** is lost completely in the system and it cannot be recovered. Similarly if node 3 and node 4 are lost, then the data node **Y** is lost completely

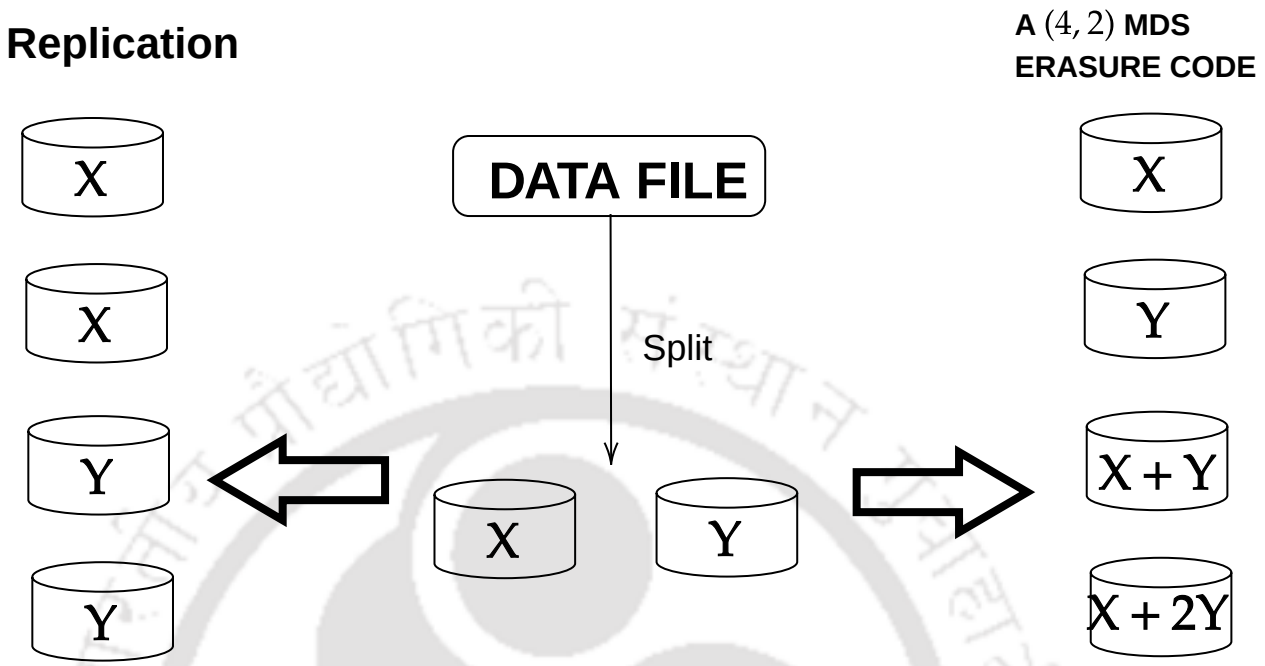

Figure 2.1: Replication and Erasure Coding

in the system and it cannot be recovered.

A $(4, 2)$ *MDS Erasure code*: Using the two data nodes, two more parity nodes which are dependent on both of these nodes are created and stored in the system. The total number of nodes is four. Due to the MDS property, this system can withstand any two erasures i.e., both the data nodes can be accessed when any two nodes fail.

It is obvious that, with the same number of nodes, erasure codes already present an advantage regarding data recovery in the presence of arbitrary node failures. In general, to achieve the same fault tolerance, erasure codes require less number of nodes than replication. Also note that MDS codes are too costly for maintenance.

This motivated to design storage systems based on erasure coding schemes (e.g. Ocean store [14]), to reduce the storage overhead. The most predominantly used erasure coding scheme is RAID [15]. The disk I/O is very high in erasure coding schemes. This is because, to recover the data in a failed node, more disks need to be accessed and the system needs to read data on all of these disks. This is called as a repair problem.

The Repair problem: To repair using an $[n, k, d]$ MDS code, the machine reads data from k other nodes. From these k nodes, the system reconstructs the entire file. This means the system needs

to access a number of nodes and read the data from all of these nodes, to recover data at a single failed node. To recover data from a failed node, is it needed to reconstruct the entire file ?

Three metrics are used to quantify the repair efficiency of a system in case of failure of a single node [16]:

(1) **Repair Bandwidth**: The amount of data communicated in the network to complete the repair of the failed node.

(2) **Disk I/O**: The amount of data read from/written on physical hard drives.

(3) **Locality**: The number of nodes needed to participate in the repair process.

In Erasure codes, the repair bandwidth is extremely high as we need to download all the data from the k nodes. This naturally increases the disk I/O of the system.

There has been considerable interest in designing codes satisfying each of these metrics. Regenerating codes [4] deal with reducing repair bandwidth. Codes that deal with disk I/O are studied in [17, 18]. Codes that deal with locality are studied in [6, 19–22].

2.2 Notations used in this thesis

Let $\mathcal{C}$ be a linear $[n, k, d]$ code and G be a generator matrix of $\mathcal{C}$. A set of k columns of G that are linearly independent is indexed by the set $\mathcal{I} \subset [n]$. Note that the set of columns of G indexed by $\mathcal{I}$ is a basis for the column space of G. The n symbols of a codeword c in the code $\mathcal{C}$ are indexed by the set $[n] \coloneqq \{1, 2, \ldots, n\}$ and we write $c = (c_1, c_2, \ldots, c_n)$. The k information symbols are indexed by i and $i \in [k]$. A punctured sub-code of $\mathcal{C}$ denoted by $\mathcal{C}|_S$ is obtained by restricting the codeword to the code symbols indexed by the set $S \subset [n]$ and the generator matrix of $\mathcal{C}|_S$ is given by $\mathrm{G}|_S$. The support of a punctured code is the subset of co-ordinates at which the punctured code has a non-zero entry. The dimension of $\mathcal{C}|_S$ is denoted as $\dim(\mathcal{C}|_S)$. The column space of $\mathrm{G}|_S$ is denoted as $\mathrm{col}(\mathrm{G}|_S)$. The cardinality of a set X is denoted by $|X|$. The union of disjoint sets is denoted as $\biguplus$.

2.3 Regenerating codes

As mentioned earlier, the efficiency of the repair process is studied by three metrics. Regenerating codes aim to minimize the repair bandwidth. Before discussing regenerating codes in detail, we introduce the types of repair. There are three types of repair process of a failed node [5]:

(1) Exact Repair : The data in the failed node is regenerated exactly in the new node.

(2) Functional Repair: The data in the new node can be different to the data in the failed node as long as MDS property in the system is maintained.

(3) Hybrid Repair: The systematic nodes are exactly repaired and the parity nodes are functionally repaired.

(A) Information Flow graph model for Functional Repair:

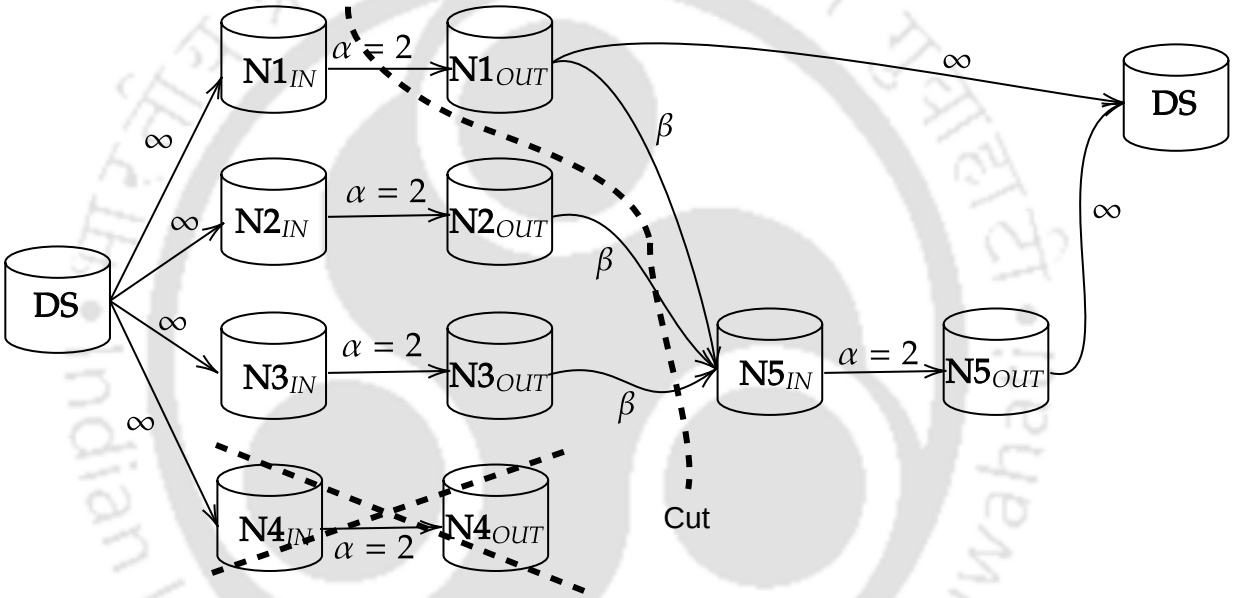

Figure 2.2: An example of Information flow graph with the cut.

The functional repair problem can be represented as multicasting over information flow graph, as shown in [4]. The information flow graph describes how information flows through a distributed storage network as failed nodes are discarded and new nodes are added into the network. An example of the information flow graph in given in Fig. 2.2.

This information flow graph is for a $(4, 2)$ MDS code. The node, Data Source (DS) corresponds to the source of the data file. This is a virtual node (a node that does not exist in real, but is assumed to exist for analysis). Let the data file be of size $\mathcal{B} = 4$ blocks. Each node stores $\alpha = 2$ blocks. Hence this code can be seen as a vector code (a code where each code symbol is a vector). The four storage nodes are $\mathbf{N1}, \mathbf{N2}, \mathbf{N3}, \mathbf{N4}$. Each storage node is denoted by two nodes in the graph. For example, the storage node $\mathbf{N1}$ is denoted by two nodes $\mathbf{N1}_{IN}$ and $\mathbf{N1}_{OUT}$ with an edge of capacity α in between

them . Since the code is MDS, the file can be reconstructed by a data collector (DC) by accessing any 2 nodes. DC is also a virtual node.

When a node fails, the objective is to reconstruct the data that is in the failed node on another disk. For example, let **N4** be the failed node in the system. The system needs to create a new node **N5** and recreate the data on it such that, the MDS property of the code is still satisfied. **N5** should download the minimum amount of data β from each of the other nodes and it should store $\alpha = 2$ two symbols.

As mentioned earlier, the functional repair problem can be viewed as multicasting problem, where the source DS is multicasting the file to all possible data collectors.

It is shown in [5] that the maximum multicast rate is equal to minimum-cut capacity of the network connecting the data source and a data sink. It is also shown that this capacity can be achieved using linear network coding. So, to obtain this rate, a cut analysis of the information flow graph is done.

(B) Cut Analysis of the Information flow graph: For the information flow graph in Fig. 2.2, $\mathbf{N5}_{IN}$ connects to $d = 3$ active nodes to reconstruct the data. It downloads β blocks from each of the active nodes. The minimum value of β is to be determined. Since, the data collector needs to access the entire data, the minimum cut of the graph should be at least the file size $\mathcal{B} = 4$. The min-cut of this graph is $\alpha + 2\beta$. So,

$$\alpha + 2\beta \geq 4$$

$$\beta \geq 1.$$

Since there are $d = 3$ nodes from which the data is downloaded, the repair bandwidth is

$$\gamma = d\beta = 3.$$

The terminology is introduced below.

(1) $\mathcal{B}$: size of the file to be stored in the system.

(2) n: total number of nodes in the system.

(3) k: total number of information (message) nodes in the system.

(4) d: number nodes that participate in the repair process.

(5) α: amount of data stored in each node.

(6) β: amount of data downloaded by the new node from each of the d other nodes for repair. And $\beta \leq \alpha$.

(7) γ: repair bandwidth, where $\gamma = d\beta$.

It was shown in [4] that in order to repair the failed node, the parameters of the code to be designed must necessarily satisfy,

$$\mathcal{B} \leq \sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\}.$$

While it is desirable to minimize both α and β, it is evident that both cannot be minimized simultaneously. There exists a trade-off. Minimizing α first and then minimizing β, will give one extreme point of the trade-off, called as Minimum Storage Regeneration (MSR) point. At MSR point we have

$$\alpha = \frac{\mathcal{B}}{k}$$
$$\beta = \frac{\mathcal{B}}{k(d-k+1)}.$$

Minimizing β first and then minimizing α, we have the other extreme point in the trade-off, called as Minimum Bandwidth regeneration (MBR) point. At MBR point, we have

$$\beta = \frac{2\mathcal{B}}{k(2d-k+1)}$$
$$\alpha = \frac{2\mathcal{B}}{k(2d-k+1)} = d\beta.$$

It is shown [4] that γ is minimized when d is maximum i.e.,

$$d = n - 1.$$

The minimum value of γ at MSR point is given by

$$\frac{\mathcal{B}(n-1)}{k(n-k)}.$$

Similarly, the minimum value of γ at MBR point is given by

$$\frac{2\mathcal{B}}{2kn - k^2 - k}.$$

In practice, functional repair is not preferred in storage systems. This is because, the code changes every time a node is repaired and this new information must be shared with all the other nodes in

the system. Since failures are frequent, this has to be done frequently. A systematic code with exact repair is extremely useful in practical applications. Hence, exact repair is of extreme interest.

Codes with exact repair at MSR and MBR points are constructed in [23]. The non-existence of codes with exact repair at all points except at the MSR point, MBR point and a point next to MBR point is shown in [24]. An outer bound for the trade-off for exact repair codes is derived in [25]. For the regenerating code with parameters $(n = 4, k = 3, d = 4)$, the exact repair trade-off region is characterized in [26]. It is is established in [25, 26], that the trade-offs for functional repair and exact repair are separate, except for the MSR point, MBR point and a small region, immediately next to MSR point.

Explicit constructions for codes with exact repair are given in [17, 23, 27–29]. For all (n, k, d), the existence of MSR codes for $n > d \geq k$ is given in [30]. Explicit constructions for MBR codes for all (n, k, d) is given in [23]. A class of MBR codes called as repair-by-transfer MBR codes are constructed in [29]. They are referred to as repair-by-transfer MBR codes because, repair is done only by data transfer without any computations. The implementation of regenerating codes in distributed storage settings is studied in [31–33]. The performance evaluation of regenerating codes for distributed storage systems is studied in [31–33]. Codes with piggybacking framework to reduce repair bandwidth are studied in [34].

Regenerating codes are a class of vector codes over a vector alphabet $\mathbb{F}_q^m$, for some m. If $m = \alpha$, then regenerating codes may be considered as array codes in which a codeword is an array of size $(m \times n)$. A survey on array codes is found in [35]. The applications of array codes in storage systems are discussed in [36–38].

2.4 Locally Recoverable codes

In regenerating codes, to achieve the minimum repair bandwidth γ to repair a failed node, the system should read all the data from all the remaining $d = n - 1$ nodes. So, the disk I/O is extremely high.

This problem was addressed in [39] by Huang et.al. Without formally coining the term "locality", a code with locality is constructed in an attempt to reduce the number of nodes accessed in repairing a failed node. Two classes of codes: pyramid codes and generalized pyramid codes [39] are constructed.

Codes with locality allow us to recover a failed node in distributed systems using a few other nodes.

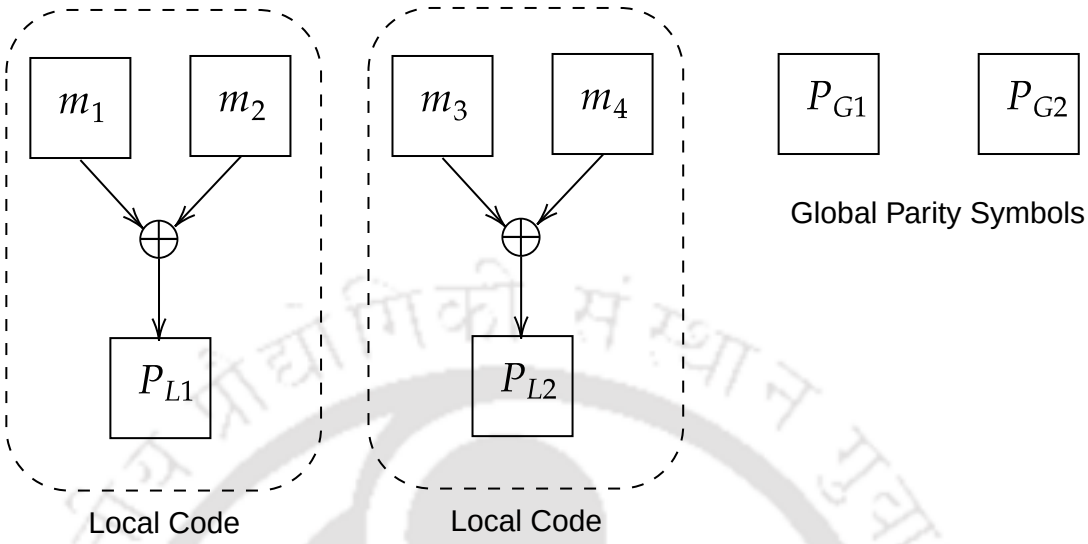

Figure 2.3: An example of a Locally Recoverable code with information locality.

These codes are referred to as *Locally Recoverable codes* (LRCs). This reduces the disk reads needed for the repair of a failed node. In an LRC any code symbol can be recovered by at most r other code symbols (called as a recovery set).

These codes are a sub-class of much general class of codes called *locally decodable codes*. Locally decodable codes are introduced in [40]. These are studied further in [41–44]. In a $r-$query locally decodable code, any message symbol can be recovered by r other symbols, even when 10% of code symbols are lost in the codeword. In other words, locality is provided even when significant number of code symbols are lost. For LRCs, locality is provided for a single failure.

The codes used in distributed data storage systems should have high rate, low locality and a large minimum Hamming distance. Gopalan et.al. carried out an in-depth study of the relations between these parameters in [6].

Definition 2.1 (Locality [6])**:** The i^{th} code symbol c_i, $1 \leq i \leq n$, of an $[n, k, d]$ linear code $\mathcal{C}$ over the field $\mathbb{F}_q$ is said to have locality r, if this symbol can be recovered by accessing at most r other code symbols of the code $\mathcal{C}$.

Definition 2.2 (Local code)**:** Consider a code symbol that has locality r in an $[n, k, d]$ code $\mathcal{C}$. This code symbol along with its recovery set will be in a sub code of $\mathcal{C}$. This sub code is called as local code.

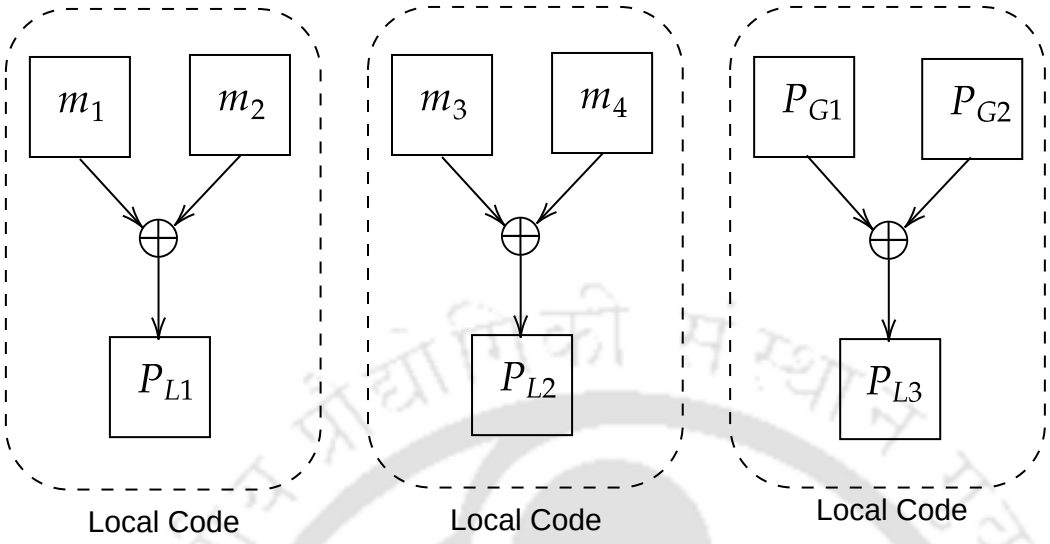

Figure 2.4: An example of a Locally Recoverable code with all-symbol locality.

Definition 2.3 (Information locality [6])**:** If the information symbols have a locality r then such codes are referred to as LRCs with *information locality*.

Codes with information locality are referred to as (r, d) codes in [6], where d is the minimum Hamming distance of the code.

Definition 2.4 (All-symbol locality [6])**:** If all the code symbols have locality r then the such codes are referred to as LRCs with *all-symbol locality*.

An example of an LRC is described in Fig. 2.3. In this code, $m_1, \cdots, m_4$ are the information symbols and P_{G1}, P_{G2} are the global parity symbols. The symbols m_1, m_2, P_{L1} form a local code as any symbol can be obtained as a linear combination of the other two. Similarly m_3, m_4, P_{L2} also form a local code. As only the information symbols has this locality property, this is a code with *information locality*.

Similarly, the code described in Fig. 2.4 is a code with all-symbol locality as it can be easily observed that all the code symbols have this locality property with locality $r = 2$. An upper-bound on the minimum Hamming distance of an $[n, k, d]$ code with (r, d) locality is derived in [6].

In the derivation of an upper-bound on the minimum Hamming distance of these codes, the following lemma is used.

Lemma 2.5 ([6, 8])**:** The minimum Hamming distance of an $[n, k, d]$ code with generator matrix G is given by

$$d \leq n - |T|$$

where $T \subseteq [n]$ is a set of largest cardinality such that, $\text{rank}(\text{G}|_T) \leq k - 1$.

Theorem 2.6 ([6])**:** For any $[n, k, d]$ linear code over $\mathbb{F}_q$ with information locality r, the minimum Hamming distance is upper-bounded by

$$d \leq n - k - \left\lceil \frac{k}{r} \right\rceil + 2. \tag{2.1}$$

An (r, d)-code $\mathcal{C}$ is optimal, if its parameters satisfy equation (2.1) with equality.

Code constructions that meet the bound with restrictions for information locality are provided in [6]. They also show the existence of codes with all-symbol locality. LRCs are used in "Windows Azure Storage (WAS)", a cloud service provided by Microsoft [7].

Advantages of LRCs: Since the locality r is small $1 \leq r \leq k$, the system needs to read the data from r other nodes. For MDS codes, the data has to be read from k nodes. For regenerating codes, the data has to be read from $n - 1$ nodes. Since $r \leq k < n - 1$, the disk I/O can be reduced using LRCs, as fewer disks need to be accessed.

The codes studied in [6] are single parity check codes. The idea of locality is extended to more general codes in [8]. These codes can recover more erasures locally. These codes are referred to as codes with (r, δ) locality. LRCs with (r, δ) locality can recover a failed node from r other nodes even in the presence of $\delta - 1$ erasures. The codes are extremely useful for recovering multiple failures in a distributed storage system.

Definition 2.7 ((r, δ) code symbol locality [8])**:** The i^{th} code symbol $c_i, i \in [n]$ of an $[n, k, d]$ code with the generator matrix G has (r, δ) locality, if there exists a subset $Q(i) \subset [n]$, such that,

(1) $i \in Q(i)$ and $|Q(i)| \leq r + \delta - 1$.

(2) $\mathcal{C}|_{Q(i)}$ is a code with minimum Hamming distance at least δ.

Note that $\dim(\mathcal{C}|_{Q(i)}) \leq r$. The following definition is for information locality.

Definition 2.8 ((r,δ) information locality [8])**:** An $[n,k,d]$ code $\mathcal{C}$ is an (r,δ) code with information locality if it has punctured sub codes $\mathcal{C}(j), j \in \mathcal{M}$ with supports $Q(j), j \in \mathcal{M}$, such that for all $j \in \mathcal{M}$ we have

(i) $|Q(j)| \leq r + \delta - 1$.

(ii) $\mathcal{C}|_{Q(j)}$ is a code with minimum Hamming distance at least δ.

(iii) $\text{Rank}(\text{G}|_{\cup_{j\in\mathcal{M}}Q(j)}) = k$

Here $\mathcal{M}$ is the index set of local codes of the code $\mathcal{C}$. Note that if $\bigcup_{j\in\mathcal{M}} Q(j) = [n]$, then code is a code with (r,δ) all-symbol locality.

Theorem 2.9 ([8])**:** For any $[n,k,d]$ linear code over with (r,δ) information locality , the minimum Hamming distance is upper-bounded by

$$d \leq n - k + 1 - (\left\lceil \frac{k}{r} \right\rceil - 1)(\delta - 1). \tag{2.2}$$

Optimal codes with information locality for equation (2.1), equation (2.2) are constructed using the pyramid code construction [39] in [6, 8] respectively.

For both the classes of codes discussed so far, the locality r is the same for any code symbol. However, due to storage networks being non-homogeneous, interest in the case where different nodes have different localities has increased recently.

In some storage systems, there might be some important information symbols which are frequently accessed. This data is called as *hot data*. When a node containing hot data fails, it should be repaired very quickly. Hence, hot data should have smaller locality. Less important symbols can have a different locality. This can be achieved by having smaller locality for hot data and larger locality for other data.

Linear codes, in which, different subsets of symbols possess different values of locality are studied in [45]. They are referred to as *codes with unequal locality*. The unequal information locality is characterized by the notion of information locality profile [45].

Definition 2.10 (Information locality profile [45])**:** For a $[n,k,d]$ systematic code $\mathcal{C}$, the information locality profile of $\mathcal{C}$ is defined as a vector $\mathbf{K}_I$,

$$\mathbf{K}_I = (k_1, \cdots, k_r),\ r \leq k,\ 0 \leq k_j \leq k,\ \forall j \in [r].$$

Here k_j is the number of information symbols of locality j. The length of the vector $\mathbf{K}_I$ is r. Here $k_r \geq 1$ and $\sum_{j=1}^{r} k_j = k$.

Theorem 2.11 ([45]): For an $[n, k, d]$ linear code with information locality profile $\mathbf{K}_I = (k_1, \cdots, k_r)$, the minimum Hamming distance is upper bounded by

$$d \leq n - k - \sum_{l=1}^{j} \left\lceil \frac{k_j}{j} \right\rceil + 2. \tag{2.3}$$

The authors also define locality profile for unequal locality for all the code symbols and a upper bound on the minimum Hamming distance for unequal locality for all symbol is also derived.

Definition 2.12 (All-symbol locality profile [45]): For a $[n, k, d]$ systematic code $\mathcal{C}$, the all-symbol locality profile of $\mathcal{C}$ is defined as a vector $\mathbf{n}_p$,

$$\mathbf{n}_p = (n_1, \cdots, n_{\hat{r}}),\ \hat{r} \leq k,\ 0 \leq n_j \leq n, \forall j \in [\hat{r}].$$

The length of $n_{\hat{r}}$ is $\hat{r}$ and

$$\sum_{j=1}^{\hat{r}} n_{\hat{r}} = n.$$

Theorem 2.13 ([45]): For an $[n, k, d]$ linear code with all-symbol locality profile $\mathbf{n}_p = (n_1, \cdots, n_{\hat{r}})$, let

(1) $r\prime = \max\{1 \leq i \leq r' : \sum_{j=1}^{i}(n_j - \left\lceil \frac{n_j}{j+1} \right\rceil) < k\}$.

(2) $r = \min\{r\prime + 1 \leq j \leq \hat{r} : n_j \geq 2\}$.

Then the minimum Hamming distance of this code is upper-bounded by

$$d \leq n - k - \sum_{j=1}^{r-1} \left\lceil \frac{n_j}{j+1} \right\rceil - \left\lceil \frac{k - \sum_{j=1}^{i}(n_j - \left\lceil \frac{n_j}{j+1} \right\rceil)}{r} \right\rceil + 2. \tag{2.4}$$

Here the unequal locality is defined only for conventional locality i.e., locality with only one parity symbol. This is extended to (r, δ) locality in [46].

Cadambe and Mazumdar gave an upper bound on the minimum Hamming distance [47] that depends on the size of the alphabet being used.

LRC type	Upper bound	Comments
(r, d) code [6]	$d \leq n - k - \lceil \frac{k}{r} \rceil + 2.$	A bound for LRC's with single parity symbol
(r, δ) code [8]	$d \leq n - k + 1 - (\lceil \frac{k}{r} \rceil - 1)(\delta - 1).$	A bound for LRC's with (r, δ) locality.
LRC's with unequal locality [45]	$d \leq n - k - \sum_{j=1}^{r-1} \left\lceil \frac{n_j}{j+1} \right\rceil - \left\lceil \frac{k - \sum_{j=1}^{i} (n_j - \lceil \frac{n_j}{j+1} \rceil)}{r} \right\rceil + 2.$	This bound is for all symbol locality
Field size dependent bound [47]	$k \leq \min_{t \in \mathbb{Z}_+} [tr + k_{opt}^q(n - t(r+1), d)], \quad q = \lvert\mathcal{Q}\rvert,$ $k_{opt}^q(n, d) = \max(\frac{log\lvert C\rvert}{log\ q})$	Binary simplex codes are optimal.

Table 2.1: A summary of the literature for locally recoverable codes .

Definition 2.14 ([47])**:** For a code with length n, minimum Hamming distance d, over a field size q, its largest possible dimension is given by

$$k_{opt}^q(n, d) = \max(\frac{log|C|}{log\ q}).$$

The maximization is over all possible $n-$length codebooks, with minimum Hamming distance d, over some alphabet $\mathcal{Q}$, where $|Q| = q$.

The field size dependent upper bound [47] is given below.

Theorem 2.15 ([47])**:** For an $[n, k, d]$ code with locality r over alphabet $\mathcal{Q}$,

$$k \leq \min_{t \in \mathbb{Z}_+} [tr + k_{opt}^q(n - t(r+1), d)], \quad q = |\mathcal{Q}|. \tag{2.5}$$

The minimizing value of t, given by t^* should satisfy

$$t^* \leq \min \left\{ \left\lceil \frac{n}{r+1} \right\rceil, \left\lceil \frac{k}{r} \right\rceil \right\}.$$

It is shown that the binary Simplex codes satisfy this bound with equality [47].

Table 2.1 summarizes the different variations of LRC's, their parameters and their upper bounds on the minimum Hamming distance.

A family of optimal locally recoverable codes with all-symbol locality are constructed in [16]. This construction is based on polynomials. Construction of LRCs with information locality is studied in detail in [39]. LRCs are studied in [19, 48–51]. The connections between locally recoverable codes and matroid theory are discussed in [21]. Locally recoverable codes over smaller finite fields are constructed in [52–54]. Locally recoverable codes are defined for non-linear codes in [55]. It is shown that a systematic non-linear code obeys the same upper-bound on the minimum Hamming distance as given by equation (2.1). It is also shown that for optimal codes, the locality constraints must be reversible.

2.4.1 Optimal Code Constructions

***(A) Polynomial based Optimal All-symbol LRC's*:**

Construction 2.16 ([16])**:** The idea behind this construction is given first and then the construction is formally given. This is a generalization of Reed-Solomon codes.

In Reed-Solomon codes, the message polynomial $m(x)$ of degree $k-1$ is constructed by using the k message symbols $(m_0, \cdots, m_{k-1})$, where $m(x) = \sum_{i=0}^{k-1} m_i x^i$. This message polynomial is evaluated at n distinct points $(x_1, \cdots, x_n)$ and the evaluations are stored as a vector $(m(x_1), \cdots, m(x_n))$, which is the codeword. If a symbol in the codeword is lost, then using k other code symbols, the polynomial $m(x)$ is reconstructed using polynomial interpolation and the lost symbol is recovered by evaluating the polynomial at the evaluation point for that location.

The construction is given for $\delta \geq 2$. The minimum Hamming distance of the codes constructed using this construction is given by equation (2.2).

The encoding polynomial $f_e(x)$ is of degree $k-1+(\lceil \frac{k}{r} \rceil - 1)(\delta - 1)$. The decoding polynomial (for local decoding) is $d(x)$ and the degree of $d(x)$ is $r-1$. If a code symbol is lost in a local code, then using r other code symbols in the local code, the decoding polynomial $d(x)$ is constructed. This is obtained using polynomial interpolation. Using $d(x)$, the lost code symbol is recovered. This is local recovery.

For example, in Fig. 2.5, let $r = 2$. Let $a_1, a_2, a_3 \in \{x_1, \cdots, x_n\}$ and $f_e(a_1), f_e(a_2), f_e(a3)$ form a local code. Say, $f_e(a_1)$ is the code symbol that is lost in the code. Since locality is 2, the local decoding polynomial is of degree 1. So using two code symbols $d(a_2)$ and $d(a_3)$, the decoding polynomial $d(x)$

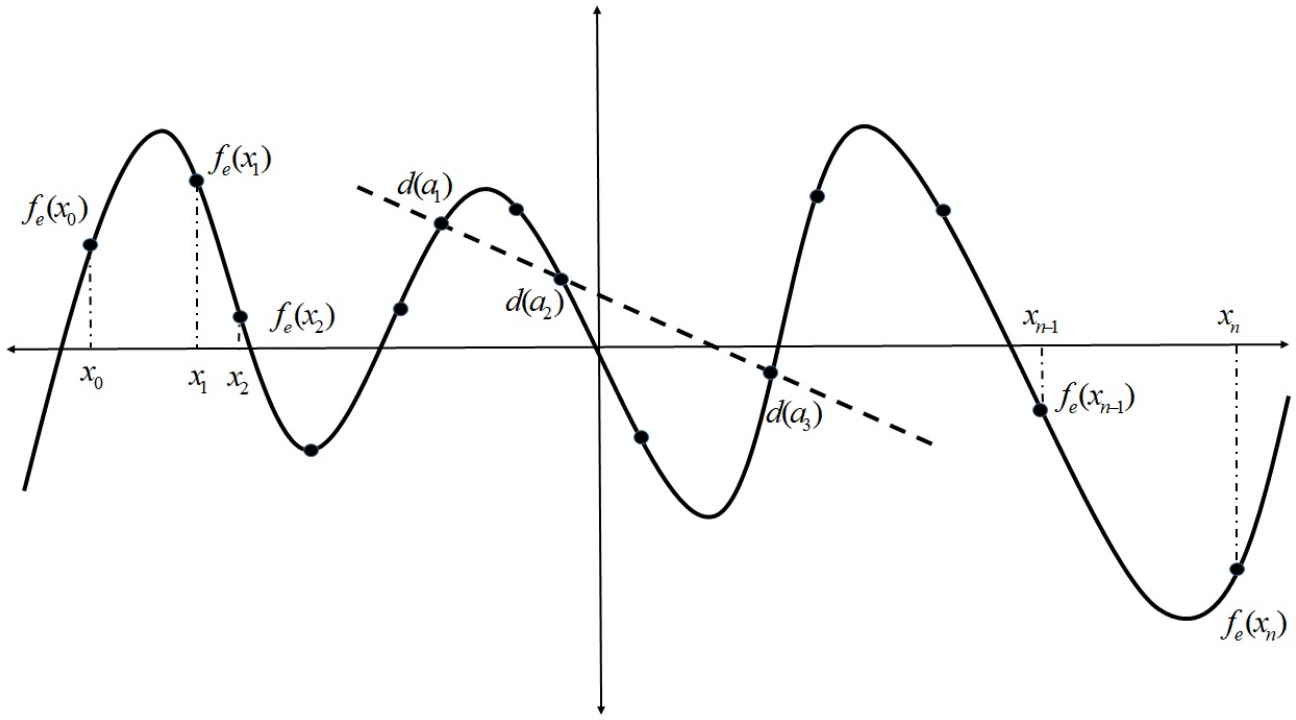

Figure 2.5: An example of a optimal code construction with all-symbol locality.

is re-constructed by polynomial interpolation. Obtain $f_e(a_1)$ as $d(a_1)$. This is because, due to the nature of this construction $d(a_l) = f_e(a_l),\ l = 1, 2, 3$.

LRCs with all symbol locality [16]: Choose n such that $r + \delta - 1 \mid n$ and $r \mid k$. Consider a set of points $\mathcal{A} \subset \mathbb{F}_q$ such that $|\mathcal{A}| = n$. Split $\mathcal{A}$ into $\frac{n}{r+\delta-1}$ subsets $\{\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_{\frac{n}{r+\delta-1}}\}$, such that $|\mathcal{A}_l| = r + \delta - 1,\ \forall l = 1, \cdots, \frac{n}{r+\delta-1}$. Here $\mathcal{A} = \biguplus_{l=1}^{\frac{n}{r+\delta-1}} \mathcal{A}_l$. A polynomial $g(x)$ is defined satisfying the following conditions.

(1) degree$(g(x)) = r + \delta - 1$

(2) Let $g(x)$ is constant for any $x \in \mathcal{A}_l$ i.e., $g(x_1) = g(x_2)$, for any $x_1, x_2 \in \mathcal{A}_l$.

The message vector is given by $\mathbf{m} = (m_{ij}, i = 0, \cdots, r-1,\ j = 0, \cdots, \frac{k}{r} - 1)$. The encoding polynomial is

$$f_e(x) = \sum_{i=0}^{r-1} f_i(x) x^i, \tag{2.6}$$

$$f_i(x) = \sum_{j=0}^{\frac{k}{r}-1} m_{ij} (g(x))^j. \tag{2.7}$$

The code $\mathcal{C}$ is the collection of the vectors

$$\mathcal{C} = \{(f_e(\alpha), \alpha \in \mathcal{A})\}. \tag{2.8}$$

Recovery of Erased symbol: Let $\alpha \in \mathcal{A}_j$ be the failed node. To recover a failed node $f_e(\alpha)$ locally, the decoding polynomial $d(x)$ is determined by considering any r evaluations of $f_e(x)$ from the set $\mathcal{A}_j \setminus \alpha$. The decoding polynomial is defined as

$$d(x) = \sum_{i=0}^{r-1} f_i(\alpha) x^i.$$

This $d(x)$ is determined using polynomial interpolation over these r points. The failed node is recovered as $d(\alpha)$. This is because, due to the nature of this code construction, for any $\alpha \in \mathcal{A}_j$,

$$d(\alpha) = f_e(\alpha).$$

Setting $\delta = 2$, will give the construction for an optimal code satisfying equation (2.1) with all-symbol locality.

Optimal codes with all symbol locality are constructed using Gabidulin codes in [50,51]. Gabidulin codes are introduced in [56]. Gabidulin codes are an example of rank-metric codes [57]. Decoding algorithms for Gabidulin codes are studied in [58].

(B) Construction of Optimal codes with all-symbol locality using Gabidulin codes [50, 51]:

Introduction to Gabidulin codes: Optimal codes with all-symbol locality are constructed in [50, 51] using Gabidulin codes. Consider a finite field $\mathbb{F}_{q^m}$, which is an extension field of the base field $\mathbb{F}_q$. For any $\alpha \in \mathbb{F}_{q^m}$, there exists a vector $(\alpha_1, \alpha_2, \cdots, \alpha_m)^T \in \mathbb{F}_q^m$ such that

$$\alpha = \sum_{j=1}^{m} \alpha_i b_i.$$

Here $b_1, \cdots, b_m$ are a fixed basis vectors of $\mathbb{F}_{q^m}$.

For a vector $\bar{v} = \{v_1, \cdots, v_N\} \in \mathbb{F}_{q^M}^N$, its matrix representation $\mathbf{V}$ of size $M \times N$ over $\mathbb{F}_q$ has the vector $(v_{i,1}, \cdots, v_{i,M})^T$ in its i^{th} column. The rank of the vector $\bar{v}$ is the rank of the matrix $\mathbf{V}$ over

$\mathbb{F}_q$. The *rank distance* for any two vectors $\bar{v}, \bar{u} \in \mathbb{F}_{q^M}^N$ is defined by

$$d_R(\bar{v}, \bar{u}) = \text{rank}(\mathbf{V} - \mathbf{U}).$$

The message vector $\bar{m} = (m_1, \cdots, m_K) \in \mathbb{F}_{q^M}^K$, is used to determine the encoding polynomial

$$f(z) = \sum_{i=1}^{K} m_i \ z^{q^{i-1}}.$$

Consider N linearly independent points $\{z_1, \cdots, z_N\} \subset \mathbb{F}_{q^M}$ over $\mathbb{F}_q$. The encoding polynomial is evaluated at these N points and the codeword is obtained as

$$\mathbf{c} = (f(z_1), f(z_2), \cdots, f(z_N)).$$

The generator matrix G_{gbd} for this Gabidulin code is

$$\text{G}_{gbd} = \begin{pmatrix} z_1 & z_2 & \cdots & z_N \\ z_1^q & z_2^q & \cdots & z_N^q \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{q^{K-1}} & z_2^{q^{K-1}} & \cdots & z_N^{q^{K-1}} \end{pmatrix}$$

This is split into two sub matrices

$$\text{G}_{gbd} = [\text{G}_{gbd}(1) \mid \text{G}_{gbd}(2)].$$

The size of $\text{G}_{gbd}(1)$ is $K \times K$ and the size of $\text{G}_{gbd}(2)$ is $K \times N - K$. An interesting property of the encoding polynomial here is that it is a linearized polynomial i.e.,

$$f(az_1) + f(bz_2) = af(z_1) + bf(z_2), \text{ where } z_1, z_2 \in \mathbb{F}_{q^M}, \ a, b \in \mathbb{F}_q.$$

Any K evaluations $f(\cdot)$ at K linearly independent points $\{z_1, \ldots, z_K\} \subset \{z_1, \ldots, z_N\}$ can used to determine the encoding polynomial $f(z)$ using polynomial interpolation. Hence Gabidulin codes are $[N, K, d_{gbd} = N - K + 1]$ MDS codes, where d_{gbd} is the minimum Hamming distance of this Gabidulin code.

Optimal LRC's using Gabidulin codes:

Construction 2.17 ([51]): This construction is given for the case where $r \mid k$ and $r \mid N$.

(1) Consider an $[N, K]$ Gabidulin code.

(2) Split the N code symbols into $\frac{N}{r}$ groups $N_1, \ldots, N_{\frac{N}{r}}$, each of size r.

(3) Encode the r code symbols in each $N_j,\ j \in [\frac{N}{r}]$ using $[r+\delta-1, r, \delta]$ MDS code over $\mathbb{F}_q$.

(4) The length of the LRC is $n = N + \frac{N}{r}(\delta - 1)$. This code is an optimal (n, k) code with (r, δ) locality.

This construction is described in the following figure.

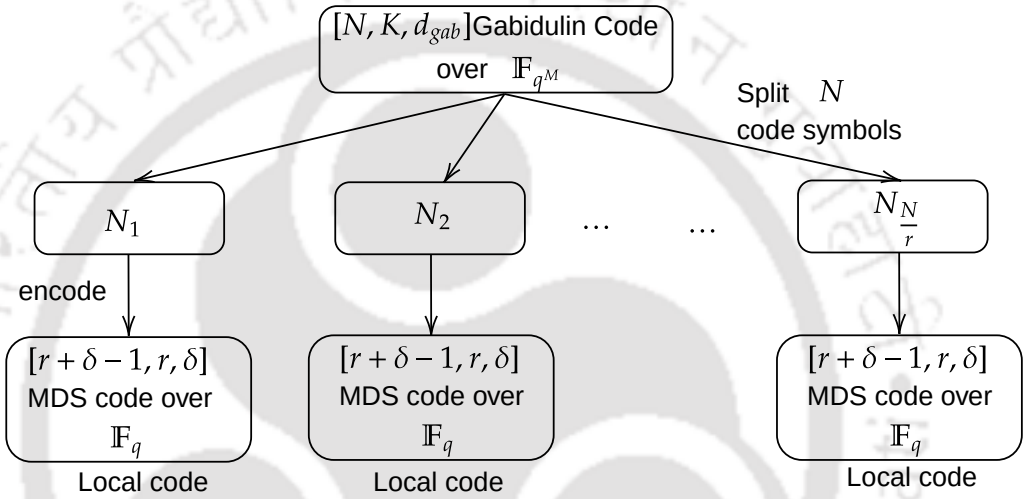

Figure 2.6: A description of construction 2.17

2.5 Locally Recoverable Codes with Availability

While the focus was on studying and constructing codes with locality, another property called "availability" had to be introduced. This is because, when a failed node is in repair, it could not be accessed by other users. From a practical point of view, the property of multiple parallel reads will be useful for distributed storage systems with *hot data*. This is studied in [9].

Availability is described using the example in Fig. 2.7. In this code $\{m_1, m_2, P_{L1}\}, \{m_3, m_4, P_{L2}\}$, $\{m_1, m_3, P_{L3}\}$,$\{m_2, m_4, P_{L4}\}$ form the four local codes. It can be observed that any information symbol can be recovered by two recovery sets simultaneously. For example, m_1 can be recovered by (m_2, P_{L1}) and (m_3, P_{L3}).

Definition 2.18 ((r, t) *availability* [9])**:** The i^{th} information symbol $c_i, i \in [k]$ of an $[n, k, d]$ code has (r, t) availability if there exists t subsets $Q_l(i) \subset [n]$ for $l = 1, 2, \cdots, t$ such that

(1) $i \in Q_l(i)$ and $|Q_l(i)| \leq r + 1$.

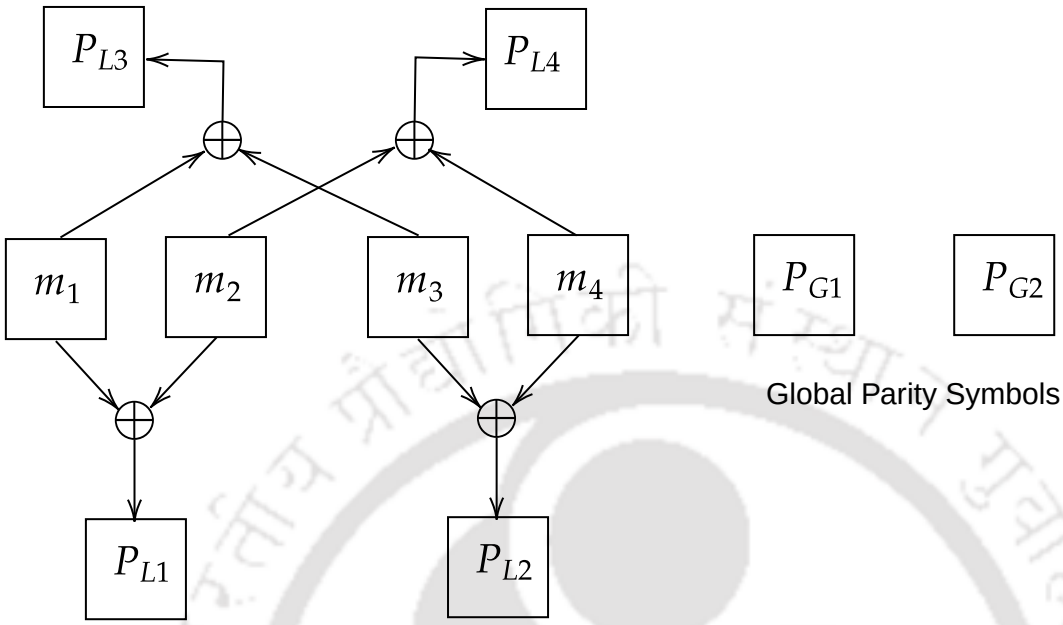

Figure 2.7: An example of a Locally Recoverable code with availability for the information symbols.

(2) $Q_l(i) \cap Q_{l'}(i) = \{i\}$, $1 \leq l \neq l' \leq t$.

(3) for each l, c_i is a function of code symbols indexed by $Q_l(i) \setminus \{i\}$.

An upper-bound on the minimum Hamming distance, availability and locality is given below.

Theorem 2.19 ([9])**:** Let $\mathcal{C}$ be a linear code with (r, t)-Availability, such that any recovery set for any information symbol has exactly 1 parity symbol. Then, the distance of the code is bounded as

$$d \leq n - k - \left\lceil \frac{kt}{r} \right\rceil + t + 1. \tag{2.9}$$

For both linear and non-linear codes with availability, without any assumption on the number of parity symbols in the local codes, an upper-bound on their minimum Hamming distance [9] is given as below.

Theorem 2.20 ([9])**:** For an $[n, k, d]$ code (linear or non-linear) with (r, t) availability, its minimum Hamming distance is upper-bounded by

$$d \leq n - k + 1 - \left(\left\lceil \frac{t(k-1)+1}{t(r-1)+1} \right\rceil - 1 \right). \tag{2.10}$$

Connections between matroid theory and codes with availability are explored in [21] and an upper-bound on the rate of code and its minimum Hamming distance is given.

Theorem 2.21 ([21])**:** For an $[n, k, d]$ code $\mathcal{C}$ with (r, t) availability, The rate is upper-bounded by

$$\frac{k}{n} \leq \frac{1}{\prod_{j=1}^{t}(1+\frac{1}{jr})}. \tag{2.11}$$

The minimum Hamming distance is upper-bounded by

$$d \leq n - k + 1 - \sum_{i=1}^{t} \left\lfloor \frac{k-1}{r^i} \right\rfloor \tag{2.12}$$

A class of codes [59] where the information symbols have t disjoint repair groups and each repair group is protected by a local code of minimum Hamming distance at least δ are studied in [59]. They give a bound for codes with availability only for information symbols as

$$d \leq n - \lceil k \rceil + 1 - \left(\left\lceil \frac{t(\lceil k \rceil - 1) + 1}{t(r-1)+1} \right\rceil - 1 \right)(\delta - 1) \tag{2.13}$$

For $n > k(t(r + \delta - 2) + 1))$, the authors prove the existence of an optimal code.

While so far, the studies focused on availability with a fixed locality, the authors in [60] studied codes with availability where the l^{th}, $1 \leq l \leq t$ disjoint recovery set for any information symbol has locality r_l. These are called as codes with unequal locality and availability.

Theorem 2.22 ([60])**:** For an $[n, k, d]$ linear code with unequal locality with availability, its minimum Hamming distance is upper-bounded by,

$$d \leq n - k + 1 - \left(\left\lceil \frac{t(k-1)+1}{\sum_{l=1}^{t}(r_l - 1) + 1} \right\rceil - 1 \right). \tag{2.14}$$

Codes similar to unequal locality are studied in [61]. Optimal codes for equation (2.9) with information locality are constructed in [9]. Optimal codes with all-symbol locality are constructed in [9]. This construction is based on Gabidulin codes. Optimal codes for equation (2.10) are given in [62]. Construction of codes with multiple disjoint recovery sets for all the symbols of the codeword using projective geometries are given in [63]. Availability for fountain codes is studied in [64]. Upper bounds on codes with availability are derived in [65–67].

2.6 Secret sharing schemes

Secret sharing was introduced by Shamir in [68]. In a secret sharing scheme, the secret s is split into n parts called shares. In a threshold secret sharing scheme, the secret can be reconstructed if and only if one has access to any k out of the n shares. The threshold secret sharing scheme is explained below. The encoding polynomial of the secret s is constructed as

$$f(x) = \sum_{i=0}^{k-1} z_i x^i.$$

The secret s is set as

$$z_0 = s.$$

The coefficients $z_1, \cdots, z_{k-1}$ are chosen at random from the finite field. This $f(x)$ is evaluated at n distinct points $\alpha_1, \cdots, \alpha_n$ from $\mathbb{F}_q$. The n shares are

$$f(\alpha_1), \cdots, f(\alpha_n).$$

To recover the secret, consider any k shares and then determine $f(x)$ by polynomial interpolation and obtain the secret by computing $f(0)$.

2.7 Distributed Storage Blockchain Systems

2.7.1 Introduction to Blockchain

Blockchain systems provide a cryptographically secure storage system for data storage using a hash chain [69]. Crypto-currencies like bitcoin [69, 70] are in use. In Blockchain, every peer in the blockchain stores the entire ledger. Blockchain finds applications apart from cryptocurrencies [71] and it has been implemented in medicine [72], supply chain management and global trade [73], government services [74]. These use smart contracts [75], which reduce the intermediaries in executing and monitoring a contract between parties. Blockchain is expected to revolutionize the financial sector [76].

The blockchain system is explained using Fig. 2.8. At any time instant τ, the system will store a set of transactions that took place between time $\tau - 1$ and τ . This set of transactions is denoted as B_τ. For each transaction in B_τ, its cryptographic hash function is calculated. The leaf nodes of a binary hash tree are labeled with the hash functions of each transaction. The non-leaf nodes will be labeled with the hash of the labels of its child nodes. Merkle root is the label of the tree top (called as root hash).

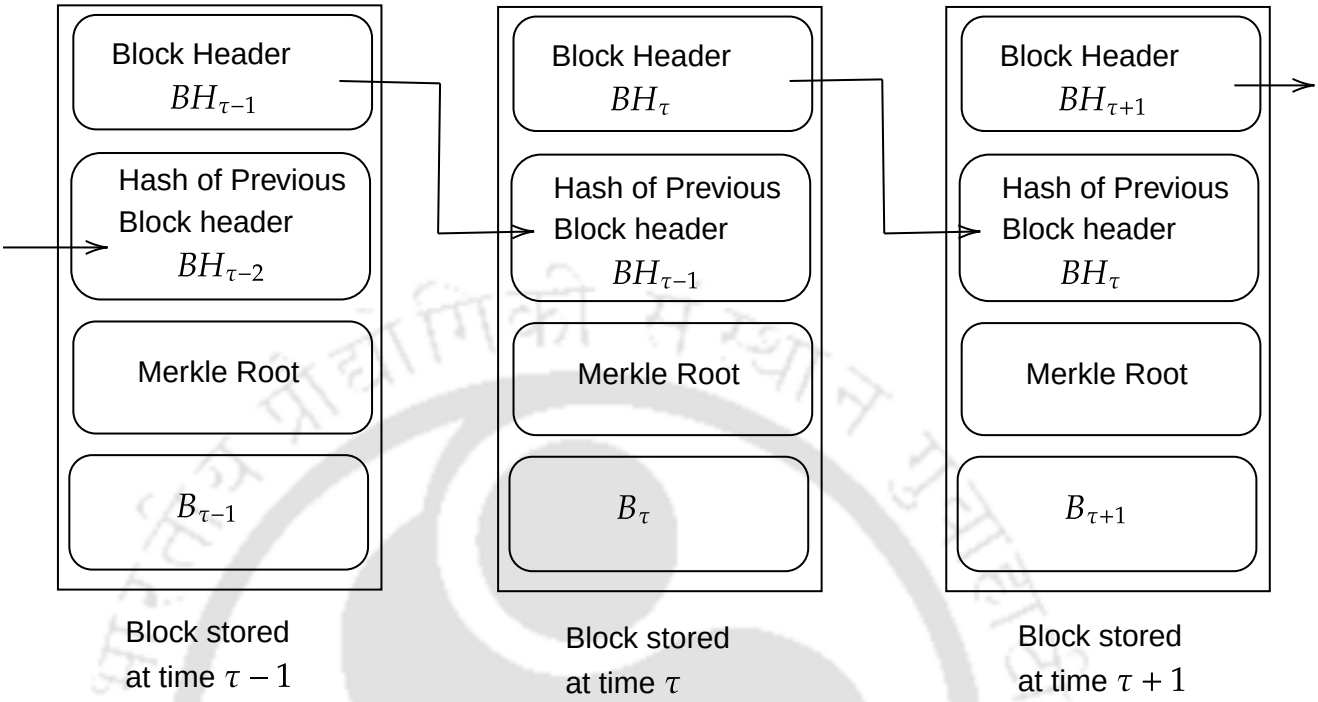

Figure 2.8: Blockchain structure

The block header for the block at time τ, denoted by BH_τ, is constructed. The hash of the previous block header $BH_{\tau-1}$ is computed and stored in the current block. So, the current block contains B_τ, the merkle root of B_τ, Block header B_τ, Hash of previous block header $BH_{\tau-1}$. The blockchain is illustrated in Fig. 2.8. Each node in the blockchain will store this entire block. So, the storage cost is high.

The efficiency of a blockchain system is measured by the following three parameters [12].

- The storage cost ($\mathbf{S}_c$) : It is the cost of storing the data. It is proportional to the amount of data stored in a node.
- Communication cost ($\mathbf{C}_c$): It is the cost of downloading the data from other nodes. It is proportional the amount of data downloaded.
- Robustness to peer failures ($\mathbf{R}_p$): It is the maximum number of nodes that are failed, after which the transaction can be recovered with the remaining nodes.

In a traditional blockchain system, let B_τ be the data block at time τ. Let $f_1(.), f_2(.)$ be two hash functions. Let $X_\tau = f_1(Z_\tau)$ be the hash value stored with the $(\tau+1)^{th}$ transaction. Let

$Z_\tau = (Z_{\tau-1}, f_2(B_\tau))$ be the concatenation of the previous hash value and a hash value of the current data block. It is assumed that $B_\tau \in \mathbb{F}_\Omega$ and $Z_\tau \in \mathbb{F}_q$. Any node stores B_τ and Z_τ. Hence

$$\mathbf{S}_c = \log_2 \Omega + \log_2 q.$$

To recover data at a failed node one needs to download the entire data stored at another node. Hence

$$\mathbf{C}_c = \log_2 \Omega + \log_2 q.$$

The data can still be accessible if there are $n-1$ node failures. Hence

$$\mathbf{R}_p = n - 1.$$

2.7.2 Distributed Storage blockchain

As storage cost is high in traditional blockchain system, the authors in [10, 11] propose a distributed storage blockchain. A distributed storage blockchain system with local secret sharing scheme is constructed in [12]. The notations are given below.

(1) B_τ: The data block which consists of all the transactions that will be stored in the block.

(2) $f_1(.), f_2(.)$: Two different hash functions.

(3) X_τ: Hash value stored in the $(\tau+1)^{th}$ transaction,

$$X_\tau = f_1(B_\tau).$$

(4) Z_τ: The concatenation of the previous hash value X_{t-1} and the hash value $f_2(B_\tau)$ of the current data block B_τ.

$$Z_\tau = (X_{\tau-1}, f_2(B_\tau)).$$

(5) $\mathbf{\Phi}$: Data encryption scheme.

(6) $K_\tau(l)$: The private key for the scheme $\mathbf{\Phi}$.

(7) a: The message vector, where $a = (a_{i,j}, i = 0, \ldots r-1, j = 0, \ldots, \frac{k}{r} - 1)$.

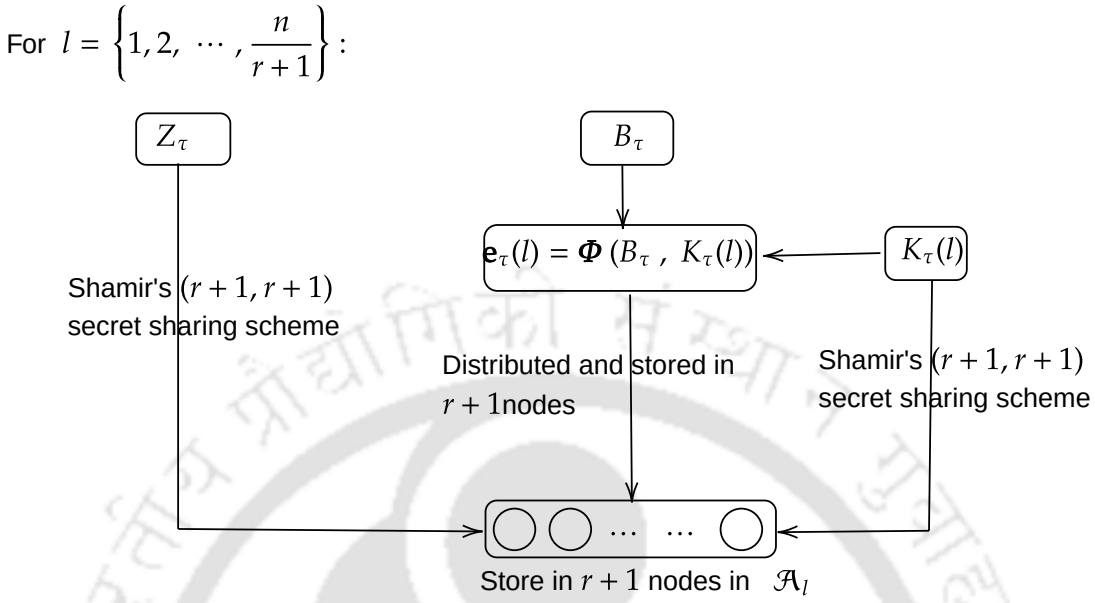

Figure 2.9: Distributed Storage Blockchain

For both the systems explained here, the encoding polynomial $z_e(x)$ is given by

$$z_e(x) = \sum_{i=0}^{r-1} y_i(x) x^i$$

where

$$y_i(x) = \sum_{j=1}^{\frac{k}{r}-1} a_{i,j} g(x)^j.$$

This polynomial is the encoding polynomial of optimal locally recoverable codes [16].

Distributed storage Blockchain [12]: The n nodes are divided into $l = \frac{n}{r+1}$ subsets $\mathcal{A}_1, \cdots, \mathcal{A}_{\frac{n}{r+1}}$ each of size $r+1$. At time τ, let B_τ be the data block that should be stored in the system. Let it be a vector i.e., $B_\tau \in \mathbb{F}_\Omega^{r+1}$. For each $\mathcal{A}_l$, generate a random private key $K_\tau(l), K_\tau(l) \in \mathbb{F}_q$. B_τ is encrypted using $K_\tau(l)$ to obtain the encrypted data $\mathbf{e}_\tau(l)$. This $\mathbf{e}_\tau(l)$ is a vector of size $r+1$. It is distributed and stored in the $r+1$ nodes of $\mathcal{A}_l$. The private key $K_\tau(l)$ for each $\mathcal{A}_l$ is encoded using Shamir's $(r+1, r+1)$ secret sharing scheme and is store in the in the $r+1$ nodes of $\mathcal{A}_l$. Similarly Z_τ, $Z_\tau \in \mathbb{F}_q$ is also encoded using Shamir's $(r+1, r+1)$ secret sharing scheme and is stored in the in the $r+1$ nodes of $\mathcal{A}_l$. This process is described in Fig. 2.9. The storage cost S_c is [12]

$$\mathbf{S}_c = \frac{\log_2 \Omega}{r+1} + 2 \log_2 q.$$

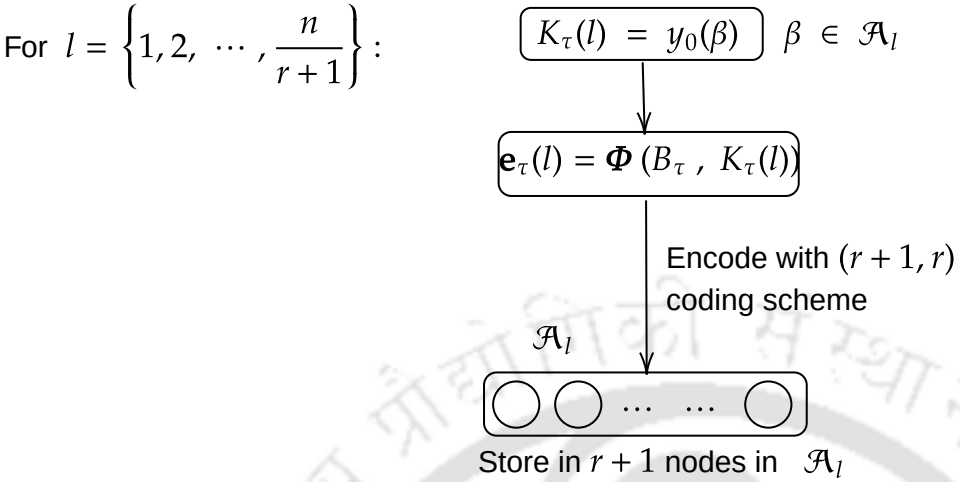

Figure 2.10: Storing local secret in DSB-LSS

This is because we store $K_\tau(l)$ and Z_τ along with $\mathbf{e}_\tau(l)$.

If a node fails then, one cannot access data using this subset of nodes, as the private key cannot be recovered. Then one has to access $r+1$ nodes in another group and download all the data stored in that node. This will be equal to $(r+1)\mathbf{S}_c + \zeta$, where ζ is the additional cost incurred to access another subset. Hence the communication cost in this system is

$$\mathbf{C}_c = \log_2 \Omega + 2(r+1)\log_2 q + \zeta.$$

If there is an erasure in each subset, then the data cannot be recovered. Hence, the data can be recovered in the presence of $\mathbf{R}_p$ erasures, where

$$\mathbf{R}_p = \frac{n}{r+1} - 1.$$

Distributed storage Blockchain with Local Secret sharing scheme (DSB-LSS) [12]: This scheme is designed to reduce the storage cost of Distributed storage Blockchain (DSB) system.

In this system as in DSB, the n nodes are divided into $l = \frac{n}{r+1}$ groups $\mathcal{A}_1, \cdots, \mathcal{A}_{\frac{n}{r+1}}$ each of size $r+1$. Set $a_{0,0} = Z_t$ as the global secret. For each $\mathcal{A}_l$, choose $\beta \in \mathcal{A}_l$ and set $y_0(\beta)$ as the local secret. Set the private key $K_\tau(l) = y_0(\beta)$. Obtain $\mathbf{e}_\tau(l) = \Phi(B_\tau, K_\tau(l))$. Here, the message vector is $B_\tau \in \mathbb{F}_\Omega^r$. Encode $\mathbf{e}_\tau(l)$ using an $(r+1, r)$ code and store the codeword in the $r+1$ nodes of $\mathcal{A}_l$. This is described in Fig. 2.10. The encoding polynomial $z_e(x)$ is evaluated at n points and they are stored in the n nodes of the system.

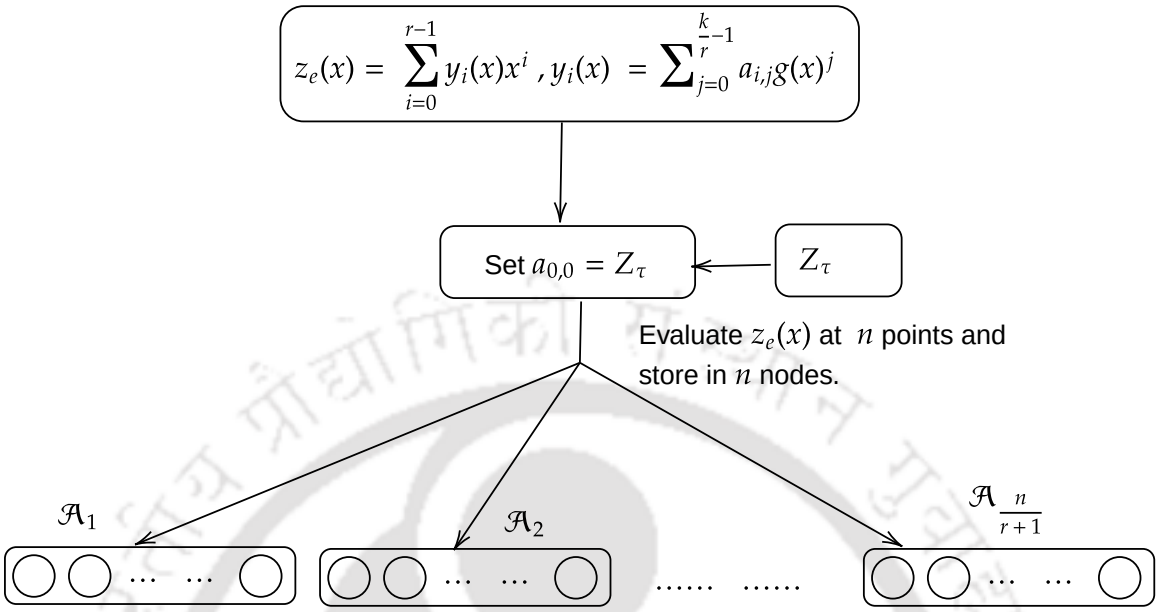

Figure 2.11: Storing global secret in DSB-LSS

Note that any $k + \frac{k}{r} - 1$ nodes can be used to determine the encoding polynomial $z_e(x)$ using polynomial interpolation. The global secret can then be recovered. This is described in Fig. 2.11. The storage cost of this system is given by [12]

$$\mathbf{S}_c = \frac{\log_2 \Omega}{r} + \log_2 q.$$

For this system, if a node fails, then one can access the remaining r nodes in the same subset and download all the data stored in that node. This will be equal to $r \cdot \mathbf{S}_c$. Hence the communication cost in this system is

$$\mathbf{C}_c = \log_2 \Omega + r \log_2 q$$

The additional cost ζ is not incurred here as one need not access the another subset. If there are 2 erasures in each subset, then the data cannot be recovered. Hence, the data can be recovered even in the presence of $\mathbf{R}_p$ erasures, where

$$\mathbf{R}_p = 2 \cdot \frac{n}{r+1} - 1.$$

3

Availability with Different Localities and Multiple Local Erasure Correction.

Contents

3.1 Introduction

In this chapter, we consider a new class of codes with availability. The l^{th}, $1 \leq l \leq t$ disjoint recovery set for any information symbol is protected by a local code with locality r_l and minimum Hamming distance at least δ_l. We derive two upper bounds on the minimum Hamming distance of these codes. These upper bounds are derived under different assumptions. Optimal codes are constructed achieving the bound with equality. This construction is based on pyramid code construction. We show that this code reduces the disk I/O when recovering more erasures locally. We also show that these optimal codes provide availability t for any information symbol even when there are any $\delta_l - 2$ code symbol erasures in each of the disjoint recovery set.

3.2 Motivation

The class of codes that we study in this chapter is motivated by the following two requirements:

(i) Recovery of multiple erasures locally with reduced disk reads:

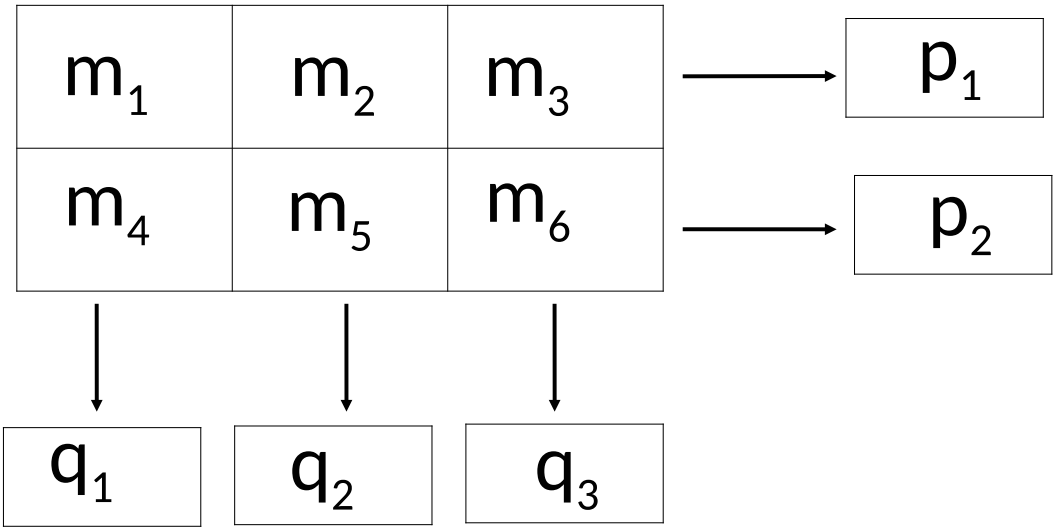

Figure 3.1: A code $\mathcal{C}_1$ with *availability* 2.

Consider the two codes $\mathcal{C}_1$ and $\mathcal{C}_2$ as shown in Fig. 3.1 and Fig. 3.2 respectively. For both the codes, there are 6 information symbols denoted by $m_1, \cdots, m_6$ and the remaining symbols are parity symbols.

Horizontal and vertical arrows shown in the figures indicate the local codes within the codes. In

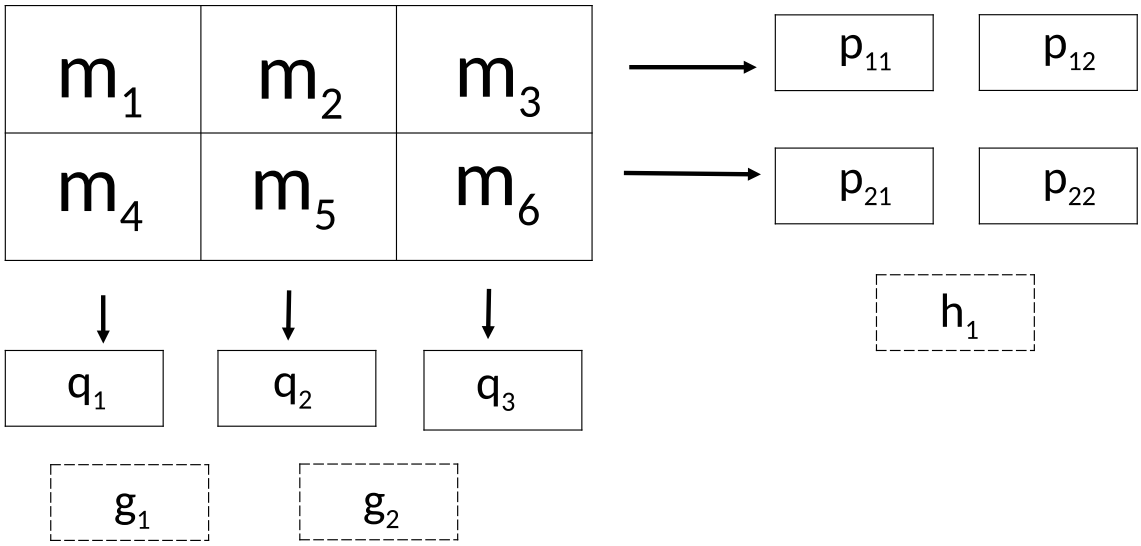

Figure 3.2: A code $\mathcal{C}_2$ with *availability* 2 and multiple local-erasures correction.

particular, for both codes $\mathcal{C}_1$ and $\mathcal{C}_2$, the information symbols m_1 and m_4 together with q_1 form a local code. Here q_1 is a local parity symbol. Similarly, the other two local codes that are common to $\mathcal{C}_1$ and $\mathcal{C}_2$ are formed by $\{m_2, m_5, q_2\}$ and $\{m_3, m_6, q_3\}$. Apart from these local codes, there are other local codes indicated by the horizontal arrows. For example, in the code $\mathcal{C}_1$, there are two more local parity symbols p_1, p_2 forming two local codes each with locality $r_1 = 3$ and local minimum Hamming distance $\delta_1 = 2$. Similarly, in the code $\mathcal{C}_2$, there are four more parity symbols $p_{11}, p_{12}, p_{21}, p_{22}$ forming two local codes with locality $r_2 = 3$ and local minimum Hamming distance $\delta_2 = 3$.

In addition to these local parity symbols, there are other parity symbols in the code $\mathcal{C}_2$, i.e., g_1, g_2 and h_1 to increase the minimum Hamming distance of the code. In the code $\mathcal{C}_2$, the set S_1 of symbols together with the remaining information symbols $\{m_1, m_4, m_5\}$ form an $[n_1 = 11, k = 6, d_1 = 4]$ code $\mathcal{C}_2^1$ with $(r_1 = 3, \delta_1 = 3)$ locality, where

$$S_1 = \{m_2, m_3, m_6, p_{11}, p_{12}, p_{21}, p_{22}, h_1\}.$$

Similarly, the set S_2 together with the remaining information symbols $\{m_2, m_3, m_6\}$ form an $[n_2 =$

$11, k = 6, d_2 = 4]$ code $\mathcal{C}_2^2$ with $(r_1 = 2, \delta_1 = 2)$ locality, where

$$S_2 = \{m_1, m_4, m_5, q_1, q_2, q_3, g_1, g_2\}.$$

Note that the sets S_1 and S_2 are disjoint. Furthermore, the codes $\mathcal{C}_2^1$ and $\mathcal{C}_2^2$ are optimal (r, δ) codes. Now we compare the performance of two codes $\mathcal{C}_1$ and $\mathcal{C}_2$ when multiple symbols are erased. Suppose the information symbols m_1, m_2, m_5 are erased. If the code $\mathcal{C}_1$ is used to repair these symbols, the total number of disk reads is 7. Note that m_1 is recovered by reading the symbols m_4 and q_1. Then, m_2 is recovered using $\{m_1, m_3, p_1\}$ and finally, m_5 is recovered using m_2 and q_2. On the other hand, the number of disk reads is 5 if $\mathcal{C}_2$ is used to repair these symbols. In this case, the symbols $\{m_1, m_2\}$ are recovered from $\{m_3, p_{11}, p_{12}\}$ and then, m_5 from $\{m_2, q_2\}$. This shows that $\mathcal{C}_2$ has advantage over $\mathcal{C}_1$. However this comes at the cost of increased storage overhead.

The codes that we consider can reduce the disk reads needed to recover more symbols locally while providing availability at the same time.

(ii) Availability in the presence of any $\delta_l - 2$ erasures: For the class of codes described in this chapter, it is possible to maintain availability property for any information symbol when some of the symbols in its disjoint repair groups are erased. Consider the code $\mathcal{C}_2$ in Fig 3.2. If the local parities p_{11} and p_{21} are erased, then it will still be possible to have availability 2 for all the information symbols. For example, m_1 can still be recovered using two disjoint set of symbols namely $\{m_2, m_3, p_{12}\}$ and $\{m_4, q_1\}$. This is possible as the local codes are MDS codes. The symbols m_1, m_2, m_3 along with p_{11}, p_{12} form an $[5, 3, 3]$ MDS code.

Note that code $\mathcal{C}_1$ does not have this property as loss of a parity symbol (say p_1) does not provide availability for all the information symbols.

In this example, the erasures are limited to the parity symbols only, as this example is a special case. A much general example is given in section 3.5, where the code provides availability in the presence of any $\delta_l - 2$ erasures in the local codes.

For the results obtained in this chapter, the notations used are introduced in section 2.2.

3.3 Codes with Availability and different localities: Bound-1

We introduce a new class of codes with information locality and information availability (i.e., availability is provided for the information symbols only). We derive an upper-bound on its minimum

Hamming distance. This bound is derived under two assumptions. In section 3.4, we derive another upper bound with different assumptions.

Codes with (r, δ) locality are defined in definition 2.8. The codes with (r, t) availability are defined in definition 2.18. We now introduce a class of codes called codes with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ *availability*.

Definition 3.1 ($\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ *availability*)**:** The code $\mathcal{C}$ is said to have $\{(r_1, \delta_1), \ldots, (r_t, \delta_t)\}$ availability if $\mathcal{C}$ has t distinct punctured sub-codes $\mathcal{C}_l, 1 \leq l \leq t$ each having (r_l, δ_l) information locality such that $\text{support}(\mathcal{C}_l) \cap \text{support}(\mathcal{C}_{l'}) = \mathcal{I}$ for all $l, l' \in [t], l' \neq l$. Furthermore, for each information symbol $c_i, i \in \mathcal{I}$, there exists at least t subsets $Q_l(i) \subset \text{support}(\mathcal{C}_l)$ for $l = 1, 2, \cdots, t$, such that

(1) $i \in Q_l(i)$ and $|Q_l(i)| \leq r_l + \delta_l - 1$,

(2) $\mathcal{C}|_{Q_l(i)}$ is a code with minimum Hamming distance at least δ_l,

(3) $Q_l(i) \cap Q_{l'}(i) = \{i\}\ \forall l \neq l'$.

(4) c_i is a function of code symbols indexed by $Q_l(i) \setminus \{i\}$.

Note that $Q_l(i)$ and $Q_l(i')$ need not be different for all $i \neq i'; i, i' \in \mathcal{I}$.

The definition says that there are t sub-codes each with (r_l, δ_l) locality for all $l = 1, 2, \ldots, t$. These sub-codes are defined over the same information symbols. The subcodes $\mathcal{C}_1, \ldots, \mathcal{C}_t$ are used to provide a property called information-set availability which is discussed later in the chapter.

Let $\Omega_l(i), i \in \mathcal{I}$ be the collection of all sets of the form $Q_l(j), j \in \mathcal{I}$ containing i, i.e.,

$$\Omega_l(i) = \{Q_l(j) | i \in Q_l(j), j \in \mathcal{I}\}.$$

$$\text{Define } \Omega(i) = \bigcup_{l=1}^{t} \Omega_l(i).$$

$|\Omega_l(i)|$ is the number of subsets for the i^{th} information symbol in the l^{th} sub-code $\mathcal{C}_l$ and $|\Omega(i)|$ is the total number of subsets for the i^{th} information symbol in the code $\mathcal{C}$. Note that $|\Omega_l(i)| \geq 1$ for all $i \in \mathcal{I}$ and for all $l \in [t]$. Therefore, $|\Omega(i)| \geq t$ for all $i \in \mathcal{I}$. Let $\omega(\mathcal{C})$ be the least of the size of the sets $\Omega(i)$ as i varies in $\mathcal{I}$. We have

$$\omega(\mathcal{C}) \geq t.$$

We now derive an upper bound on the minimum Hamming distance of these codes.

Theorem 3.2: Let $\mathcal{C}$ be a code with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ availability. Suppose

$$(1)\ \dim(\mathcal{C}|_{Q_l(i)}) = |Q_l(i) \cap \mathcal{I}| \quad \forall i \in \mathcal{I},\ \forall l \in [t], \tag{3.1}$$

$$(2)\ \omega(\mathcal{C}) = t. \tag{3.2}$$

Then, the minimum Hamming distance of the code is upper-bounded by

$$d \leq n - k + 1 - \sum_{l=1}^{t} \left(\left\lceil \frac{k}{r_l} \right\rceil - 1 \right)(\delta_l - 1). \tag{3.3}$$

Proof : The proof of the theorem is based on the techniques used in the chapter [8] to upper-bound the minimum Hamming distance of the (r, δ) codes.

Let G be the generator matrix of the code $\mathcal{C}$. For each l, using algorithm 1, a set $U_l \subset \text{support}(\mathcal{C}_l)$ is computed such that

$$\text{rank}(\text{G}|_{U_l}) \leq k - 1.$$

Furthermore, the sets $U_l, l = 1, 2, \cdots, t$ are determined in such a way that the columns of $\text{G}|_{U_l}$ span the same vector space, i.e.,

$$\text{col}(\text{G}|_{U_l}) = \text{col}(\text{G}|_{U_p}) \text{ for } l, p \in [t],$$

where $\text{col}(\text{G}|_{U_p})$ is the column space of the matrix $\text{G}|_{U_p}$. Hence,

$$\text{rank}(\text{G}|_{U}) \leq k - 1,$$

where $U = \bigcup_{l=1}^{t} U_l$. Using Lemma 2.5, we get the desired result.

Let $\widehat{\mathcal{M}}(l)$ denote the index set for the local codes in $\mathcal{C}_l$ of the form $\mathcal{C}|_{Q_l(i)}$. Using this index set, we describe the local codes as $\{\mathcal{C}_l^j\}_{j \in \widehat{\mathcal{M}}(l)}$ with support $\{S_l(j)\}_{j \in \widehat{\mathcal{M}}(l)}$, where $S_l(j) = Q_l(i)$ for some $i \in \mathcal{I}$. The cardinality of the set $\widehat{\mathcal{M}}(l)$ is at most k, the dimension of the code $\mathcal{C}$. As $\omega(\mathcal{C}) = t$, there exists an information symbol $c_m, m \in \mathcal{I}$ such that $|\Omega_l(m)| = 1$ for all $l \in [t]$. As $m \in Q_l(m)$, we note that there is no other subset $Q_l(i), i \in \mathcal{I}$ (different from $Q_l(m)$) such that $m \in Q_l(i)$. Let $\mathcal{M}(l) \subset \widehat{\mathcal{M}}(l)$ be the index set for all the local codes in $\mathcal{C}_l$ except the local code $\mathcal{C}|_{Q_l(m)}$. The cardinality of the set $\mathcal{M}(l)$ is one less than that of $\widehat{\mathcal{M}}(l)$. Let

$$W_l = \cup_{i \in \mathcal{M}(l)} S_l(i).$$

Note that

$$m \notin W_l \cap \mathcal{I} \quad \forall l. \tag{3.4}$$

Furthermore, $\beta_l < k$ for all l, where $\beta_l = \text{rank}(\text{G}|_{W_l})$. If not, then $v \in W_l \ \forall \ v \in \mathbb{F}_q^k$. Consider the vector $w = \text{G}|_{\{m\}}$, i.e.,, w is the column of G corresponding to the m-th information symbol, $m \in \mathcal{I}$. We have $w = \sum_{j=1}^{q} c_j v_j$, $\{v_j\}$ are the columns of $\text{G}|_{\mathcal{I}}$. As w is a column of $\text{G}|_{\mathcal{I}}$, then $w = v_j$ for some $j \in W_l \cap \mathcal{I}$. This contradicts Equation (3.4).

The algorithm starts with $l = 1$. For each l and for each $i \in \mathcal{M}(l)$, we define the vector space $V_i = \text{col}(\text{G}|_{S_l(i)})$. Corresponding to the j^{th} iteration ($j \leq J_l - 2$) of the algorithm, one can always pick an $i \in \mathcal{M}(l)$ such that $V_i \not\subset \text{col}(\text{G}|_{T_j})$ as long as

$$\text{rank}(\text{G}|_{T_j}) \leq \beta_l - 1.$$

Note that

$$\text{rank}(\text{G}|_{T_j}) = \beta_l, \text{ and } \text{rank}(\text{G}|_{T_j \bigcup Q_l(m)}) = k \text{ for } j = J_l - 1.$$

Finally, at the iteration $j = J_l$, a set of columns of G indexed by U_l is computed and the algorithm exits from the while loop. The value of l is increased by 1 and the above procedure takes place till l reaches t.

As any local code in $\mathcal{C}_l$ has dimension at most r_l, we have

$$J_l \geq \left\lceil \frac{k}{r_l} \right\rceil.$$

Note that

$$\begin{aligned}
\text{col}(\text{G}|_{W_l}) &= \bigoplus_{i \in \mathcal{M}(l)} \text{col}(\text{G}|_{S_l(i)}) = \bigoplus_{i \in \mathcal{M}(l)} \text{col}(\text{G}|_{S_l(i) \cap \mathcal{I}}). \\
\text{col}(\text{G}|_{U_l}) &= \text{col}(\text{G}|_{W_l}) \bigoplus \text{col}(\text{G}|_{(Q_l(m) \cap \mathcal{I}) \setminus \{m\}}) \\
&= \bigoplus_{i \in \mathcal{M}(l)} \text{col}(\text{G}|_{S_l(i) \cap \mathcal{I}}) \bigoplus \text{col}(\text{G}|_{(Q_l(m) \cap \mathcal{I}) \setminus \{m\}}) \\
&= \text{G}|_{(W_l \cup Q_l(m)) \cap \mathcal{I} \setminus \{m\}} = \text{col}(\text{G}|_{\mathcal{I} \setminus \{m\}}),
\end{aligned} \tag{3.5}$$

Algorithm 1 :Used in the Proof of Theorem 3.2

for $l = 1$ to t **do**
  Let $T_0 = \phi, j = 0$
  Let $W_l = \cup_{i \in \mathcal{M}(l)} S_l(i), \beta_l = \text{rank}(\text{G}|_{W_l})$
  while 2 **do**
    Pick any $i \in \mathcal{M}(l)$ such that $V_i \not\subset \text{col}(\text{G}|_{T_j})$
    if rank $(\text{G}|_{T_j \bigcup S_l(i)}) \leq \beta_l - 1$ **then**
      $j = j + 1$
      $T_j = T_{j-1} \bigcup S_l(i)$
    else if $\text{rank}(\text{G}|_{T_j \bigcup S_l(i)}) = \beta_l$ **then**
      $j = j + 1$
      $T_j = T_{j-1} \bigcup S_l(i)$
      Let $S' = (Q_l(m) \cap \mathcal{I}) \setminus \{m\}$
      $j = j + 1$
      $T_j = T_{j-1} \bigcup S'$
      BREAK
    end if
  end while
  $U_l = T_j, J_l = j$
end for
$U = \bigcup_{l=1}^{t} U_l$.

where

$$V_1 \bigoplus V_2 := \{v_1 + v_2 | v_1 \in V_1, v_2 \in V_2\}.$$

Here V_1 and V_2 are two sub-spaces of some vector space V. Note that the column space of $\text{G}|_{U_l}$ is not dependent on the index l. Therefore,

$$\text{col}(\text{G}|_{U_l}) = \text{col}(\text{G}|_{U_p}) \forall 1 \leq l, p \leq t.$$

Furthermore,

$$\text{Rank of col}(\text{G}|_{U_l}) = k - 1 \text{ for all } l \in [t]$$

$$\text{rank}(\text{G}|_U) = k - 1, \text{ where } U = \bigcup_{l=1}^{t} U_l.$$

Using the method given in the proof of the bound on the minimum Hamming distance of (r, δ)

codes [8], it follows that

$$|U_l| \geq k - 1 + (J_l - 1)(\delta_l - 1).$$

$$\text{Hence, } |U| \geq k - 1 + \sum_{l=1}^{t} \Big((J_l - 1)(\delta_l - 1) \Big)$$

$$\geq k - 1 + \sum_{l=1}^{t} \Big((\left\lceil \frac{k}{r_l} \right\rceil - 1)(\delta_l - 1) \Big).$$

Using Lemma 2.5, we have

$$d \leq n - k + 1 - \sum_{l=1}^{t} \Big((\left\lceil \frac{k}{r_l} \right\rceil - 1)(\delta_l - 1) \Big).$$

The assumption given by equation (3.1) plays an important role in the proof of the theorem. Without this assumption, deriving an upper-bound for the minimum Hamming distance of the code is an extremely hard problem. Note that this assumption reduces to the statement that there is exactly one parity symbol in each local code as assumed in deriving the upper-bound (see equation (2.9)) for the minimum Hamming distance of the (r, t) codes.

If any of the two assumptions (i.e., assumptions given by equations (3.1) and (3.2)) is relaxed, it may be possible to construct codes whose minimum Hamming distance exceed the bound given in theorem 3.2.

When $r_1 = \cdots = r_t = r$ and $\delta_1 = \cdots = \delta_t = 2$, we compare our bound with the bound [9] given by equation (2.9). When r divides k, both bounds give the same value. But when r does not divide k, we get

$$d \leq n - k + 1 - \left\lceil \frac{k}{r} \right\rceil t + t.$$

This bound will give a smaller value than the one given by equation (2.9) for this case.

3.3.1 Optimal Code Constructions

We now construct a family of codes achieving the bound with equality. This is based on the existence of a certain binary matrix denoted by $\mathbf{R}$. This matrix is known as membership matrix of a code with availability [9] [77]. These matrices are introduced below.

Proposition 3.3: Let $K = \{1, 2, \ldots, k\}$. We need a partition of K into $\lceil \frac{k}{r} \rceil$ number of subsets of K satisfying the following property:

If $r \mid k$, then K is partitioned into subsets $K_1, K_2, \ldots, K_{\frac{k}{r}}$ of size exactly r. If $r \nmid k$, split K into $a+1 = \lceil \frac{k}{r} \rceil$ subsets such that 'a' sets are of size exactly r and the remaining one set is of size exactly $k - ar$.

Assumption 3.4: Let $L_R = \{r_1, r_2, \ldots, r_t\}$. Let $k = a_l r_l + b_l$, $\forall l \in [t]$. There are t distinct partitions of K denoted by $K_1, K_2, \ldots, K_t$ where $K_l = \{K_{l,q(l)} | 1 \leq q(l) \leq \left\lceil \frac{k}{r_l} \right\rceil\}$. For any $i \in [k]$ and for any $l, l' \in [t], l \neq l'$, suppose i lies in both $K_{l,q(l)}$ and $K_{l',q(l')}$ for some $q(l), q(l') \in [\left\lceil \frac{k}{r_l} \right\rceil]$, then $K_{l,q(l)} \cap K_{l',q(l')} = \{i\}$. Construct a binary matrix $\mathbf{R}_l$ of size $k \times \left\lceil \frac{k}{r_l} \right\rceil$ such that $q(l)$-th column of $\mathbf{R}_l$ has ones in the positions indexed by $K_{l,q(l)}$ and zeros every where else. Let the membership matrix of the code be

$$\mathbf{R} = [\mathbf{R}_1 | \ldots | \mathbf{R}_t].$$

The $\left\lceil \frac{k}{r_l} \right\rceil$ columns of $\mathbf{R}_l$ are denoted as $\mathbf{R}_l(j)$, $1 \leq j \leq \left\lceil \frac{k}{r_l} \right\rceil$. Note that the construction of this membership matrix is an open problem. This membership matrix is defined below.

Definition 3.5: Let there be t binary matrices denoted by $\mathbf{R}_l$ of size $k \times \left\lceil \frac{k}{r_l} \right\rceil$ for $l = 1, 2, \cdots, t$. Construct a binary matrix $\mathbf{R}$ of size $k \times (\sum_{l=1}^{t} \left\lceil \frac{k}{r_l} \right\rceil)$ as

$$\mathbf{R} = [\mathbf{R}_1 | \cdots | \mathbf{R}_t]. \tag{3.6}$$

These matrices satisfy the following properties:

(1) each column of $\mathbf{R}_l$ has at most r_l ones,

(2) the support of all the columns of $\mathbf{R}_l$ give a partition of the set $[k]$,

(3) any two rows of $\mathbf{R}$ intersect in at most one position.

For example, the matrix $\mathbf{R}$ given by equation (3.8) satisfies the above properties. In general, there is no systematic construction of such matrices. When all the r_l are identical, this matrix is equivalent [9] to a certain combinatorial object called resolvable design.

3.3.1.1 Optimal codes with Information locality and availability

This construction allows us to construct codes with availability for the information symbols. The construction of the code $\mathcal{C}$ with $\{(r_1, \delta_1), \ldots, (r_t, \delta_t)\}$ availability is given below. Before that a binary operator which is used in the construction is defined.

A binary operator $\otimes$: Before describing the construction, we define a binary operator $\otimes$, which is used in the construction. For a positive integer n, let $[n] \coloneqq \{1, 2, \ldots, n\}$. Given two matrices A and B of size $\alpha \times \beta, \alpha \times \gamma$ respectively, the binary operation $\otimes$ between A and B as follows: $A \otimes B$ is another matrix C of size $\alpha \times (\beta\gamma)$. Let $i \in [\alpha], j_1 \in [\beta], j_2 \in [\gamma]$. We index the columns of C by the ordered pair (j_1, j_2). Let $C(i, (j_1, j_2)) = A(i, j_1)B(i, j_2)$ for all $i \in [\alpha], j_1 \in [\beta], j_2 \in [\gamma]$, where $A(m, n)$ is the (m, n)-th entry of A.

Construction 3.6: Step 1: Consider an $[n', k, d]$, $d \geq \sum_{l=1}^{t}(\delta_l - 1) + 1$ systematic MDS code over $\mathbb{F}_q$ with the generator matrix

$$\mathrm{G} = [I_k | \Lambda].$$

Here I_k is a $k \times k$ identity matrix and Λ is a $k \times (d-1)$ matrix.

Step 2: Partition Λ into sub-matrices as

$$\Lambda = [\Lambda_1 | \ldots | \Lambda_t | \Lambda'].$$

Each Λ_l, $1 \leq l \leq t$, is a $k \times (\delta_l - 1)$ matrix and Λ' is a $k \times (d - 1 - (\sum_{l=1}^{t}(\delta_l - 1)))$ matrix.

Step 3: Define $\Delta_l = \Lambda_l \otimes \mathbf{R}_l$, where the binary operation $\otimes$ is defined right before this construction.

Step 4: The generator matrix of the optimal code with information availability is given by

$$\widehat{\mathrm{G}} = \left[\begin{array}{c|c|c|c|c} I_k & \Delta_1 & \cdots & \Delta_t & \Lambda' \end{array} \right].$$

Let $\mathcal{C}$ be the code corresponding to the generator matrix $\widehat{\mathrm{G}}$. It is easy to verify that the code $\mathcal{C}$ is a code with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ availability where

$$n = k + d - 1 + \sum_{l=1}^{t}(\left\lceil \frac{k}{r_l} \right\rceil - 1)(\delta_l - 1)).$$

Let d_{lrc} be the minimum Hamming distance of the code $\mathcal{C}$. We now show that the code $\mathcal{C}$ is optimal, i.e.,

$$d_{lrc} = n - k + 1 - (\sum_{l=1}^{t}(\left\lceil \frac{k}{r_l} \right\rceil - 1)(\delta_l - 1)).$$

By comparing G and $\widehat{\mathrm{G}}$, the code generated by $\widehat{\mathrm{G}}$ has minimum Hamming distance at least d i.e.,

$d_{lrc} \geq d$. Using Theorem 3.2, we have

$$d_{lrc} \leq n - k + 1 - \sum_{l=1}^{t} \left(\left\lceil \frac{k}{r_l} \right\rceil - 1 \right)(\delta_l - 1).$$

But for this code we have

$$d = n - k + 1 - \sum_{l=1}^{t} \left(\left\lceil \frac{k}{r_l} \right\rceil - 1 \right)(\delta_l - 1).$$

Hence the minimum Hamming distance of this code is

$$d_{lrc} = d = n - k + 1 - \left(\sum_{l=1}^{t} \left(\left\lceil \frac{k}{r_l} \right\rceil - 1 \right)(\delta_l - 1)\right).$$

We give two examples of optimal codes that satisfy this bound with equality. In the first example all r do not divide k. In the second example all r divide k.

Example 3.7: Consider an $[n = 10, k = 5, d = 6]$ MDS code over $\mathbb{F}_{11}$. Using this code, we construct an $[n = 14, k = 5, t = 2, \{(r_1 = 3, \delta_1 = 3), (r_2 = 2, \delta_2 = 2)\}]$ LRC with distance $d = 6$ with $t = 2$.

Let G_{MDS} be the generator matrix of the MDS code given by

$$\mathrm{G}_{MDS} = \left[\begin{array}{c|cc|c|cc} & 1 & 5 & 4 & 2 & 4 \\ & 6 & 9 & 7 & 5 & 4 \\ I_5 & 10 & 1 & 5 & 5 & 1 \\ & 1 & 4 & 5 & 7 & 9 \\ & 5 & 4 & 2 & 4 & 5 \\ & \underbrace{\quad\quad} & _{\Lambda^1} & \underbrace{\quad} & _{\Lambda^2} & \underbrace{\quad\quad} \; _{\Lambda'} \end{array}\right].$$

Here the membership matrix $\mathbf{R}$ is

$$\mathbf{R} = [\mathbf{R}_1 | \mathbf{R}_2], \mathbf{R}_1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{R}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

The generator matrix G_{LRC} of the code is given by

$$\mathrm{G}_{LRC} = \left[I_5 | \Omega^1 | \Omega^2 | \Lambda'\right],$$

where,

$$\Omega^1 = \left[\begin{array}{cc|cc} 1 & 5 & 0 & 0 \\ 6 & 9 & 0 & 0 \\ 10 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 \\ \underbrace{0}_{\Lambda_1^1} & \underbrace{0}_{\Lambda_1^1} & \underbrace{5}_{\Lambda_2^1} & \underbrace{4}_{\Lambda_2^1} \end{array}\right], \Omega^2 = \left[\begin{array}{ccc} 4 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 5 \\ 5 & 0 & 0 \\ \underbrace{0}_{\Lambda_1^2} & \underbrace{2}_{\Lambda_2^2} & \underbrace{0}_{\Lambda_3^2} \end{array}\right].$$

The field size needed for these constructions is the field size of the underlying MDS codes.

Another example is given below. This gives the generator matrix of the code $\mathcal{C}_2$ which is described in Fig. 3.2.

Example 3.8: In this example, we construct an $[n = 16, k = 6, t = 2, \{(r_1 = 3, \delta_1 = 3), (r_2 = 2, \delta_2 = 2)\}]$ code with minimum Hamming distance $d = 7$. Let G be a generator matrix of an $[n = 12, k = 6, d = 7]$ MDS code over $\mathbb{F}_{13}$ given by

$$\mathrm{G} = \left[\begin{array}{c|cc|c|c|cc} & 12 & 7 & 5 & 9 & 4 & 8 \\ & 6 & 9 & 3 & 3 & 11 & 8 \\ & 11 & 7 & 6 & 8 & 11 & 1 \\ I_6 & 7 & 1 & 11 & 8 & 6 & 7 \\ & 11 & 8 & 11 & 3 & 3 & 9 \\ & \underbrace{6}_{\Lambda_1} & \underbrace{8}_{\Lambda_1} & \underbrace{4}_{\Lambda'(1)} & \underbrace{9}_{\Lambda_2} & \underbrace{5}_{\Lambda'(2)} & \underbrace{7}_{\Lambda'(2)} \end{array}\right]. \tag{3.7}$$

Here the membership matrix $\mathbf{R}$ is

$$\mathbf{R} = [\mathbf{R}_1|\mathbf{R}_2], \mathbf{R}_1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{R}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.8}$$

The generator matrix $\widehat{\mathrm{G}}$ of the code is given by

$$\widehat{\mathrm{G}} = \left[I_6|\Delta_1|\Lambda'(1)|\Delta_2|\Lambda'(2)\right],$$

where,

$$\Delta_1 = \begin{bmatrix} 12 & 7 & 0 & 0 \\ 6 & 9 & 0 & 0 \\ 11 & 7 & 0 & 0 \\ 0 & 0 & 7 & 1 \\ 0 & 0 & 11 & 8 \\ 0 & 0 & 6 & 8 \end{bmatrix}, \Delta_2 = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 8 \\ 8 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 9 \end{bmatrix}.$$

The field size needed for these constructions is the field size of the underlying MDS codes.

3.3.2 Codes with all-symbol locality using Gabidulin codes

We now provide a construction for codes that provide information availability and all symbol locality. The construction is based on Gabidulin codes. The only restriction is that $r_t \mid k,\ r_t \mid N$.

(A) Generator matrix of systematic Gabidulin codes:

First we transform the generator matrix of a Gabidulin code into systematic form. Then we give the construction of the codes. Let G_g be the generator matrix of a $[N, k]$ Gabidulin code.

$$\mathrm{G}_g = [\mathrm{G}_{g1} | \mathrm{G}_{g2}].$$

The sub-matrix G_{g1} is of size $k \times k$ and the sub-matrix G_{g2} is of size $k \times N - k$. We obtain a systematic generator matrix as below

$$\begin{aligned} \widehat{\mathrm{G}}_g = \mathrm{G}_{g1}^{-1}\mathrm{G}_g &= \mathrm{G}_{g1}^{-1}[\mathrm{G}_{g1}|\mathrm{G}_{g2}] \\ &= [I_k|\mathrm{G}_{gp}]. \end{aligned}$$

I_k is of size $k \times k$ and $\mathrm{G}_{gp} = \mathrm{G}_{g1}^{-1}\mathrm{G}_{g2}$ is of size $k \times N - k$.

The membership matrix given in equation (3.6) is used here. It is given as

$$\mathbf{R} = [\mathbf{R}_1 | \ldots | \mathbf{R}_t].$$

Construction 3.9: Step 1: Consider an $[N + \sum_{l=1}^{t-1}(\delta_l - 1), k]$ Gabidulin code. Let its generator

matrix be

$$\mathrm{G}_g = [\mathrm{G}_{g1}|\mathrm{G}_{g2}|\mathrm{G}_{g3}]$$

where G_{g1} is of size $k \times k$, G_{g2} is of size $k \times \sum_{l=1}^{t-1}(\delta_l - 1)$ and G_{g3} is of size $k \times (N-k)$.

Let the generator matrix in systematic form be

$$\widehat{\mathrm{G}}_{gbd} = [I_k|\Lambda|\Gamma]$$

where $\Lambda = \mathrm{G}_{g1}^{-1}\mathrm{G}_{g2}$ is of size $k \times \sum_{l=1}^{t-1}(\delta_l - 1)$ and $\Gamma = \mathrm{G}_{g1}^{-1}\mathrm{G}_{g3}$ is of size $k \times (N-k)$.

Step 2: Partition Λ into sub-matrices as

$$\Lambda = [\Lambda_1| \ldots |\Lambda_{t-1}].$$

Each Λ_l, $1 \leq l \leq t-1$, is a $k \times (\delta_l - 1)$ matrix.

Step 3: Construct a $k \times (\delta_l - 1)$ matrix $\Lambda_l(j)$, $1 \leq l \leq t-1$, $1 \leq j \leq \left\lceil \frac{k}{r_l} \right\rceil$ as follows: The rows indexed by $\mathbf{R}_l(j)$ in $\Lambda_l(j)$ are the entries from rows of Λ_l that are indexed by $\mathbf{R}_l(j)$. All others rows in $\Lambda_l(j)$ are zero. Define a $k \times (\left\lceil \frac{k}{r_l} \right\rceil (\delta_l - 1))$ matrix Ω_l as

$$\Omega_l = [\Lambda_l(1)| \ldots |\Lambda_l(Y_l)], \text{ where } Y_l = \left\lceil \frac{k}{r_l} \right\rceil.$$

Step 4: Let

$$\Omega = [\Omega_l| \ldots |\Omega_{t-1}].$$

Obtain the following matrix

$$\left[\begin{array}{c|c|c} I_k & \Omega & \Gamma \end{array} \right].$$

Step 5: Split I_k into $\frac{k}{r_t}$ groups as given by the columns $[\mathbf{R}_t(1), \ldots, \mathbf{R}_t(\frac{k}{r_t})]$ of $\mathbf{R}_t$. Similarly split the $N-k$ columns of Γ into $\frac{N-k}{r_t}$ groups each of size exactly r_t. The total number of local groups is $\frac{N}{r_t}$. Encode the symbols in each of this local code using an $[r_t + \delta_t - 1, r_t, \delta]$ code over $\mathbb{F}_q$. The parity columns for these local codes are denoted as $\omega_1, \ldots, \omega_{\frac{N}{r_t}}$. Let

$$\omega = [\omega_1| \ldots |\omega_{\frac{N}{r_t}}].$$

Step 6: The generator matrix of the code with all symbol locality is given by

$$\widehat{\mathrm{G}}_g^{lrc} = \left[\begin{array}{c|c|c|c} I_k & \Omega & \Gamma & \omega \end{array}\right].$$

We now give show that this construction meets the bound in equation (3.4). The proof is similar to the one given in [9].

Theorem 3.10: Let $\mathcal{C}$ be an optimal $[n, k, d]$ code with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ availability and all-symbol locality. The minimum Hamming distance of this code is

$$d = n - k + 1 - \sum_{l=1}^{t} \Big(\left\lceil \frac{k}{r_l} \right\rceil - 1 \Big)(\delta_l - 1).$$

Proof : We show that this code can correct at most $d-1$ erasures. Since $n = N + \frac{N}{r_t}(\delta_t - 1) + \sum_{l=1}^{t-1}(\frac{k}{r_l})(\delta_l - 1)$, we show that this code can correct at most any $d-1$ erasures, where

$$\begin{aligned} d - 1 &= N + \frac{N}{r_t}(\delta_t - 1) + \sum_{l=1}^{t-1}(\left\lceil \frac{k}{r_l} \right\rceil)(\delta_l - 1) \\ &\quad - k - \sum_{l=1}^{t-1} \Big(\left\lceil \frac{k}{r_l} \right\rceil - 1 \Big)(\delta_l - 1) - (\frac{k}{r_t} - 1)(\delta_t - 1) \\ &= N + \frac{N}{r_t}(\delta_t - 1) - k - (\frac{k}{r_t} - 1)(\delta_t - 1) + \sum_{l=1}^{t-1}(\delta_l - 1). \end{aligned}$$

Let code symbols of the codeword $\mathcal{C}$ be indexed from 1 to n. Let $\mathbf{I}, \mathcal{P}, \widehat{\mathcal{Y}}, \widehat{\mathcal{Z}}$ denote the set of indices of the systematic symbols (generated by the columns I_k of $\widehat{\mathrm{G}}_g^{lrc}$), global parity symbols (generated by the columns Γ of $\widehat{\mathrm{G}}_g^{lrc}$), the local parity symbols generated in steps 3,4 (generated by the columns Ω of $\widehat{\mathrm{G}}_g^{lrc}$), the local parity symbols generated in step 5 (generated by the columns ω of $\widehat{\mathrm{G}}_g^{lrc}$) respectively.

All the code symbols in $\widehat{\mathcal{Y}}$ are disjoint union (denoted by $\biguplus$) of $t-1$ partitions $\widehat{\mathcal{Y}}_1, \ldots, \widehat{\mathcal{Y}}_{t-1}$ of the code symbols i.e

$$\widehat{\mathcal{Y}} = \biguplus_{l=1}^{t-1} \widehat{\mathcal{Y}}_l.$$

The code symbols in any $\widehat{\mathcal{Y}}_l$ are disjoint union of $\delta_l - 1$ partitions $\widehat{\mathcal{Y}}_l^1, \ldots, \widehat{\mathcal{Y}}_l^{\delta_l - 1}$ i.e.,

$$\widehat{\mathcal{Y}}_l = \biguplus_{j=1}^{\delta_l - 1} \widehat{\mathcal{Y}}_l^j.$$

Any $\widehat{\mathcal{Y}}_l^j$ has $\left\lceil \frac{k}{r_l} \right\rceil$ code symbols in it. These $\left\lceil \frac{k}{r_l} \right\rceil$ code symbols are denoted by $y_{l,1}, \ldots, y_{l,\left\lceil \frac{k}{r_l} \right\rceil}$. We now

prove it using two cases:

Case (i): Consider the set of indices $\mathcal{E}$, where $\mathcal{E} = \mathbf{I} \cup \mathcal{P} \cup \widehat{\mathcal{Z}}$. Denote the code $\mathcal{C}$ restricted to the coordinates indexed by $\mathcal{E}$ as $\mathcal{C}_{\mathcal{E}}$. Note that $\widehat{\mathcal{Z}}$ is the local parity symbols generated in step 4 are for the code symbols given by

$$\mathbf{m} \cdot [I_k | \Gamma]$$

The same code symbols can be obtained by

$$\widetilde{\mathbf{m}} \cdot [\mathrm{G}_{g1} | \mathrm{G}_{g3}] \text{ where } \widetilde{\mathbf{m}} = \mathbf{m} \cdot \mathrm{G}_{g1}^{-1}.$$

This is precoding the message vector $\mathbf{m}$. From [51], the code $\mathcal{C}_{\mathcal{E}}$ is a optimal $[N, k]$ code with (r_t, δ_t) locality. So, the minimum Hamming distance of $\mathcal{C}_{\mathcal{E}}$ is given by [51]

$$d_{\mathcal{C}_{\mathcal{E}}} = N + \frac{N}{r_t}(\delta_t - 1) - k + 1 - (\frac{k}{r_t} - 1)(\delta_t - 1).$$

This code can recover the message symbols $\mathbf{m}$ even when there are at most $d_{\mathcal{C}_{\mathcal{E}}} - 1$ erasures, where

$$d_{\mathcal{C}_{\mathcal{E}}} - 1 = N + \frac{N}{r_t}(\delta_t - 1) - k - (\frac{k}{r_t} - 1)(\delta_t - 1).$$

This is because, even after $d_{\mathcal{C}_{\mathcal{E}}} - 1$ erasures of the code symbols, we are still left with at least k code symbols. These code symbols are evaluations of the encoding polynomial $f(\cdot)$ with $\widetilde{\mathbf{m}}$ as its coefficients at k linearly independent (in $\mathbb{F}_q^M$) points over $\mathbb{F}_q$. Using these recover $\widetilde{\mathbf{m}}$, then proceed to recover $\mathbf{m}$.

Case (ii): There exists $N + \frac{N}{r_t}(\delta_t - 1) - k - (\frac{k}{r_t} - 1)(\delta_t - 1) + \mathcal{X}, \mathcal{X} \leq \sum_{l=1}^{t-1}(\delta_l - 1)$ erasures in $\mathcal{C}_{\mathcal{E}}$ and $\sum_{l=1}^{t-1}(\delta_l - 1) - \mathcal{X}$ erasures in $\widehat{\mathcal{Y}}$. Note that $\sum_{l=1}^{t-1}(\delta_l - 1) - \mathcal{X}$ erasures, in the worst case will be in in $\sum_{l=1}^{t-1}(\delta_l - 1) - \mathcal{X}$ partitions in $\{\widehat{\mathcal{Y}}_1^1, \ldots, \widehat{\mathcal{Y}}_1^{\delta_1 - 1}, \ldots, \widehat{\mathcal{Y}}_{t-1}^1, \ldots, \widehat{\mathcal{Y}}_{t-1}^{\delta_{t-1} - 1}\}$. So, There will be $\mathcal{X}$ sets with out any erasures. Combine the $\left\lceil \frac{k}{r_l} \right\rceil$ code symbols in each of the $\mathcal{X}$ partitions. We have $\mathcal{X}$ evaluations of the encoding polynomial $f(\cdot)$ with $\widetilde{\mathbf{m}}$ as its coefficients.

In $\mathcal{C}_{\mathcal{E}}$, in the worst case, we shall have $k - \mathcal{X}$ evaluations of the encoding polynomial $f(\cdot)$. When we combine all of these code symbols, we have at least k evaluations of $f(\cdot)$. Hence, $\widetilde{\mathbf{m}}$ can be recovered and using that $\mathbf{m}$ can be recovered.

3.4 Codes with Availability and different localities: Result-2

We derive another upper bound for the codes defined in definition 3.1. It is derived with different assumptions than the assumptions considered in theorem 3.2.

Differences between Result-1 and Result-2:

- Result-1 is derived for any value of t.
- In Result-2, given availability t, there will be t' localities $(r_l,\ l \in [t'])$ that do not divide k and $t - t'$ localities $(r_l, l \in [t - t'])$ that divide k. We show that there is a limit on t'.

The upper bound in this section is also derived under an assumption, which is given below.

Assumption 3.11: For the code defined in definition 3.1, suppose $\dim(\mathcal{C}|_{Q_l(i)}) = |Q_l(i) \cap [k]|$ for all $i \in [k]$ and all $l \in [t]$.

As mentioned earlier, this assumption means the number of information symbols in any local code is exactly its dimension.

For each sub-code $\mathcal{C}_l$, let D^l be the union of the collection of subsets $Q_l(i)$ such that

$$\text{Rank}(\text{G}|_{D^l}) = k.$$

We now construct a $k \times J_l$ membership matrix $\mathbf{R}$ for $\mathcal{C}_l$. The $(i,j)^{th}$ entry in $\mathbf{R}_l$ is 1 if the information symbol $i,\ i \in [k]$ is in the j^{th} local code $j \in J_l$.

For an $[n, k, t, \{(r_1, \delta_1), \ldots, (r_t, \delta_t)\}]$ code $\mathcal{C}$, we have t codes i.e., $(r_1, \delta_1), (r_2, \delta_2), \cdots, (r_t, \delta_t)$ codes. We will have J_l local codes for an (r_l, δ_l) code satisfying the property mentioned above for each l. It follows that $J_l \geq \left\lceil \frac{k}{r_l} \right\rceil$.

Definition 3.12: Given a $k \times J$ binary matrix $\mathbf{R}$, Let Ω_i be the set of ones in each row $i,\ 1 \leq i \leq k$. Let

$$\omega_{mn}(\mathbf{R}) = min\{|\Omega_1|, |\Omega_2|, \ldots, |\Omega_k|\}.$$

So, $|\Omega_i| \geq \omega_{mn}(\mathbf{R}), \forall i \in [k]$. Let

$$\Omega_{min} = \{\Omega_i \mid |\Omega_i| = \omega_{mn}(\mathbf{R})\}.$$

we proceed to construct a membership matrix for the code defined in definition 3.1.

Lemma 3.13: Let a $k \times J_l$ binary matrix $\mathbf{R}_l$ be a membership matrix of the (r_l, δ_l) code $\mathcal{C}_l$ for $l = 1, 2, \cdots, t$. Let $J_l = \left\lceil \frac{k}{r_l} \right\rceil + \alpha_l$. Let $\mathbf{R}$ be the membership matrix of the code $\mathcal{C}$ given by $\mathbf{R} = [\mathbf{R}_1 | \mathbf{R}_2 | \dots | \mathbf{R}_t]$. Then, the minimum row weight of $\mathbf{R}$ is given by

$$\omega_{mn}(\mathbf{R}) \leq t + \frac{\sum_{l=1}^{t} \alpha_l r_l}{k} + \beta(\mathcal{C}), \quad \beta(\mathcal{C}) = \frac{\sum_{l=1}^{t} p_l}{k}.$$

Proof : From assumption 3.11, for any local code in $\mathcal{C}_l$, $\dim(\mathcal{C}|_{Q_l(i)}) = |Q_l(i) \cap [k]|$ for all $i \in [k]$ and all $l \in [t]$. Since $|Q_l(i) \cap [k]| \leq r_l$ for any local code in any $\mathcal{C}_l$, the number of ones in any column of $\mathbf{R}_l$ are at most r_l. If $\omega_{mn}(\mathbf{R})$ is the least number of ones in $\mathbf{R}$, then

$$\begin{aligned}
\omega_{mn}(\mathbf{R}) \cdot k &\leq \sum_{l=1}^{t} (J_l \cdot r_l) \\
\omega_{mn}(\mathbf{R}) &\leq \sum_{l=1}^{t} \frac{(J_l \cdot r_l)}{k} \\
&\leq \sum_{l=1}^{t} \left\lceil \frac{(\left\lceil \frac{k}{r_l} \right\rceil + \alpha_l) r_l)}{k} \right\rceil \\
&\leq \sum_{l=1}^{t} \left\lceil \frac{(\left\lceil \frac{k}{r_l} \right\rceil r_l - k + \alpha_l r_l)}{k} + 1 \right\rceil \\
&\overset{(a)}{\leq} t + \sum_{l=1}^{t} \left(\frac{p_l + \alpha_l r_l}{k} \right) \\
&\leq t + \frac{\sum_{l=1}^{t} \alpha_l r_l}{k} + \beta(\mathcal{C}).
\end{aligned}$$

Since $p_l = \left\lceil \frac{k}{r_l} \right\rceil r_l - k$, (a) follows.

We now give an upper bound on its minimum Hamming distance. Note that this bound is derived under assumption 3.11.

Definition 3.14: For the $[n, k, t, \{(r_1, \delta_1), \dots, (r_t, \delta_t)\}]$ code $\mathcal{C}$, We define a parameter $\beta(\mathcal{C})$ as follows:

$$\beta(\mathcal{C}) = \frac{\sum_{l=1}^{t} p_l}{k}, \text{ where } p_l = \left\lceil \frac{k}{r_l} \right\rceil r_l - k.$$

Before deriving a bound on the minimum Hamming distance for the codes defined in this chapter, we give a lemma which will be used.

Theorem 3.15: Let $\mathcal{C}$ be an $[n, k, t, \{(r_1, \delta_1), \dots, (r_t, \delta_t)\}, d]$ code with $0 \leq \beta(\mathcal{C}) < 1$. The minimum

Hamming distance of the code is upper-bounded by

$$d \leq n - k + 1 - \sum_{l=1}^{t} \Big(\left\lceil \frac{k}{r_l} \right\rceil - 1 \Big)(\delta_l - 1). \tag{3.9}$$

Proof : Let G be the generator matrix of the code $\mathcal{C}$. Availability is provided for the code symbols in $[k]$. For each sub code $\mathcal{C}_l$, obtain J_l subsets of $[n]$.

Let D^l be the collection of these local codes. These subsets should be picked up in such a way that

$$\text{Rank}(\text{G}|_{D^l}) = k.$$

Let $\widehat{D^l} = \bigcup_{S \in D^l} S$, and

$$\widehat{D} = \bigcup_{l=1}^{t} \widehat{D^l}, \text{ and } D = \bigcup_{S \in \widehat{D}} S = \bigcup_{l=1}^{t} \{ \bigcup_{S \in \widehat{D^l}} S \}. \tag{3.10}$$

The restriction of the set of columns of G to the set of columns indexed by D^l is given by $\text{G}|_{D^l}$. So,

$$|\text{G}|_{D^l}| \geq k + J_l(\delta_l - 1).$$

$$\text{And}, |\text{G}|_D| \geq k + \sum_{l=1}^{t} J_l(\delta_l - 1) \tag{3.11}$$

Equation (3.11) is due to definition 3.1, which says $\text{support}(\mathcal{C}_l) \cap \text{support}(\mathcal{C}_{l'}) = \mathcal{I}$ for all $l, l' \in [t], l' \neq l$.

Let c_m be the code symbol from $[k]$ that occurs in *least number of local codes* in $\widehat{D}$. Let

$$\Delta_m^l = \{S \in D^l | c_m \in S\}, \ \Delta_m = \bigcup_{l=1}^{t} \Delta_m^l$$

Note that

$$|\Delta_m| = \omega_{mn}(\mathbf{R})| \leq t + \frac{\sum_{l=1}^{t} \alpha_l r_l}{k} + \beta(\mathcal{C}), \tag{3.12}$$

$$|D| = \sum_{l=1}^{t} \Big(\left\lceil \frac{k}{r_l} \right\rceil + \alpha_l \Big) \tag{3.13}$$

$$\implies |D \setminus \Delta_m| \geq \sum_{l=1}^{t} \Big(\left\lceil \frac{k}{r_l} \right\rceil + \alpha_l \Big) - \Big(t + \frac{\sum_{l=1}^{t} \alpha_l r_l}{k} + \beta(\mathcal{C}) \Big) \tag{3.14}$$

$$\geq \Bigg(\sum_{l=1}^{t} \Big(\left\lceil \frac{k}{r_l} \right\rceil - 1 + \alpha_l - \left\lceil \frac{\alpha_l r_l}{k} \right\rceil \Big) \Bigg) - \beta(\mathcal{C}).$$

Note that $\left\lceil \frac{\alpha_l r_l}{k} \right\rceil \geq \frac{\alpha_l r_l}{k}$. Since c_m will be in at least one local code in each $\mathcal{C}_l$, we can delete one local code from each $\mathcal{C}_l$. For sub-codes $\mathcal{C}_l$ with $\alpha_l = 0$, no more local codes need to be deleted from $\mathcal{C}_l$. But for codes with $\alpha_l \neq 0$, $\lceil \frac{\alpha_l r_l}{k} \rceil$ local codes will only need to be deleted from these codes only. When $\alpha_l \neq 0$, we have $\alpha_l - \lceil \frac{\alpha_l r_l}{k} \rceil \geq 0$. Therefore, the remaining number of local groups is given by

$$|D \setminus \Delta_m| \geq \sum_{l=1}^{t} (\left\lceil \frac{k}{r_l} \right\rceil - 1). \tag{3.15}$$

The Equation (3.15) means that any sub-code $\mathcal{C}_l$ has at least $\left\lceil \frac{k}{r_l} \right\rceil - 1$ local codes in them. Note for any $D \in \Delta_m$, c_m is the only information symbol deleted and all other information symbols in it are not deleted. So, its dimension will be reduced by 1. And all the parity columns which are a function of c_m will also be deleted in these local codes.

For any (r, δ) code [8] with T as the set of columns in its generator matrix such that $\text{rank}(\text{G}|_T) \leq k-1$, it is given that

$$|T| \geq k - 1 + (\left\lceil \frac{k}{r} \right\rceil - 1)(\delta_l - 1).$$

This is because the $k-1$ information symbols remain and there will be parity columns from at least $\left\lceil \frac{k}{r} \right\rceil - 1$ local codes.

For the code considered in this chapter let T_l be the set of columns of the (r_l, δ_l) code $\mathcal{C}_l$ in G that have rank $k-1$. Let $T = \cup_{l=1}^{t} T_l$. The total number of columns of rank $k-1$ in the code is given by as

$$|\text{G}|_T| \geq k - 1 + \sum_{l=1}^{t} \left((\left\lceil \frac{k}{r_l} \right\rceil - 1)(\delta_l - 1) \right)$$

This is because the $k-1$ information symbols for each $\mathcal{C}_l$ are same and each $\mathcal{C}_l$ has at least $\lceil \frac{k}{r} \rceil - 1$ local codes. And they will have at least $(\left\lceil \frac{k}{r_l} \right\rceil - 1)(\delta_l - 1)$ parity symbols in them [8]. Using lemma 2.5, we get the result.

We shall derive a condition on when $0 \leq \beta(\mathcal{C}) < 1$. Since $\beta(\mathcal{C}) = \frac{\sum_{l=1}^{t} p_l}{k}$, we obtain $0 \leq \sum_{l=1}^{t} p_l < k$. We get,

$$\sum_{l=1}^{t} (\left\lceil \frac{k}{r_l} \right\rceil r_l - k) < k$$

$$\sum_{l=1}^{t} (\left\lceil \frac{k}{r_l} \right\rceil r_l) - tk < k$$

Let there be $t' \leq t$ localities r_l, $1 \leq l \leq t'$ such that $r_l \nmid k$. The remaining $t - t'$ localities r_l, $1 \leq l \leq t - t'$

are such that $r_l \mid k$. Then the above equation can be reduced to

$$\sum_{l=1}^{t} \left(\left\lceil \frac{k}{r_l} \right\rceil r_l \right) - tk + \sum_{l=1}^{t'} \left(\left\lceil \frac{k}{r_l} \right\rceil r_l \right) - t'k < k$$
$$\sum_{l=1}^{t'} \left(\left\lceil \frac{k}{r_l} \right\rceil r_l \right) - t'k < k$$
$$\sum_{l=1}^{t'} \left(\left\lceil \frac{k}{r_l} \right\rceil r_l \right) < (t'+1)k. \tag{3.16}$$

So for t' localities $r_l \nmid k$ that satisfy equation (3.16) and $t-t'$ localities $r_l \mid k$, we will have $0 \leq \beta(\mathcal{C}) < 1$. When all t localities $r_l \mid k$, then there will be no restriction on t as $t' = 0$.

If $r_l = r, \forall l \in [t']$, then we obtain

$$t' < \frac{z}{\lceil z \rceil - z}, \; z = \frac{k}{r}.$$

If the minimum Hamming distance is derived without the assumption 3.11, then the minimum Hamming distance may be increased.

3.4.1 Optimal Code Constructions

We give two different constructions: Construction 3.16 is for information availability. Construction 3.17 is for information availability with all symbol locality. The constructions are similar to The constructions are similar to construction 3.6 and construction 3.9.

3.4.1.1 Optimal code constructions with information availability

We construct optimal codes for both $r \mid k$ and $r \nmid k$. Choose t such that $0 \leq \beta(\mathcal{C}) < 1$.

Construction 3.16: The construction is similar to construction 3.6.

Example 3.7 is an example of this code. It can be observed that for the code considered in this example, $t = t' = 2$ and $\beta(\mathcal{C}) = 0.4$.

3.4.1.2 Codes with all-symbol locality using Gabidulin codes

Construction 3.17: The construction is similar to construction 3.9.

The storage overhead (the parity symbols for the local codes) for the optimal codes with information availability is given by

$$\sum_{l=1}^{t} \left(\left\lceil \frac{k}{r_l} \right\rceil (\delta_l - 1) \right).$$

	Result-1 (Section 3.3)	Result-2 (Section 3.4)
Availability t	For any value of t.	Restriction on t', no restriction on $t-t'$.
Upper bound	$d \leq n-k+1-\sum_{l=1}^{t}\left(\left\lceil \frac{k}{r_l} \right\rceil -1\right)(\delta_l - 1)$.	Same bound
Optimal code constructions	(i) Information availability. (ii) Information availability, all symbol locality.	(i) Information availability. (ii) Information availability, all symbol locality.

Table 3.1: A summary results obtained in this chapter with t' localities $r_l \nmid k, l \in [t']$, $t-t'$ localities $r_l \mid k, l \in [t-t']$

We provide a summary of the results obtained in this chapter in table 3.1.

3.5 Availability in the presence of erasures

In this section, we show that the optimal codes constructed in section 3.4 has the property to provide availability t (i.e. t disjoint recovery sets) for the information symbols even in the presence of erasures.

Consider the code in Fig. 3.3. This code has availability $t = 3$ and since it is as per definition 3.1, it has 3 sub codes $\mathcal{C}_1, \mathcal{C}_2$ and $\mathcal{C}_3$. This code is constructed using construction 3.6.

This is an optimal $[24, 6, 8]$ code with $\{(3, 4), (2, 3), (2, 3)\}$ locality. Any local code is an MDS code. The messages symbols are

$$m_1, m_2, m_3, m_4, m_5, m_6$$

In this code, the sub code $\mathcal{C}_1$ is a code with $(r_1 = 3, \delta_1 = 4)$ locality. The parity symbols in $\mathcal{C}_1$ are $p_{11}, p_{12}, p_{13}, p_{21}, p_{22}, p_{23}$. The code $\mathcal{C}_1$ has two local codes and each one is an $[6, 3, 4]$ MDS code. The two local codes are given by the symbols

$$\{m_1, m_2, m_3, p_{11}, p_{12}, p_{13}\}, \{m_4, m_5, m_6, p_{21}, p_{22}, p_{23}\}$$

The sub code $\mathcal{C}_2$ is a code with $(r_2 = 2, \delta_2 = 3)$ locality. The parity symbols in $\mathcal{C}_2$ are $q_{11}, q_{12}, q_{21}, q_{22}$, q_{31}, q_{32}. The code $\mathcal{C}_2$ has three local codes and each one is an $[4, 2, 3]$ MS code. The three local codes

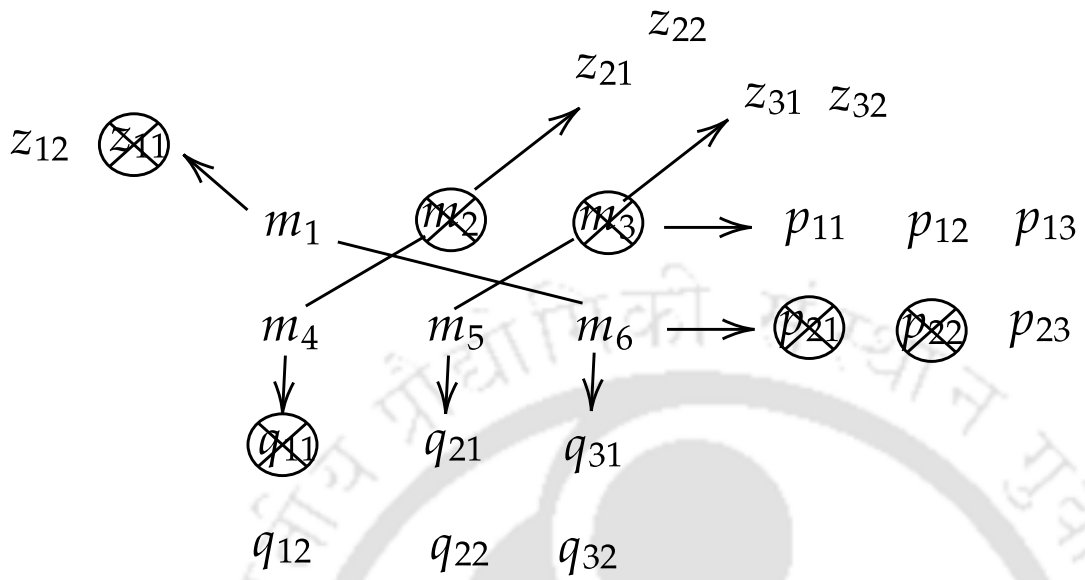

Figure 3.3: A code with Availability 3

in $\mathcal{C}_2$ are given by the symbols

$$\{m_1, m_4, q_{11}, q_{12}\}, \{m_2, m_5, q_{21}, q_{22}\}, \{m_3, m_6, q_{31}, q_{32}\}.$$

The sub code $\mathcal{C}_3$ is a code with $(r_3 = 2, \delta_3 = 3)$ locality. The parity symbols in $\mathcal{C}_3$ are $z_{11}, z_{12}, z_{21}, z_{22}$, z_{31}, z_{32}. The code $\mathcal{C}_3$ has three local codes and each one is an $[4, 2, 3]$ MDS code. The three local codes are given by the symbols

$$\{m_1, m_6, z_{11}, z_{12}\}, \{m_2, m_4, z_{21}, z_{22}\}, \{m_3, m_5, z_{31}, z_{32}\}.$$

The erasures are denoted by $\bigotimes$. For any local code in $\mathcal{C}_l$, it has exactly $\delta_l - 2$ erasures for $l = 1, 2, 3$. Since this code is an optimal code any information symbol will be in exactly one local code in $\mathcal{C}_l, l = 1, 2, 3$.

For example, the three local codes containing m_1 are

$$\{m_1, m_2, m_3, p_{11}, p_{12}, p_{13}\}, \{m_1, m_4, q_{11}, q_{12}\}, \{m_1, m_6, z_{11}, z_{12}\}$$

There are two erasures in $\{m_1, m_2, m_3, p_{11}, p_{12}, p_{13}\}$. The erased code symbols are m_2, m_3. Using any three of the remaining symbols (say p_{11}, p_{12}, p_{13}), the message vector for this local code m_1, m_2, m_3 can be recovered. Obtain m_1 from the message vector.

There is a single erasure in $\{m_1, m_4, q_{11}, q_{12}\}$. The erased symbol is q_{11}. Using two of the remaining symbols (say m_4, q_{12}), the message vector for this local code m_1, m_4 can be recovered. Obtain m_1 from the message vector.

There is a single erasure in $\{m_1, m_6, z_{11}, z_{12}\}$. The erased symbol is z_{11}. Using two of the remaining symbols (say m_6, z_{12}), the message vector for this local code m_1, m_6 can be recovered. Obtain m_1 from the message vector. So, even in the presence of erasures, there are $t = 3$ disjoint recovery sets

This can be done for any information symbol. In general, in any local code of $\mathcal{C}_l$, using r_l of the remaining $r_l + 1$ local code symbols recover all the message symbols in the local code. Then proceed to obtain the required message symbol. This way t disjoint recovery sets are provided even in the presence of erasures.

If $\delta_l = 2$, for some $l \in [t]$ (at least one δ_l), then all the erasures in all the local codes of all the sub codes $\mathcal{C}_1, \ldots, \mathcal{C}_t$ will be in the parity symbols only. This is explained below.

In the code $\mathcal{C}$, let a sub code $\mathcal{C}_{l'}$ for some $l' \in [t]$ has $\delta_{l'} = 2$. Hence, there should be $\delta_{l'} - 2 = 0$ erasures in it. If any other sub code $\mathcal{C}_l, l \neq l', l \in [t]$ with $\delta_l > 2$ has erasures in the message symbols, then these symbols are also erased in $\mathcal{C}_{l'}$, which should not happen. This is because for any $\mathcal{C}_l, \mathcal{C}_{l''}$, $l, l'' \in [t]$, the common symbols are k information symbols. So, for any sub-code $\mathcal{C}_l, l \neq l', l \in [t]$ all of its erasures should be in its parity symbols. The code described using Fig 3.2 falls in this category.

Note that, this property holds for codes with all-symbol locality, which are constructed using Gabidulin codes. This is because the local codes are MDS codes in this construction.

3.6 Codes with Information-set availability

Suppose there are two users who want to access all the information symbols at the same time. We assume that each codeword symbol is stored at one server and at any time a server can cater to one request only. Keeping this in mind, it is still possible to serve both the users at the same time using code $\mathcal{C}_2$. This code is described in Fig 3.4 with two recovery sets for all the information symbols. This code is the same code $\mathcal{C}_2$ described in Fig 3.2. We request the reader to refer to section 3.2 for the description of codes $\mathcal{C}_1$ and $\mathcal{C}_2$.

User 1 and user 2 are given access to the set of symbols S_1 and S_2 respectively (which are defined in section 3.2). Note that both user 1 and user 2 can recover the remaining information symbols as both the codes $\mathcal{C}_2^1$ and $\mathcal{C}_2^2$ have minimum Hamming distance 4.

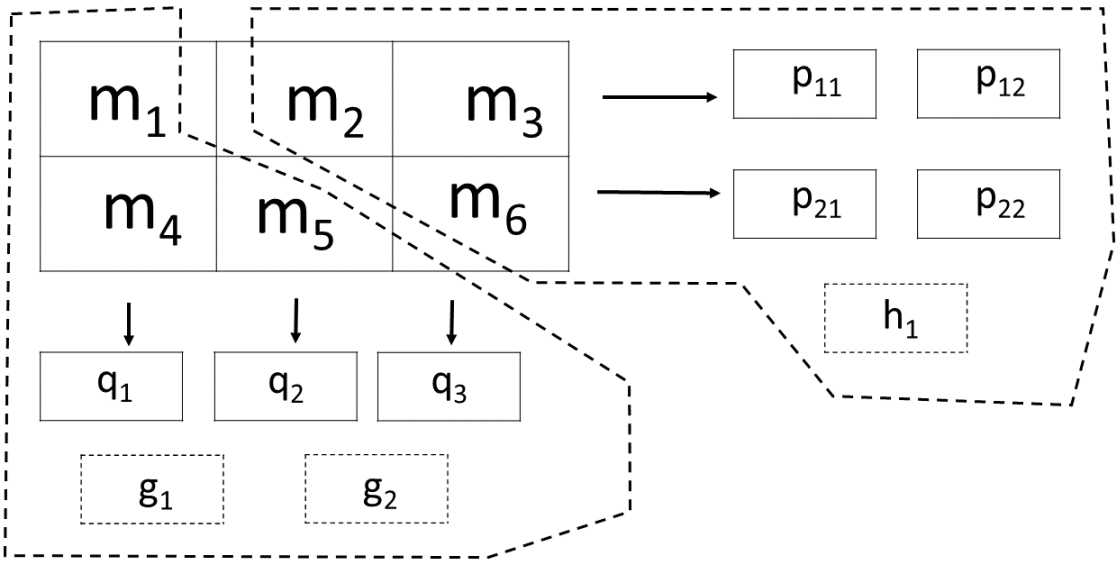

Figure 3.4: A code with *availability* 2 and *information-set availability* 2.

Thus using $\mathcal{C}_2$, it is possible to access the 6 information symbol set in two different ways simultaneously. This property is referred to as *information-set availability*. Note that $\mathcal{C}_1$ (described by Fig 3.1) does not have this property.

Definition 3.18 (*Information-set availability*)**:** An $[n, k, d]$ linear code $\mathcal{C}$ has information-set availability $t_{\mathcal{I}}$ if the set of information symbols can be accessed simultaneously $t_{\mathcal{I}}$ times.

For the code $\mathcal{C}_2$ described in Fig 3.4, the 6 information symbols $m_1, m_2, m_3, m_4, m_5, m_6$ can be accessed $t = 2$ times simultaneously. Suppose there are two users who want to access all the information symbols at the same time. We assume that each codeword symbol is stored at one server and at any time a server can cater to one request only. Keeping this in mind, it is still possible to serve both the users at the same time using code $\mathcal{C}_2$. User 1 and user 2 are given access to the set of symbols S_1 and S_2 respectively. Note that both user 1 and user 2 can recover the remaining information symbols as both the codes $\mathcal{C}_2^1$ and $\mathcal{C}_2^2$ have minimum Hamming distance 4.

We present the following result for systematic codes.

Theorem 3.19: Let $\mathcal{C}_l$ be an $[n_l, k, d_l]$ systematic code with (r_l, δ_l) locality and its generator matrix be $\mathrm{G}_l = [I_k|\mathrm{P}_l]$ for $l = 1, 2, \cdots, t$, where I_k is an identity matrix of order k. Consider the systematic code $\mathcal{C}$ with generator matrix $\mathrm{G} = [I_k|\mathrm{P}_1|\mathrm{P}_2|\cdots|\mathrm{P}_t]$. Suppose $d_l \leq k$ for $l = 1, 2, \cdots, t_k$ and $d_l \geq k+1$

for $l = t_k + 1, t_k + 2, \cdots, t$. Assume that $d_1 \geq d_2 \geq \cdots \geq d_{t_k}$. Then the information-set availability of $\mathcal{C}$ is given by

$$t_{\mathcal{I}} = p + t - t_k$$

where p is the largest positive integer not exceeding t_k satisfying the relation

$$\sum_{j=1}^{p} (d_j - 1) \geq (p-1)k.$$

Proof : If a code $\mathcal{C}_l$ has minimum Hamming distance $d_l \geq k+1$, then even without accessing the k information symbols, all the k information symbols can be recovered by this code. As there are $(t - t_k)$ such codes, it is possible to achieve information-set availability $t - t_k$ using these codes only.

Let p be an integer less than or equal to t_k. We want to achieve availability p using the codes $\mathcal{C}_l, l = 1, 2, \cdots, p$. It is done as follows: we distribute the $k' \leq k$ information symbols among these p codes such that the code $\mathcal{C}_l$ has access to only k_l information symbols and any information symbol can be accessed by at most one of the p codes. If $k - k_l = d_l - 1$, then the code $\mathcal{C}_l$ can recover the remaining $k - k_l$ information symbols. It follows that $k_1 + k_2 + \ldots + k_p = k' \leq k$ and $k_l = k - (d_l - 1)$ for $l = 1, 2, \cdots, p$. We have

$$\sum_{l=1}^{p} (k - (d_l - 1)) \leq k, \text{ i.e., } \sum_{l=1}^{p} (d_l - 1) \geq (p-1)k.$$

It follows that the $t - t_k$ sub-codes $\mathcal{C}_l$ with $d_l > k$, for $l = t_k + 1, \cdots, t$ and the p sub-codes $\mathcal{C}_l$ with $d_l \leq k$, for $l = 1, 2, \cdots, p$, $p \leq t_k$ can recover all the information symbols. Therefore, the information-set availability is

$$t_{\mathcal{I}} = p + t - t_k.$$

We now describe how the code given in example 3.8 allows all the $k = 6$ information symbols to be accessed simultaneously 2 times.

Example 3.20: Consider the code with generator matrix $\widehat{\mathrm{G}}$ (described in Example 3.8). This is a generator matrix for the code $\mathcal{C}_2$ described in Fig. 3.2. For the codeword $c = (c_1, c_2, \cdots, c_{16})$, the first 6 symbols $\{c_1, c_2, \cdots, c_6\}$ are information symbols. Consider the two sub-matrices $\widehat{\mathrm{G}}(1), \widehat{\mathrm{G}}(2)$ of $\widehat{\mathrm{G}}$

given by

$$\widehat{\mathrm{G}}(1) = \left[I_6|\Delta_1|\Lambda'(1)\right],$$
$$\widehat{\mathrm{G}}(2) = \left[I_6|\Delta_2|\Lambda'(2)\right].$$

Note that $\widehat{\mathrm{G}}(1)$ is the generator matrix of an $[n_1 = 11, k = 6, d_1 = 4]$ code with $(r_1 = 3, \delta_1 = 3)$ locality. Its codeword has the following symbols: $(c_1, c_2, \cdots, c_6, c_7, c_8, c_9, c_{10}, c_{11})$. As $d_1 = 4$, all the 6 information symbols can be recovered in the presence of $d_1 - 1 = 3$ erasures. Hence using

$$S_1 = \{c_2, c_3, c_6, c_7, c_8, c_9, c_{10}, c_{11}\},$$

the information symbols can be obtained.

Similarly, for the generator matrix $\widehat{\mathrm{G}}(2)$, the corresponding code is an $[n_2 = 11, k = 6, d_2 = 4]$ code with $(r_2 = 2, \delta_2 = 2)$ locality. Its codeword has the following symbols: $(c_1, c_2, \cdots, c_6, c_{12}, c_{13}, c_{14}, c_{15}, c_{16})$. As $d_2 = 4$, all the 6 information symbols can be recovered in the presence of $d_2 - 1 = 3$ erasures. Hence using

$$S_2 = \{c_1, c_4, c_5, c_{12}, c_{13}, c_{14}, c_{15}, c_{16}\},$$

the information symbols can be obtained.

There are two disjoint repair sets of symbols S_1 and S_2 for the recovery of all the information symbols . Thus the information-set availability is 2.

4

Codes with Availability for Multiple code symbols

Contents

4.1 Introduction

Codes with availability allow us to access any code symbol simultaneously t times using t disjoint recovery sets in distributed data storage. Codes with availability that are studied in the literature provide availability for individual code symbols in the codeword. We study a new class of codes with availability. These codes provide availability for subsets of code symbols called multi-symbol sets. This will allow us to provide availability t to a subset of code symbols.

The codes studied in this chapter are referred to as $[n, k, r, \rho, t, \delta]$ codes. In this class of codes, availability is provided for a subset of information symbols called *multi-symbol sets*. The size of each multi-symbol set is $\rho, \rho \leq r$. Each multi-symbol set can be recovered by t disjoint recovery sets. Each local code in this code is protected by a local code of minimum Hamming distance at least δ, $\delta \geq \rho+1$. This allows us to recover the message symbols of the local code in the presence of ρ erasures. In other words, the local codes can recover multiple erasures locally in a local code. One can recover the multi-symbol set from the message symbols of the local code. An upper bound on the minimum Hamming distance of these class of codes is derived. A construction of optimal codes is given. When compared with codes with availability for individual codes symbols, the optimal codes constructed in this chapter allow us to access more symbols locally with the same locality thus reducing the disk I/O. These codes can recover multiple erasures locally in a local code. We define a new class of membership matrices. The ones studied in [9, 77] are a special case of the membership matrices constructed here.

For the results obtained in this chapter, the notations used are introduced in section 2.2.

4.2 Codes with multi-symbol availability

Definition 4.1 (Multi-Symbol set)**:** Let $\mathcal{M}$ be a collection of subsets of $[k]$ such that the union of sets in $\mathcal{M}$ is $[k]$ and the size of each set in $\mathcal{M}$ is at most $\rho \leq r$. In other words,

$$\bigcup_{\kappa \in \mathcal{M}} \kappa = [k], \ |\kappa| \leq \rho \leq r.$$

The elements of $\mathcal{M}$ are called the multi-symbol sets.

Let $|\mathcal{M}| = Z$ and let the multi-symbol sets be enumerated as $M_1, \ldots, M_Z$.

Example 4.2: Consider the generator matrix G of an $[n = 18, k = 6, r = 3, \rho = 2, t = 2, \delta = 3]$ systematic code as shown on the top of the next page. A codeword c of this code is denoted as

$$
\mathrm{G} = \left[\begin{array}{cccccc|cc|cc|cc|cc|cc}
1 & 0 & 0 & 0 & 0 & 0 & a_{11} & a_{21} & 0 & 0 & c_{11} & c_{21} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & a_{12} & a_{22} & 0 & 0 & c_{12} & c_{22} & 0 & 0 & e_{11} & e_{21} & f_{11} & f_{21} \\
0 & 0 & 1 & 0 & 0 & 0 & a_{13} & a_{23} & 0 & 0 & 0 & 0 & d_{11} & d_{21} & 0 & 0 & f_{12} & f_{22} \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & b_{11} & b_{21} & c_{13} & c_{23} & d_{12} & d_{22} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & b_{12} & b_{22} & 0 & 0 & d_{13} & d_{23} & e_{12} & e_{22} & f_{13} & f_{23} \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & b_{13} & b_{23} & 0 & 0 & 0 & 0 & e_{13} & e_{23} & 0 & 0 \\
\underbrace{\qquad\qquad}_{I_6} & & & & & & \underbrace{\qquad}_{g_1} & & \underbrace{\qquad}_{g_2} & & \underbrace{\qquad}_{g_3} & & \underbrace{\qquad}_{g_4} & & \underbrace{\qquad}_{g_5} & & \underbrace{\qquad}_{g_6} &
\end{array}\right]
$$

G: The Generator matrix of $[n = 18, k = 6, r = 3, \rho = 2, t = 2, \delta = 3]$ code.

$c = (c_1, c_2, \ldots, c_{18})$.

In this code, the first 6 symbols $c_1, c_2, c_3, c_4, c_5, c_6$ are the information symbols. For this code, we have $Z = 4$ distinct multi-symbol sets, which are given by

$$M_1 = \{1, 2\}, M_2 = \{2, 3\}, M_3 = \{4, 5\}, M_4 = \{5, 6\}.$$

Note that $|\kappa| \leq 2, \forall \kappa \in \mathcal{M}$.

The code has $J = 6$ distinct local codes that provide availability 2 for all the multi-symbol sets. The set of parity columns corresponding to the six local codes are denoted by g_1, g_2, g_3, g_4, g_5 and g_6. The generator matrix for each of the six local codes is given by $[I_j|g_j], 1 \leq j \leq 6$ where I_j is a subset of columns of the identity matrix I_6 and $|I_j| \leq 3$ for all $j = 1, 2, 3, 4, 5, 6$. The six local codes are given by the following code symbols:

$$(c_1, c_2, c_3, c_7, c_8), (c_4, c_5, c_6, c_9, c_{10}), (c_1, c_2, c_4, c_{11}, c_{12}), (c_3, c_4, c_5, c_{13}, c_{14}), (c_2, c_5, c_6, c_{15}, c_{16}),$$
$$(c_2, c_3, c_5, c_{17}, c_{18})$$

The coefficients can be chosen such that the local codes are MDS codes.

It can be easily verified that every multi-symbol set has availability $t = 2$. For example, for the multi-symbol set $\kappa = \{1, 2\}$, we have

$$Q_\kappa(1) = \{1, 2, 3, 7, 8\}, Q_\kappa(2) = \{1, 2, 4, 11, 12\}$$

and

$$Q_\kappa(1) \cap Q_\kappa(2) = \kappa.$$

Since any local code is an MDS code, using the code symbols $\{c_3, c_7, c_8\}$, the three information symbols $\{c_1, c_2, c_3\}$ are recovered. From this obtain $\{c_1, c_2\}$. Similarly using $\{c_4, c_{11}, c_{12}\}$, $\{c_1, c_2\}$ are obtained. Hence $\{c_1, c_2\}$ can be recovered simultaneously using the two disjoint recovery sets.

Definition 4.3: An $[n, k, d]$ code $\mathcal{C}$ has multi-symbol information availability t, if for any $\kappa \in \mathcal{M}$, there exists t subsets $Q_\kappa(1), Q_\kappa(2), \cdots, Q_\kappa(t) \subset [n]$ containing κ such that

(1) $|\kappa| \leq \rho$ and $|Q_\kappa(l)| \leq r + \delta - 1$, where $\delta \geq \rho + 1$ for all $l \in [t]$.

(2) $\mathcal{C}|_{Q_\kappa(l)}$ is a local code of minimum Hamming distance at least δ.

(3) $Q_\kappa(l) \cap Q_\kappa(l') = \kappa, \forall l.l' \in [t], l \neq l'$.

These codes are referred to as $[n, k, r, \rho, t, \delta]$ codes. Since $\delta \geq \rho + 1$, any local code can recover the local message symbols even in the presence of ρ erasures. To recover the multi-symbol set of size ρ, recover the local message vector from the remaining symbols (the symbols which do not contain the ρ symbols of the multi-symbol set) of the local codeword. From the local message vector, consider only the symbols in the multi-symbol set.

If all the code symbols have locality at most r, then the code is said to have all-symbol locality. Let J be the total number of distinct local codes in this code such that availability property is satisfied for all of the multi-symbol sets.

The *membership matrices* for codes with availability are studied in [9, 77]. A *membership matrix* $\mathbf{R}$ is a binary matrix of size $k \times J$. Its $(i, j)^{th}$ entry has a 1, if the $i^{th}, i \in [k]$ information symbol is in the $j^{th}, j \in [J]$ local code. All other entries in this matrix are zero.

We define a membership matrix $\mathbf{R}_\rho$ for the Z multi-symbol sets. The membership matrix is of size is $Z \times J$, where $Z = |\mathcal{M}|$. Let the Z distinct multi-symbol sets be enumerated as $M_1, \ldots, M_Z$. $\mathbf{R}_\rho$ has a 1 in the $(z, j)^{th}$ entry if the multi-symbol set $M_z, z \in [Z]$ is in the j^{th} local code, $j \in [J]$. All other entries in this matrix are zero. Every row of $\mathbf{R}_\rho$ has weight at least $t = 2$.

For the code described in example 4.2, the membership matrix $\mathbf{R}_\rho$ of size 4×6 and the membership

matrix $\mathbf{R}$ of size 6×6 are respectively given below as

$$\mathbf{R}_\rho = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}, \mathbf{R} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Every row of $\mathbf{R}$ has weight at least $t = 2$ and every column has weight at most $r = 3$. In general the weight of any column is at most r because there will be at most r information symbols in any local code. The weight of any row is at least t. Observe that in this example, (i) the dimension of any local code is the number of information symbols in it and (ii) any two local codes have disjoint parity symbols.

The following two lemmas are used to derive an upper-bound on the minimum Hamming distance of the $[n, k, r, \rho, t, \delta]$ code.

Lemma 4.4: Let a matrix $\mathbf{R}$ of size $k \times J$ with entries from $\{0, 1\}$ be constructed such that the row weight of $\mathbf{R}$ is at least t and the column weight of $\mathbf{R}$ is at most r. Then [9],

$$J \geq \left\lceil \frac{kt}{r} \right\rceil.$$

We now derive an upper bound on the minimum Hamming distance of the $[n, k, r, \rho, t, \delta]$ code.

Theorem 4.5: Let $\mathcal{C}$ be an $[n, k, r, \rho, t, \delta]$ code with information locality. Suppose

(1) $\dim(\mathcal{C}|_{Q_\kappa(l)}) = |Q_\kappa(l) \cap [k]|$ for all $\kappa \in \mathcal{M}$ and for all $l \in [t]$.

(2) Either $Q_\kappa(l) = Q_{\kappa'}(l')$ or $Q_\kappa(l) \cap Q_{\kappa'}(l') \subseteq [k],\ \forall \kappa, \kappa' \in \mathcal{M},\ \forall l, l' \in [t]$.

Then, the minimum Hamming distance of the code is upper bounded by

$$d \leq n - k + 1 - \left\lceil \frac{kt}{r} \right\rceil (\delta - 1) + t(\delta - 1). \tag{4.1}$$

Proof : Let G be the generator matrix of the code $\mathcal{C}$. Let $\mathcal{D}$ be the collection of sets of the form $Q_\kappa(l)$ for $l = 1, \ldots, t$ such that for any $i \in [k]$, there exists some $\kappa \in \mathcal{M}$ such that $i \in \kappa$. Though it

may happen that some of these sets are identical, that is, for some $i, i' \in [k], i \in \kappa, i' \in \kappa$, we only include the distinct sets in the collection $\mathcal{D}$.

Let

$$T = \bigcup_{S \in \mathcal{D}} S$$

and

$$J = |\mathcal{D}|.$$

Let G_T be the sub-matrix of G consisting of the columns of G indexed by the set T. The first k columns of G_T is identical to that of G.

Let $m \in [k]$ be an information symbol that occurs in the least number of distinct local groups. Define

$$\mathcal{D}_m = \{S \in \mathcal{D} \big| m \in S\} \text{ and } t' = |\mathcal{D}_m|.$$

Note that $t' \geq t$. Let

$$E_m = \{m\} \bigcup \{ \biguplus_{S \in \mathcal{D}_m} S \setminus [k].\} \tag{4.2}$$

Note that $E_m \subseteq T$. Furthermore,

$$|T| = k + |\biguplus_{S \in \mathcal{D}} S \setminus [k]|$$

$$|E_m| = 1 + |\biguplus_{S \in \mathcal{D}_m} S \setminus [k]|$$

$$\text{and } |T \setminus E_m| = (k-1) + |\biguplus_{S \in \mathcal{D} \setminus \mathcal{D}_m} S \setminus [k]|. \tag{4.3}$$

Let P_m be the set that includes the information symbol m and all the parity columns of which it is a part of. Note that,

$$P_m \subseteq E_m.$$

Since $|\mathcal{D}| = J$, from Lemma 4.4,

$$J \cdot r \geq k \cdot t'$$

$$\implies J \geq \left\lceil \frac{kt'}{r} \right\rceil. \tag{4.4}$$

The rank of G_T is k. A rank $(k-1)$ set T_{end} is obtained as

$$T_{end} = T \setminus P_m.$$

We have

$$|T_{end}| = |T| - |P_m|.$$

So

$$\begin{aligned} |T_{end}| &\geq |T| - |E_m| \\ &= k - 1 + |\biguplus_{S \in \mathcal{D} \setminus \mathcal{D}_m} S \setminus [k]| \qquad (4.5) \\ &\overset{(a)}{\geq} k - 1 + (\left\lceil \frac{kt'}{r} \right\rceil - t')(\delta - 1), \qquad (4.6) \\ &\geq k - 1 + (\left\lceil \frac{kt}{r} \right\rceil - t)(\delta - 1), \text{ as } t' \geq t. \end{aligned}$$

We have the inequality (a) as

$$|\mathcal{D} \setminus \mathcal{D}_m| = J - t' \geq \left\lceil \frac{kt'}{r} \right\rceil - t'$$

and $|S \setminus [k]| \geq \delta - 1$ (which is due to assumption 1 and assumption 2 in the statement of the theorem). Therefore using lemma 2.5,

$$d \leq n - k + 1 - \left\lceil \frac{kt}{r} \right\rceil (\delta - 1) + t(\delta - 1).$$

The assumption $\dim(\mathcal{C}|_{Q_\kappa(l)}) = |Q_\kappa(l) \cap [k]|)$ in theorem 2.5 is equivalent to the assumption considered in deriving equation (2.9), which is derived in [9]. This means the number of information symbols from $[k]$ in a local code $\mathcal{C}|_{Q_\kappa(l)}$ is exactly $\dim(\mathcal{C}|_{Q_\kappa(l)})$. The second assumption means the parity symbols for all the local codes are disjoint.

4.3 Optimal Code Constructions

4.3.1 Optimal codes with information multi-symbol availability

In this section we provide optimal code constructions that meet the upper bound derived in section 4.2 with equality. The construction provides availability for the multi-symbol sets defined over the information symbols. This construction is given for when $r \mid k$ and $\rho \mid r$. Note that, there will be Z

distinct multi-symbol sets $M_1, \ldots, M_Z$. Here

$$M_z \cap M_{z'} = \phi, \forall z, z' \in [Z]$$

and

$$Z = \frac{k}{\rho}. \tag{4.7}$$

Here

$$|M_z| = \rho, \forall z \in [Z].$$

The construction of the codes is based on *membership matrices* $\mathbf{R}_\rho, \mathbf{R}$. The following assumption is made regarding the *membership matrix* $\mathbf{R}_\rho$.

Assumption 4.6: There exists a $Z \times \frac{t \times k}{r}$ matrix $\mathbf{R}_\rho$ such that

(i) The number of ones in any of the columns is exactly

$$r_\rho = \frac{r}{\rho}. \tag{4.8}$$

(ii) The support of all the columns of $\mathbf{R}_\rho$ gives t partitions of $[Z]$ multi-symbol sets.

(iii) Any two rows of $\mathbf{R}_\rho$ intersect in at most one position.

The t partitions of $[Z]$ are denoted as $\mathbf{R}_\rho^1, \ldots, \mathbf{R}_\rho^t$. So,

$$\mathbf{R}_\rho = [\mathbf{R}_\rho^1 | \ldots | \mathbf{R}_\rho^t].$$

Any $\mathbf{R}_\rho^l$ has $\frac{k}{r}$ columns i.e.,

$$\mathbf{R}_\rho^l = \{\mathbf{R}_\rho^l(1), \ldots, \mathbf{R}_\rho^l(\frac{k}{r})\} \ \forall l \in [t].$$

Each column has exactly r_ρ ones and the rest are zeros.

This matrix should be designed in such a a way that

$$\sum_{j=1}^{\frac{k}{r}} \mathbf{R}_\rho^l(j) = [1 \ 1 \ldots 1]^T$$

is a $Z \times 1$ matrix, where A^T is the transpose of a matrix A. Note that $\mathbf{R}_\rho$ can be constructed for all the parameters for which they are constructed in [9, 77].

Using $\mathbf{R}_\rho$, we construct the membership matrix $\mathbf{R}$ of size $k \times (\frac{t \times k}{r})$. This membership matrix is for individual information symbols and we have

$$\mathbf{R} = [\mathbf{R}^1 | \dots | \mathbf{R}^t]. \tag{4.9}$$

Any $\mathbf{R}^l$ has $\frac{k}{r}$ columns

$$\mathbf{R}^l(1), \cdots, \mathbf{R}^l(\frac{k}{r}).$$

Each column $\mathbf{R}^l(j), j \in [\frac{k}{r}]$ has exactly r ones and the rest are zeros. If the $z \in [Z]$-th row of $\mathbf{R}^l_\rho(j)$ has a 1, then the rows indexed by $M_z \subseteq [k]$ will have 1's in $\mathbf{R}^l(j)$ and its other entries are zeros.

The membership matrix $\mathbf{R}$ constructed here is such that for the rows indexed by $M_z, M_{z'}, z, z' \in [Z]$ in $\mathbf{R}$, their support intersect in at most one position. We now give a construction of optimal codes as follows.

Construction 4.7: Step 1: Consider an $[n', k, d]$, $d \geq t(\delta - 1) + 1$ systematic MDS code over $\mathbb{F}_q$ with the following generator matrix

$$\mathrm{G} = [I_k | \Lambda], \tag{4.10}$$

where I_k is a $k \times k$ identity matrix and Λ is a $k \times (d-1)$ matrix.

Step 2: Partition Λ into sub-matrices as

$$\Lambda = [\Lambda^1 | \dots | \Lambda^t | \Lambda']. \tag{4.11}$$

where Λ^l is a $k \times (\delta - 1)$ matrix, $1 \leq l \leq t$.

Step 3: Construct a $k \times (\delta - 1)$ sub-matrix Λ^l_j for $1 \leq l \leq t$, $\quad 1 \leq j \leq \frac{k}{r}$ as follows: It will have non-zeros entries in the rows that are indexed by the rows that have ones in $\mathbf{R}^l(j)$ and they are the entries from rows of Λ^l that are indexed by rows that have ones in $\mathbf{R}^l(j)$. All others rows in Λ^l_j are zero. Define a $k \times \frac{k}{r}(\delta - 1)$ matrix Ω^l as

$$\Omega^l = [\Lambda^l_1 | \dots | \Lambda^l_{\frac{k}{r}}].$$

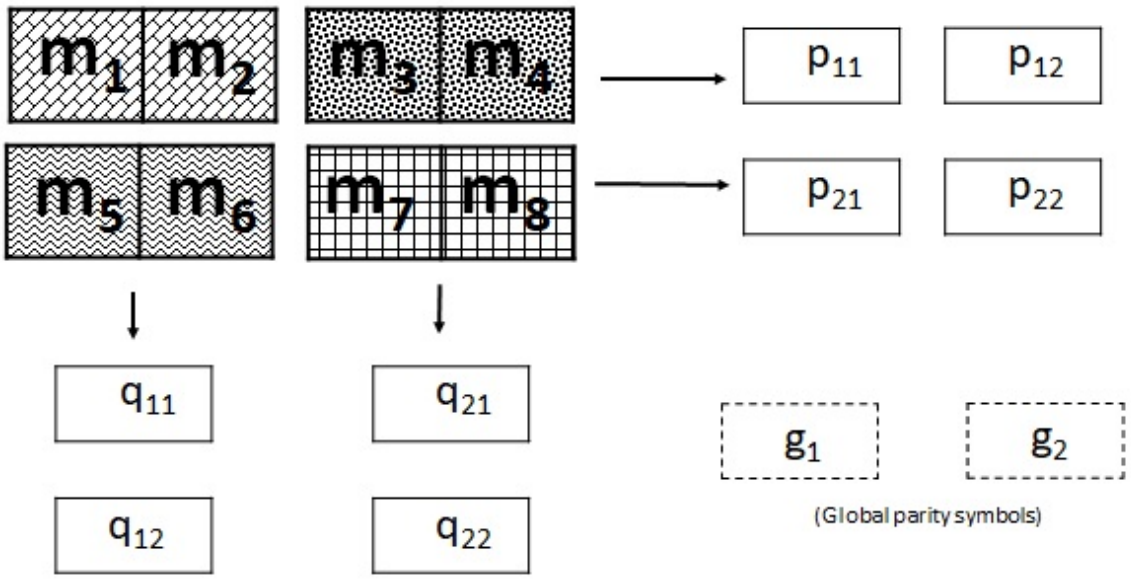

Figure 4.1: A code with multi-symbol availability

Step 4: Form the matrix $\widehat{\mathrm{G}}$ as follows:

$$\widehat{\mathrm{G}} = \left[\begin{array}{c|c|c|c|c} I_k & \Omega^1 & \cdots & \Omega^t & \Lambda' \end{array} \right].$$

The code generated by the generator matrix $\widehat{\mathrm{G}}$ will have minimum Hamming distance given by (4.1). It can be easily verified using the techniques employed in the proof of theorem 3, given in [9].

We explain the construction using the following example.

Example 4.8: This example is for $[n = 18, k = 8, r = 4, \rho = 2, t = 2, \delta = 3]$ code. It is illustrated in Fig. 4.1.

The 8 information symbols are $m_1, m_2, \cdots, m_8$. The multi-symbol sets are indicated with 4 different patterns. Each row and column is an $[6, 4, 3]$ MDS code. The symbols $p_{11}, p_{12}, p_{21}, p_{22}$ are the parity symbols for the local codes which are represented horizontally in Fig. 4.1. The symbols $q_{11}, q_{12}, q_{21}, q_{22}$ are the parity symbols for the local codes which are represented vertically in Fig. 4.1. The global parity symbols are g_1, g_2. The disjoint recovery sets for the multi-symbols $\{m_1, m_2\}$ are $\{m_3, m_4, p_{11}, p_{12}\}$ and $\{m_5, m_6, q_{11}, q_{12}\}$. It can be easily verified that all the 4 multi-symbol sets have availability $t = 2$.

Using the generator matrix G of an $[n = 14, k = 8, d = 7]$ systematic MDS code over $\mathbb{F}_{17}$ given

below, we construct the generator matrix $\widehat{\mathrm{G}}$ of the code with *multi-symbol set* availability.

$$\mathrm{G} = \left[\begin{array}{c|cc|cc|cc} & 16 & 9 & 15 & 16 & 10 & 7 \\ & 8 & 12 & 8 & 6 & 4 & 5 \\ & 6 & 5 & 7 & 14 & 14 & 13 \\ & 5 & 12 & 15 & 12 & 15 & 13 \\ I_8 & 15 & 6 & 8 & 13 & 15 & 12 \\ & 5 & 4 & 16 & 13 & 14 & 14 \\ & 6 & 2 & 16 & 5 & 4 & 6 \\ & \underbrace{8 \quad 2}_{\Lambda^1} & & \underbrace{1 \quad 7}_{\Lambda^2} & & \underbrace{10 \quad 16}_{\Lambda'} & \end{array}\right].$$

There are $\frac{k}{\rho} = 4$ multi-symbol sets: $\{1,2\}$, $\{3,4\}$, $\{5,6\}$, $\{7,8\}$. Since $\rho \mid r$ each local code has exactly 2 multi-symbol sets. The *membership matrices* $\mathbf{R}_\rho$ and $\mathbf{R}$ are given below as

$$\mathbf{R}_\rho = \left[\begin{array}{c|c} \mathbf{R}_\rho^1 & \mathbf{R}_\rho^2 \end{array}\right], \mathbf{R}_\rho^1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{R}_\rho^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\mathbf{R} = \left[\begin{array}{c|c} \mathbf{R}^1 & \mathbf{R}^2 \end{array}\right], \mathbf{R}^1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{R}^2 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

Now using the columns of $\mathbf{R}$, we obtain

$$\Omega^1 = \left[\begin{array}{cc|cc} 16 & 9 & 0 & 0 \\ 8 & 12 & 0 & 0 \\ 6 & 5 & 0 & 0 \\ 5 & 12 & 0 & 0 \\ 0 & 0 & 15 & 6 \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 6 & 2 \\ 0 & 0 & 8 & 2 \end{array}\right] \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{cc} \multicolumn{2}{c}{\underbrace{}_{\Lambda_1^1}} & \multicolumn{2}{c}{\underbrace{}_{\Lambda_2^1}} \end{array}, \Omega^2 = \left[\begin{array}{cc|cc} 15 & 16 & 0 & 0 \\ 8 & 6 & 0 & 0 \\ 0 & 0 & 7 & 14 \\ 0 & 0 & 15 & 12 \\ 8 & 13 & 0 & 0 \\ 16 & 13 & 0 & 0 \\ 0 & 0 & 16 & 5 \\ 0 & 0 & 1 & 7 \end{array}\right] \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{cc} \multicolumn{2}{c}{\underbrace{}_{\Lambda_1^2}} & \multicolumn{2}{c}{\underbrace{}_{\Lambda_2^2}} \end{array}.$$

The generator matrix $\widehat{\mathrm{G}}$ of an code with minimum Hamming distance 7 is given by

$$\widehat{\mathrm{G}} = \left[\begin{array}{c|c|c|c} I_8 & \underbrace{\begin{array}{c|c} \Lambda_1^1 & \Lambda_2^1 \end{array}}_{\Omega^1} & \underbrace{\begin{array}{c|c} \Lambda_1^2 & \Lambda_2^2 \end{array}}_{\Omega^2} & \Lambda' \end{array}\right].$$

When $\rho = 1$, this code will have availability for individual symbols of the codeword and each local code is protected by a local code of minimum Hamming distance at least $\delta, \delta \geq 2$.

4.3.2 Reduction in Disk I/O

With the optimal codes constructed using construction 4.8, with r disk reads we can access ρ symbols. With tr disk reads we can access these ρ symbols simultaneously t times. With optimal codes with individual code symbol availability [9], using r disk reads, we can access 1 code symbol. To access the same ρ symbols, we need ρr disk reads. To access ρ symbols simultaneously t times, we need ρtr disk reads. Hence to provide availability to ρ symbols, our codes need fewer disk reads than the codes with availability for individual code symbols, thus reducing disk I/O.

4.3.3 Optimal codes with all-symbol locality using Gabidulin codes

Optimal codes with availability for multi-symbol sets and locality for all the symbols are constructed using the following construction.

Construction 4.9: The construction is similar to construction 3.9. The membership matrix given by equation (4.9) should be used in this construction. Note that $r_1 = r_2 = \cdots = r_t = r$ here.

For the optimal codes with information availability , the storage overhead is by a factor of $\lceil \frac{kt}{r} \rceil (\delta - 1)$ for codes with information availability. For the optimal codes with all symbol locality with information availability, the storage overhead is $\lceil \frac{kt}{r} \rceil (\delta - 1) + \frac{N-k}{r}(\delta - 1)$.

4.4 Special cases

4.4.1 The case when $\rho = 1$.

When $\rho = 1$, this code will have availability for individual symbols of the codeword and each local code is protected by a local code of minimum Hamming distance at least $\delta, \delta \geq 2$. The minimum Hamming distance is upper bounded by the same bound, i.e., equation (4.1).

From (4.7),(4.8) we have $Z = k, r_\rho = r$. Hence, we have

$$\mathbf{R}_\rho = \mathbf{R}.$$

4.4.2 The case when $\rho = 1, \delta = 2$.

When $\rho = 1$ and $\delta = 2$, the code will have individual code symbol availability and any local code has minimum hamming distance at least 2. In this case, the bound (4.1) reduces to,

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

This bound is same as derived in equation (2.9). However, when deriving equation (2.9), in [9], the assumption was that there will be exactly one parity symbol in each local code. This means $\delta = 2$ exactly.

In this result, we show that the minimum Hamming distance is upper bounded by the same bound even when there is at least 1 parity symbol in the local code (minimum Hamming distance at least 2) i.e., $(\delta \geq 2)$. As $\rho = 1$, we have

$$\mathbf{R}_\rho = \mathbf{R}.$$

The codes with all-symbol locality along with multi-symbol set availability can be constructed using Gabidulin codes and the construction is similar to the construction given in [9].

5

Dynamic Recovery sets for Codes with Availability and their Applications to Blockchain

Contents

5.1 Introduction

In distributed data storage, a code symbol can be accessed simultaneously multiple times using disjoint recovery sets using codes with availability. These codes are studied in literature for the case where the disjoint recovery sets are fixed. This restriction imposes limitations on the construction of optimal codes. Hence optimal codes with availability do not exist for many parameters. In this chapter, we define codes with availability, where the disjoint recovery sets for a code symbol are not fixed but dynamic in nature. One can choose at most any r other code symbols to recover the code symbol, thus enabling us to choose the t recovery sets arbitrarily. We derive an upper bound on the minimum Hamming distance of this code. We show that locally recoverable codes with (r, δ) locality (which are studied in literature) are optimal codes.

We show the advantages of these optimal codes in distributed storage blockchain system with Local secret sharing scheme. This system stores two secrets: a global secret and a local secret. These optimal codes increases the ability of the system to recover the transactions in the presence of more erasures.

For the results obtained in this chapter, the notations used are introduced in section 2.2.

5.2 Dynamic recovery sets for codes with availability

Codes with availability are studied in the literature for the case where the disjoint recovery sets are fixed. The disjoint recovery sets of the code are described using special matrices called as *membership matrices* [9, 77]. The following are two issues with these class of codes.

(1) The recovery sets are fixed. Loss of any code symbol in a recovery set of a code symbol will render the recovery set unusable for providing availability. This is because there will be two erasures in the local code and hence cannot be recovered. The remaining code symbols will remain unused in local recovery process.

(2) Membership matrices are difficult to construct. They do not exist for a lot of parameters. So, optimal codes do not exist for many parameters.

5.2.1 Availability with dynamic recovery sets

The features of the codes that are studied in this chapter are given below.

i) Any r code symbols in a local code can be used for recovery and availability.

ii) These codes exist for a lot of parameters.

iii) These codes shall provide all symbol locality and availability.

We show that (r, δ) codes [8] have these features. The optimal code construction of $[n, k, d]$ LRC with (r, δ) locality [16] is given below :

Example 5.1: We shall construct an optimal $[n = 10, k = 4, r = 2, \delta = 4]$ code over $\mathbb{F}_{11}$ with all symbol locality using the construction given in [16]. We then show that this code has the desirable properties.

The good polynomial is $g(x) = x^5$ and the message vector is $\mathbf{m} = (m_{00}, m_{01}, m_{10}, m_{11})$. The encoding polynomial as

$$f_e(x) = m_{00} + m_{10}x + m_{01}x^5 + m_{11}x^6.$$

The evaluation points are given by $\mathcal{A}_1 = \{3, 9, 5, 4, 1\}$, $\mathcal{A}_2 = \{2, 6, 7, 8, 10\}$ and $\mathcal{A} = \mathcal{A}_1 \biguplus \mathcal{A}_2$. The codeword c for $\mathbf{m} = (1, 1, 1, 1)$ is obtained by evaluating $f_e(x)$ at all the points in $\mathcal{A}$ and

$$c = (f_e(3), f_e(9), f_e(5), f_e(4), f_e(1), f_e(2), f_e(6), f_e(7), f_e(8), f_e(10)),$$
$$c = (8, 9, 1, 10, 4, 0, 0, 0, 0, 0).$$

This code provides all symbol locality and availability with dynamic recovery sets. It is explained below. Let $(c_1, \cdots, c_{10})$ be the codeword.

(1) Availability $t = 2$ for c_1 is provided as follows. The degree of $f_e(x)$ when restricted to the set $\mathcal{A}_1$ is $r - 1$. It can be reconstructed by any r evaluations of $\mathcal{A}_1$. So, the encoding polynomial can be obtained using two different sets: $\{c_2, c_3\}$ and $\{c_4, c_5\}$.

(2) Obtain the decoding polynomial $d(x) = 2x + 2$ from $\{c_2, c_3\}$. Recover c_1 by evaluating $d(x)$ as the first evaluation point 3 in $\mathcal{A}_1$, i.e., $d(3) = 8$, which is the desired value.

(3) Similarly obtain $d(x) = 2x + 2$ from $\{c_4, c_5\}$ and determine $d(3) = 8$.

(4) The choice of the code symbols in a recovery set to recover c_1 is arbitrary. The recovery sets can also be chosen as $\{c_2, c_5\}$ and $\{c_3, c_4\}$. Hence the recovery sets are not fixed.

(5) This code provides availability $t = 2$ for all the symbols in a local code. Hence this code can be viewed as a code with all symbol locality and all symbol availability.

Definition 5.2: The i^{th} code symbol $c_i,\ i \in \mathcal{I}$ of an $[n, k, d]$ linear code with the generator matrix G is said to have (r, t) availability, if there exists sets $Q_1(i), Q_2(i), \ldots, Q_t(i) \subseteq [n]$, such that

(1) c_i is a function of the code symbols indexed by $Q_1(i), Q_2(i), \ldots, Q_t(i)$ and $i \in Q_l(i)$ such that $\text{Rank}(\text{G}|_{Q_l(i)}) \leq r$, and $|Q_l(i)| \leq r + 1,\ 1 \leq l \leq t$.

(2) $Q_l(i) \cap Q_{l'}(i) = \{i\}$, for any $l, l' \in [t]$.

(3) $Q(i) = \bigcup_{l=1}^{t} Q_l(i)$ and $Q_l(i)$ can be chosen arbitrarily from $Q(i)$.

Since $|Q_l(i)| \leq r + 1$ and $Q_l(i) \cap Q_{l'}(i) = \{i\}$ for any $l, l' \in [t]$, we have

$$|Q(i)| \leq 1 + tr. \tag{5.1}$$

Theorem 5.3: Let $\mathcal{C}$ be a code as defined in definition 5.2. Suppose

(1) $|Q(i)| - \text{Rank}(\text{G}|_{Q(i)}) \geq (t-1)r + 1$.

The minimum Hamming distance of this code is bounded as

$$d \leq n - k + 1 - \left(\left\lceil \frac{k}{r} \right\rceil - 1 \right)((t-1)r + 1) \tag{5.2}$$

Proof : Let $\mathcal{C}$ be an $[n, k, d]$ code with availability t. Let G be the generator matrix of the code $\mathcal{C}$. A set of columns of rank $k-1$ is constructed using Algorithm 2. Let $\mathcal{I}$ be the set of columns of G which are of rank k and $|\mathcal{I}| = k$. For any $i \in \mathcal{I}$, the collection of all its disjoint recovery sets is given by

$$Q(i) = \bigcup_{l=1}^{t} Q_l(i).$$

At the j^{th} iteration, an $i \in \mathcal{I}$ can be picked, as long as

$$\text{rank}(\text{G}|_{V_{j-1}}) \leq k - 1.$$

The algorithm runs for $\mathcal{J}$ iterations. We have

$$\text{Rank}\left(\text{G}|_{V_{\mathcal{J}-1} \cup Q_{\text{end}}}\right) = k.$$

Algorithm 2 Used in the Proof of Theorem 5.3

Let $V_0 = \{\ \}$, $j = 0$
while 1 **do**
  Pick $i \in \mathcal{I}$ such that $X_i \notin \text{Col}(\text{G}|_{V_j})$
  if Rank $\left(\text{G}|_{V_j \cup Q(i)}\right) \leq k-1$ **then**
    $j = j+1$
    $V_j = V_{j-1} \cup Q(i)$
  else if Rank $\left(\text{G}|_{V_j \cup Q(i)}\right) = k$ **then**
    Pick any subset Q'' of $Q(i)$ of maximum size such that rank $\left(\text{G}|_{V_j \cup Q''}\right) = k-1$
    $Q_{\text{end}} = Q(i)$
    $j = j+1$
    $V_j = V_{j-1} \cup Q''$
    Exit
  end if
end while

From equation (5.1) and assumption (1) in the statement of theorem 5.3, we have Rank $\left(\text{G}|_{Q(i)}\right) \leq r$. We have

$$\mathcal{J} = \left\lceil \frac{k}{r} \right\rceil. \tag{5.3}$$

For any $j \in [\mathcal{J}]$, let

$$u_j = |V_j| - |V_{j-1}|$$
$$v_j = \text{Rank}(\text{G}|_{V_j}) - \text{Rank}(\text{G}|_{V_{j-1}}).$$

For $j \in [\mathcal{J}-1]$, we have

$$V_j = V_{j-1} \cup Q(i)$$
$$|V_j| = |V_{j-1} \cup Q(i)|.$$

Similarly, for $j \in [\mathcal{J}-1]$, we have

$$\begin{aligned} u_j &= |Q(i)| - |V_{j-1} \cap Q(i)| \\ &\geq \text{Rank}(\text{G}|_{Q(i)}) + (t-1)r + 1 - |V_{j-1} \cap Q(i)| \\ &= \text{Rank}(\text{G}|_{Q(i)}) - |V_{j-1} \cap Q(i)| + (t-1)r + 1 \\ &\overset{(a)}{=} v_j + (t-1)r + 1. \end{aligned}$$

Here (a) follows because

$$|V_{j-1} \cap Q(i)| \leq \text{Rank}(\text{G}|_{Q(i)}) - v_j.$$

This is because in each step, we have to add at least $(t-1)r+1$ columns (due to assumption 1 in theorem 5.3) which are not in V_{j-1} and at least v_j other columns which are also not in V_{j-1} which add to the dimension. The remaining columns (if any) do not increase the dimension and they may lie in $V_{j-1} \cap Q(i)$. Hence $|V_{j-1} \cap Q(i)| \leq \text{Rank}(\text{G}|_{Q(i)}) - v_j$. For $j = \mathcal{J}$,

$$\begin{aligned} V_{\mathcal{J}} &= V_{\mathcal{J}-1} \cup Q'' \\ u_{\mathcal{J}} &\geq |Q''| - |V_{\mathcal{J}-1} \cap Q''| \\ &\geq v_{\mathcal{J}}. \\ \text{So, } |V_{\mathcal{J}}| &= \sum_{j=1}^{\mathcal{J}} u_j, \qquad (5.4) \\ &\geq \sum_{j=1}^{\mathcal{J}} v_j + (\mathcal{J}-1)((t-1)r+1) \\ &\geq k - 1 + \Big(\left\lceil \frac{k}{r} \right\rceil - 1 \Big)((t-1)r+1). \qquad (5.5) \end{aligned}$$

Applying lemma 2.5, we get the result.

Lemma 5.4: For an optimal code (with $r \mid k$) that meets the bound (5.2) with equality, the following holds:

(1) $\mathcal{J} = \frac{k}{r}$.

(2) Any local code is an $[1+tr, r, (t-1)r+2]$ MDS code.

Proof : For the code to be optimal, equation (5.5) should be satisfied with equality.

(i) For equation (5.5) to be satisfied, we should have $\mathcal{J} = \frac{k}{r}$. This means that the dimension of any local code $\mathcal{C}|_{Q_i(j)} = r$ for all $j \in \mathcal{J}$.

(ii) From (1) in the statement of theorem 5.3

$$|Q(i)| - \text{Rank}(\text{G}|_{Q(i)}) \geq (t-1)r + 1.$$

Since $\mathcal{J} = \frac{k}{r}$, we have dimension of any local code to be r. Hence, the corresponding columns of the generator matrix of any local code will be such that,

$$\text{rank}(\text{G}|_{Q(i)}) = r.$$

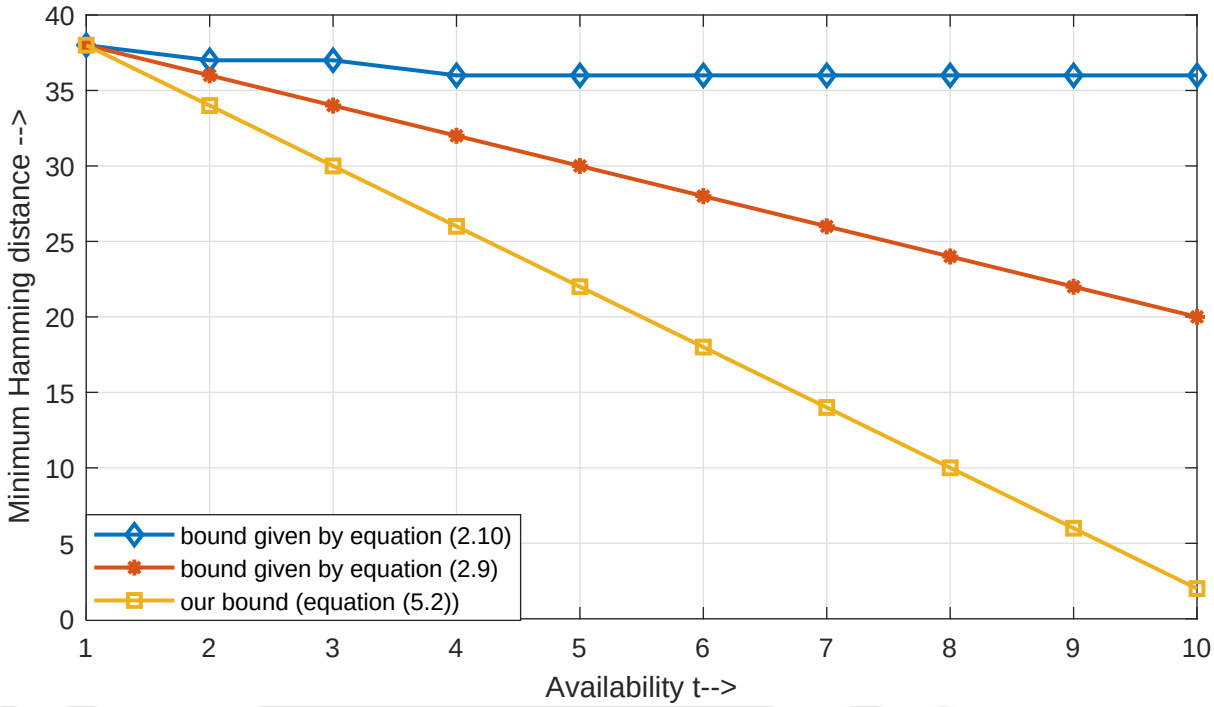

Figure 5.1: Comparison of the bounds for $n = 45, k = 6, r = 2$ and $t = \{1, 2, \ldots, 10\}$.

From these two equations and equation (5.1), we have

$$|Q(i)| = r + (t-1)r + 1 = 1 + tr.$$

As any code symbol can be recovered from any r code symbols, any r columns of $\mathrm{G}|_{Q(i)}$ are of rank r. This means, the local code is an $[1 + tr, r, (t-1)r + 2]$ MDS code.

Construction 5.5: Any optimal LRC with all symbol locality which is constructed by setting $\delta = (t-1)r + 2$ is an optimal code that provides all symbol availability and dynamic recovery sets. For example, the constructions given in [16,51] provide all symbol availability with dynamic recovery sets.

Please note that as t increases the gap between equation (2.9), equation (2.10)and equation (5.2) increases. This is shown in Fig. 5.1. Note that, equation (5.2) gives a smaller value of minimum Hamming distance than equation (2.9), equation (2.10) (i.e., with fixed recovery sets).

5.3 Application of (r, δ) codes to distributed storage blockchain

Distributed storage blockchain (DSB) is introduced in [10,11]. Distributed storage blockchain with Local secret sharing scheme (DSB with LSS) is introduced in [12]. In DSB with LSS, the important

secret is stored as a global secret and the lesser important secret is stored as a local secret. DSB with LSS uses an $[r+1, r]$ local codes for local secret sharing. We use (r, δ) code for local secret sharing. This system is referred to as DSB with (r, δ) LSS.

The efficiency of a blockchain system is measured by the following three parameters [12]. These are introduced in section 2.7. For the convenience of the reader these are described again.

- The storage cost ($\mathbf{S}_c$) : It is the cost of storing the data. It is proportional to the amount of data stored in a node.
- Communication cost ($\mathbf{C}_c$): It is the cost of downloading the data from other nodes. It is proportional the amount of data downloaded.
- Robustness to peer failures ($\mathbf{R}_p$): It is the maximum number of nodes that are failed, after which the transaction can be recovered with the remaining nodes.

We compare these for different systems. These are tabulated in table 5.1. We request the reader to refer to section 2.7, to familiarize themselves on how these parameters are determined. We show that with the same storage cost ($\mathbf{S}_c$) and communication cost ($\mathbf{C}_c$) as DSB with LSS, the local secret can be recovered with larger $\mathbf{R}_b$ than for DSB. The recovery of transaction in the presence of failed nodes is increased to $\delta \cdot \frac{n}{r+\delta-1} - 1$ nodes when we use DSB with (r, δ) LSS. Distributed blockchain systems are reviewed below.

Traditional blockchain system [69]: In a traditional blockchain system, let B_τ be the data block at time τ. Let $f_1(.), f_2(.)$ be two hash functions. Let $X_\tau = f_1(Z_\tau)$ be the hash value stored with the $(\tau+1)^{th}$ transaction. Let $Z_\tau = (Z_{\tau-1}, f_2(B_\tau))$ be the concatenation of the previous hash value and a hash value of the current data block. It is assumed that $B_\tau \in \mathbb{F}_\Omega$ and $Z_\tau \in \mathbb{F}_q$. Any node stores

	$\mathbf{S}_c$	$\mathbf{C}_c$	$\mathbf{R}_p$
Traditional blockchain [12]	$\log_2 \Omega + \log_2 q$	$\log_2 \Omega + \log_2 q$	$n-1$
DSB [12]	$\frac{\log_2 \Omega}{r+1} + 2\log_2 q$	$\log_2 \Omega + 2(r+1)\log_2 q + \rho$	$\frac{n}{r+1} - 1$
DSB with LSS [12]	$\frac{\log_2 \Omega}{r} + \log_2 q$	$\log_2 \Omega + r \log_2 q$	$2 \cdot \frac{n}{r+1} - 1$
DSB with (r, δ) LSS	$\frac{\log_2 \Omega}{r} + \log_2 q$	$\log_2 \Omega + r \log_2 q$	$\delta \cdot \frac{n}{r+\delta-1} - 1$

Table 5.1: A table for the comparison of storage cost, communication cost and robustness to peer failure.

Algorithm 3 Algorithm for DSB with (r, δ) LSS

1: Set the hash value Z_τ as the global secret. Set $m_{00} = Z_\tau$ where $m_{00} \in \mathbb{F}_q$.
2: The message vector is $\mathbf{m} = [m_{00}\ v_\tau]$, where $v_\tau \in \mathbb{F}_q^{k-1}$ is a random vector.
3: **for** $l = 1$ to $\frac{n}{r+\delta-1}$ **do**
4: Set local secret $s_\tau(l) = y_0(\beta) = \sum_{j=0}^{\frac{k}{r}-1} m_{0j}(g(x))^j$ for $\beta \in \mathcal{A}_l$.
5: **end for**
6: Compute $(z_e^\tau(\alpha), \alpha \in \mathcal{A})$ by equation (4) and distribute among n peers.
7: **for** $l = 1$ to $\frac{n}{r+\delta-1}$ **do**
8: Encrypt B_τ with $s_\tau(l)$ to obtain $\mathbf{e}_\tau(l) = \mathbf{\Phi}(B_\tau; s_\tau(l))$.
9: Encode $\mathbf{e}_\tau(l)$ into $\mathbf{c}_\tau(l)$ by $(r + \delta - 1, r)$ code.
10: Distribute and store $\mathbf{c}_\tau(l)$ among peers in $\mathcal{A}_l$.
11: **end for**

B_τ and Z_τ. To recover data at a failed node one needs to download the entire data stored at another node. The data can still be accessible if there are $n-1$ node failures.

Distributed storage blockchain system(DSS) [12]: The n nodes are divided into $l = \frac{n}{r+1}$ subsets $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_{\frac{n}{r+1}}$ each of size $r+1$. In this, the message vector $B_\tau \in \mathbb{F}_\Omega^{r+1}$ at time τ, is first encrypted with a secret key $K_\tau(l) \in \mathbb{F}_q$ using the encoding function $\mathbf{\Phi}$ to obtain $\mathbf{e}_\tau(l)$. This $\mathbf{e}_\tau(l)$ is distributed and stored in $r+1$ nodes. $K_\tau(l)$ and Z_τ are stored using Shamir's $(r+1, r+1)$ secret sharing scheme. In the communication cost, ρ is the extra cost incurred to access $r+1$ nodes in another subset.

DSB with LSS [12]: The authors provide Distributed storage blockchain with local secret sharing scheme. Setting $\delta = 2$ in Algorithm 3 will give us this scheme. The optimal (r, δ) codes which are constructed in [16] are used in this system. This construction is explained in construction 2.16.

Consider n points over a finite field which are divided into $l = \frac{n}{r+1}$ subsets $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_{\frac{n}{r+1}}$, each of size $r+1$. Given a good polynomial $g(x)$, for any $\beta, \beta' \in \mathcal{A}_l$, $g(\beta) = g(\beta')$.

In this, the hash value Z_τ is set as a global secret in the encoding polynomial and $l = \frac{n}{r+1}$ local secrets $s_\tau(l)$ are computed using the local encoding polynomial. The encoding polynomial at time τ is $z_e^\tau(x)$. It is evaluated at n points $\mathcal{A}$. It is distributed and stored among the n nodes. Obtain $\mathbf{e}_\tau(l)$ using the message vector $B_\tau \in \mathbb{F}_\Omega^r$ and $s_\tau(l)$ at time τ. Encode $\mathbf{e}_\tau(l)$ into $\mathbf{c}_\tau(l)$ using $(r+1, r)$ coding. Distribute and store $\mathbf{c}_\tau(l)$ among $r+1$ peers in $\mathcal{A}_l$.

We explain the local secret sharing scheme here. The encoding polynomial given by

$$z_e(x) = \sum_{i=0}^{r-1} y_i(x) x^i.$$

This polynomial is described in equation (2.6), equation (2.7). The encoding polynomial can be

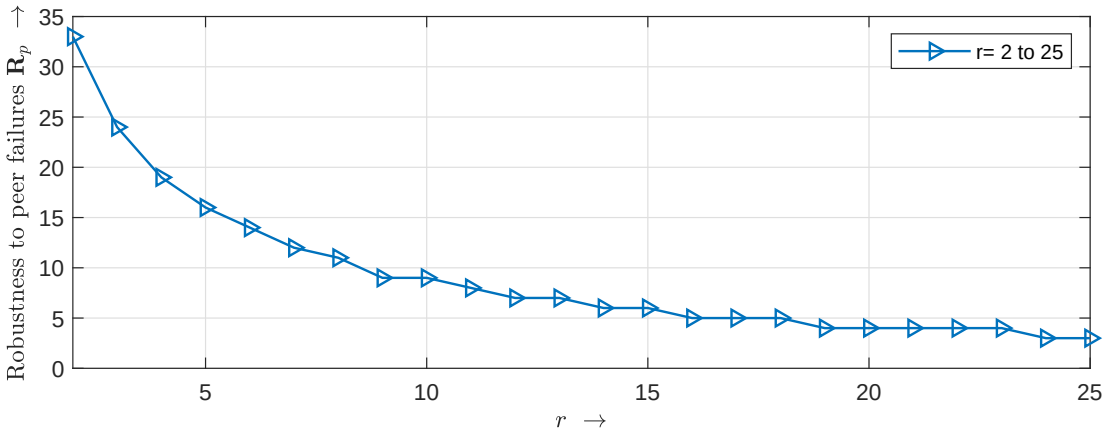

Figure 5.2: $\mathbf{R}_p$ for a DSB with LSS with $n = 50$ and $2 \leq r \leq 14$.

viewed as a polynomial of degree r with its coefficients being given by $y_i(\alpha), \alpha \in \mathcal{A}_l$. So, if we set $y_0(\beta)$ for a $\beta \in \mathcal{A}_l$ as secret this means the constant term of $z_e(x)$ is the secret. Lines 3, 4 and 5 in Algorithm 3 does this. This secret can be recovered by accessing any r other evaluations.

For this system, as r is increased, $\mathbf{R}_p$ decreases. This is shown in Fig. 5.2. For $n = 50$ we vary r from 1 to 25. Note that this construction is for $r + 1 \mid n$. As it can clearly be observed that as r is increased, $\mathbf{R}_p$ clearly decreases.

DSB with (r, δ) *LSS*: Consider n points over a finite field which are divided into $l = \frac{n}{r+\delta-1}$ sets $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_{\frac{n}{r+\delta-1}}$, each of size $r + \delta - 1$. Given a good polynomial $g(x)$, for any $\beta, \beta' \in \mathcal{A}_l$, $g(\beta) = g(\beta')$.

The scheme is illustrated using algorithm 3. The encoding polynomial is evaluated at the n points of $\mathcal{A}$. In this, the important secret Z_τ is stored as global secret. And the lesser important secret $s_\tau(l)$ is stored as local secret. This local secret is encoded locally using an (r, δ) local code.

For this system the storage cost is given by

$$\mathbf{S}_c = \frac{\log_2 \Omega}{r} + \log_2 q.$$

Similarly, the communication cost to recover the data in the event of a node failure is given by

$$\mathbf{C}_c = \log_2 \Omega + r \log_2 q.$$

This system can recover the data in each subset even in the presence of $\delta - 1$ node failures. If there are δ erasures in each subset, then the data cannot be recovered. Hence, the data can be recovered in

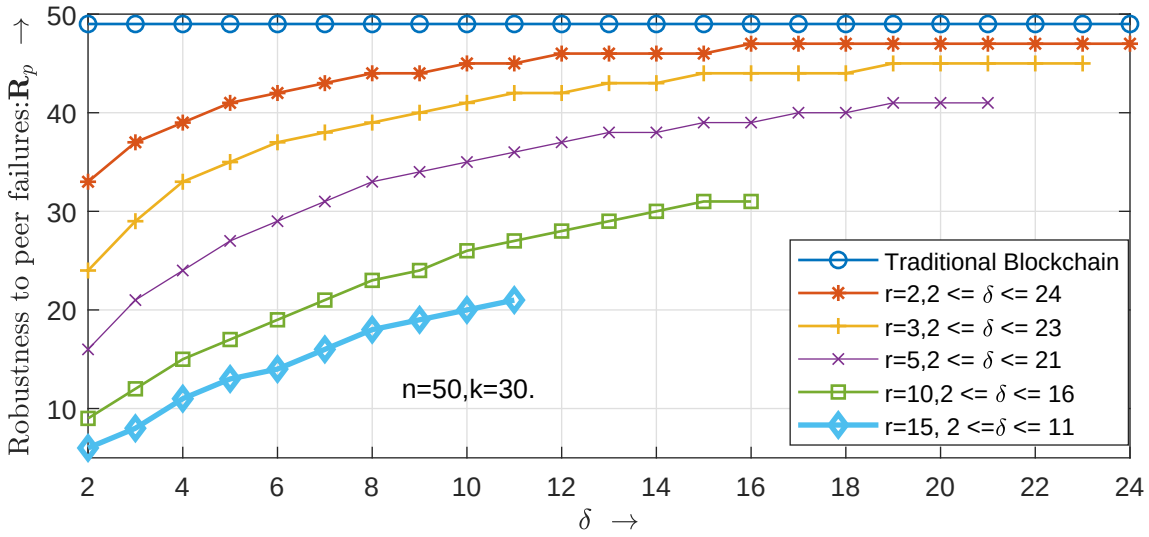

Figure 5.3: $\mathbf{R}_p$ for a DSB with (r, δ) LSS, with varying r and δ.

the presence of

$$\mathbf{R}_p = \delta \cdot \frac{n}{r + \delta - 1} - 1$$

erasures.

Availability to DSS with (r, δ) *LSS*: There will be $\mathbf{L} = \frac{n}{r+\delta-1}$ local codes. Any local code has t sets of r symbols each. These sets can be chosen at random. Hence, the system can provide $\mathbf{L} \cdot t$ copies of $s_\tau(l)$ as well as $c_\tau(l)$. Hence any local code can provide $\mathbf{L} \cdot t$ copies of the transaction B_τ.

In Fig. 5.3, we plot the graphs for $n = 50$. For a DSS with (r, δ) LSS, for different values of r and δ, the respective values of $\mathbf{R}_p$ is plotted. For traditional blockchain system, $\mathbf{R}_p$ is $n - 1 = 49$. Let $r, r = \{2, 3, 5, 10, 14\}$. For each value of r we vary $\delta \geq 2$ such that $\frac{n}{r+\delta-1} \leq 2$. When $r = 14, \delta = 2$, the gap between traditional blockchain system and DSS with (r, δ) LSS is huge. However as r is reduced and δ is increased, the gap reduces. For $r = 2, \delta = 14$, the gap is the least.

Hence, for a DSB with (r, δ) LSS, as r is kept small and δ is made large such that $r + \delta - 1 \mid n$, we have the robustness to peer failures ($\mathbf{R}_p$) much better than for a DSB with LSS (obtained by setting $\delta = 2$ [12]). This is because for any r, increasing δ means increasing t. As t increases, $\mathbf{L} \cdot t$ and $\mathbf{R}_p$ increases. It is shown that $\mathbf{R}_p$ of DSS with (r, δ) LSS can be made almost close to the traditional blockchain system. This is in addition to providing more copies of the transaction B_τ.

6

Conclusion

Contents

6.1 Summary of contributions

We consider codes with availability for distributed data storage and three issues concerning these codes are addressed in this thesis.

The results are summarized in the two tables 6.1 and 6.2. The first table summarizes the difference between the different class of codes studied in this thesis. The second table summarizes the optimal code constructions and the advantages of these codes.

Chapter	Upper bound	Availability	Recovery sets	Comments
3	$d \leq n-k+1-\sum_{l=1}^{t}\left(\left\lceil \frac{k}{r_l} \right\rceil -1\right)(\delta_l -1)$.	For individual code symbol	Recovery sets are fixed	Multiple local erasure correction
4	$d \leq n-k+1-\left\lceil \frac{kt}{r} \right\rceil (\delta-1)+t(\delta-1)$.	For multiple code symbols	Recovery sets are fixed	Multiple local erasure correction
5	$d \leq n-k+1-\left(\left\lceil \frac{k}{r} \right\rceil -1\right)((t-1)r+1)$	For individual code symbol	Recovery sets are dynamic	Multiple local erasure correction

Table 6.1: A Comparison of the results obtained in this thesis .

Chapter	Optimal codes	Membership matrices	Features
3	(i) For information availability (ii) For information availability and all symbol locality	Needed, Construction is an open problem.	Provides availability even in the presence of erasures
4	(i) For information availability, (ii) For information availability and all symbol locality	Needed, they exist for certain parameters	Provides availability for multiple symbols
5	For all symbol locality and all symbol availability	Not needed	Any r other symbols can be used as recovery set

Table 6.2: A summary of the features of codes studied in this thesis .

The features of the codes studied in this thesis are described in table 6.2. Based on one's require-

ments, an appropriate code can be chosen. For example, if you want to have a code with availability for certain parameters but the membership matrices do not exist for those parameters, one can chose the codes with availability with dynamic recovery sets (studied in chapter 5) to provide availability. This is because these codes do not need membership matrices.

The results obtained in this thesis are summarized chapter wise. The summary is given below.

6.1.1 Chapter 3: Availability with different localities and multiple local erasure correction.

- We define a new class of codes with availability. These are called as $[n, k, d]$ codes with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ availability. This code $\mathcal{C}$, has t distinct sub codes $\mathcal{C}_l$, $1 \leq l \leq t$. Each $\mathcal{C}_l$ is a sub code with (r_l, δ_l) locality [8]. In these codes, availability is provided for information symbols
- Any information symbol will be in t recovery sets $Q_l(1), \ldots, Q_l(t)$. Each recovery set $Q_l(i) \in \mathcal{C}_l$ has locality r_l and is protected by a local code of minimum Hamming distance at least δ_l.
- We show that these codes reduce the disk reads in the recovery of multiple erasures in a local code.
- Two upper bounds on the minimum Hamming distance of this code are derived.
- Optimal codes constructed using pyramid code construction with information locality have a unique property. We show that the optimal codes provide availability for any information symbol even when there are any $\delta_l - 2$ erasures in each of its t disjoint recovery sets.
- Optimal codes with information availability and all symbol locality are constructed using Gabidulin codes.
- When designed carefully, these codes allow us to recover all k information symbols t times simultaneously. This property is called as information set availability.

6.1.2 Chapter 4: Codes with Availability for Multiple code symbols

- Codes with availability are studied in literature. These codes focus on providing availability for individual code symbols.

- We consider a new class of codes with availability. These codes provide availability for a subset of code symbols.
- These codes provide availability for more symbols with the same locality. This is an advantage because, more symbols are recovered with fewer disk reads. These codes are referred to as $[n, k, r, \rho, t, \delta]$ codes.
- Optimal codes with information availability are constructed using pyramid constructions. Optimal codes with information availability and all symbol locality are constructed using Gabidulin codes.
- A new class of membership matrices are introduced. They provide availability for a subset of code symbols instead of a single code symbol. The membership matrices studied in literature are a special case of these new membership matrices.
- We provide a construction of these membership matrices.

6.1.3 Chapter 5: Dynamic Recovery sets for Codes with Availability and their Applications to Blockchain

- Codes with availability are studied in literature for the case where the disjoint recovery sets are fixed.
- If any other code symbol is lost in the local code of an information symbol, then this local code cannot be used for providing availability.
- Membership matrices do not exist for many parameters. Hence optimal codes do not exist for many parameters.
- We define a new class of codes. These codes provide dynamic recovery sets i.e., any r other code symbols can be used to recover a code symbol.
- We derive an upper bound on the minimum Hamming distance of these codes.
- We show that optimal (r, δ) codes [8] with $\delta = (t-1)r + 2$ constructed using the polynomial based construction given by [16] has all these properties.
- We show that these optimal codes have interesting applications in distributed storage blockchain system.

- Using these codes, we show that as r is kept low and t is increased, the ability of the system to recover the transaction from node failures increases, i.e., the system can recover the transaction when many nodes fail.

6.2 Problems for Future work

The following research problems were encountered while deriving the results in this thesis.

- Providing availability for all the code symbols even when some symbols are erased in the local codes. (An extension of the result obtained in section 3.5 for all the code symbols)
- Construction of membership matrices for codes with $\{(r_1, \delta_1), \cdots, (r_t, \delta_t)\}$ availability.
- Construction of membership matrices for $[n, k, r, \rho, t, \delta]$ code i.e., membership matrices for codes with multiple code symbols.
- Deriving a much general upper bound on the minimum Hamming distance for codes with dynamic recovery sets.

6.3 Conclusion

In this thesis, we address three issues which are related to codes with availability. We show that all these three issues can be resolved in the storage system, by providing any local code, the ability to recovery multiple erasures locally. Hence these codes are referred to as codes with availability and multiple local erasure correction. We introduce three classes of codes with availability and multiple local erasure correction. We derive an upper bound on the minimum Hamming distance of these three codes. We provide optimal code constructions.

Bibliography

[1] B. Calder, J. Wang, A. Ogus *et al.*, "Windows azure storage: a highly available cloud storage service with strong consistency," in *SOSP'11 - Proceedings of the 23rd ACM Symposium on Operating Systems Principles*, Oct 2011, pp. 143–157.

[2] Hadoop. [Online]. Available: https://hadoop.apache.org/

[3] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the society for industrial and applied mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[4] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE transactions on information theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[5] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 476–489, 2011.

[6] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Transactions on Information Theory*, vol. 58, no. 11, pp. 6925–6934, Nov 2012.

[7] C. Huang, Simitci *et al.*, "Erasure coding in windows azure storage," in *USENIX ATC'12*, 2012, p. 2.

[8] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar, "Codes with local regeneration and erasure correction," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4637–4660, Aug 2014.

[9] A. S. Rawat, D. S. Papailiopoulos, A. G. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," *IEEE Transactions on Information Theory*, vol. 62, no. 8, pp. 4481–4493, Aug 2016.

[10] R. K. Raman and L. R. Varshney, "Dynamic distributed storage for blockchains," in *2018 IEEE International Symposium on Information Theory (ISIT)*, 2018, pp. 2619–2623.

[11] ——, "Distributed storage meets secret sharing on the blockchain," in *2018 Information Theory and Applications Workshop*, 2018, pp. 1–6.

[12] Y. Kim, R. K. Raman, Y. Kim, L. R. Varshney, and N. R. Shanbhag, "Efficient local secret sharing for distributed blockchain systems," *IEEE Communications Letters*, vol. 23, no. 2, pp. 282–285, 2019.

[13] B. Schroeder and G. A. Gibson, "Understanding disk failure rates: What does an MTTF of 1,000,000 hours mean to you?" *ACM Transactions on Storage (TOS)*, vol. 3, no. 3, pp. 8–es, 2007.

[14] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer *et al.*, "Oceanstore: An architecture for global-scale persistent storage," *ACM SIGOPS Operating Systems Review*, vol. 34, no. 5, pp. 190–201, 2000.

[15] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson, "Raid: High-performance, reliable secondary storage," *ACM Computing Surveys (CSUR)*, vol. 26, no. 2, pp. 145–185, 1994.

[16] I. Tamo and A. Barg, "A family of optimal locally recoverable codes," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4661–4676, Aug 2014.

[17] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Transactions on Information Theory*, vol. 59, no. 3, pp. 1597–1616, 2013.

[18] O. Khan, R. Burns, J. Park, and C. Huang, "In search of i/o-optimal recovery from disk failures," in *Proceedings of the 3rd USENIX Conference on Hot Topics in Storage and File Systems*, ser. HotStorage'11. USA: USENIX Association, 2011, p. 6.

[19] F. Oggier and A. Datta, "Self-repairing homomorphic codes for distributed storage systems," in *2011 Proceedings IEEE INFOCOM*, 2011, pp. 1215–1223.

[20] D. S. Papailiopoulos, J. Luo, A. G. Dimakis, C. Huang, and J. Li, "Simple regenerating codes: Network coding for cloud storage," in *2012 Proceedings IEEE INFOCOM*, 2012, pp. 2801–2805.

[21] I. Tamo, D. S. Papailiopoulos, and A. G. Dimakis, "Optimal locally repairable codes and connections to matroid theory," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 6661–6671, 2016.

[22] N. Silberstein, A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath, "Optimal locally repairable codes via rank-metric codes," in *2013 IEEE International Symposium on Information Theory*, 2013, pp. 1819–1823.

[23] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, 2011.

[24] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Distributed storage codes with repair-by-transfer and non-achievability of interior points on the storage-bandwidth tradeoff," *IEEE Transactions on Information Theory*, vol. 58, no. 3, pp. 1837–1852, 2011.

[25] B. Sasidharan, K. Senthoor, and P. V. Kumar, "An improved outer bound on the storage-repair-bandwidth tradeoff of exact-repair regenerating codes," in *2014 IEEE International Symposium on Information Theory*, 2014, pp. 2430–2434.

[26] C. Tian, "Rate region of the (4, 3, 3) exact-repair regenerating codes," in *2013 IEEE International Symposium on Information Theory*, 2013, pp. 1426–1430.

[27] C. Suh and K. Ramchandran, "Exact-repair MDS code construction using interference alignment," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1425–1442, 2011.

[28] D. S. Papailiopoulos, A. G. Dimakis, and V. R. Cadambe, "Repair optimal erasure codes through hadamard designs," *IEEE Transactions on Information Theory*, vol. 59, no. 5, pp. 3021–3037, 2013.

[29] N. B. Shah, K. Rashmi, P. V. Kumar, and K. Ramchandran, "Interference alignment in regenerating codes for distributed storage: Necessity and code constructions," *IEEE Transactions on Information Theory*, vol. 58, no. 4, pp. 2134–2158, 2011.

[30] V. R. Cadambe, S. A. Jafar, H. Maleki, K. Ramchandran, and C. Suh, "Asymptotic interference alignment for optimal repair of MDS codes in distributed storage," *IEEE Transactions on Information Theory*, vol. 59, no. 5, pp. 2974–2987, 2013.

[31] Y. Hu, C.-M. Yu, Y. K. Li, P. P. Lee, and J. C. Lui, "NCFS: On the practicality and extensibility of a network-coding-based distributed file system," in *2011 International Symposium on Networking Coding*, 2011, pp. 1–6.

[32] Y. Hu, H. C. Chen, P. P. Lee, and Y. Tang, "NCCloud: applying network coding for the storage repair in a cloud-of-clouds," in *FAST*, 2012, p. 21.

[33] A. Duminuco and E. Biersack, "A practical study of regenerating codes for peer-to-peer backup systems," in *2009 29th IEEE International Conference on Distributed Computing Systems*, 2009, pp. 376–384.

[34] K. Rashmi, N. B. Shah, D. Gu, H. Kuang, D. Borthakur, and K. Ramchandran, "A solution to the network challenges of data recovery in erasure-coded distributed storage systems: A study on the facebook warehouse cluster," in *5th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 13)*, 2013.

[35] M. Blaum, P. G. Farrell, and H. C. van Tilborg, "Array codes," in *Handbook of Coding Theory, vol. 2.* Elsevier, 1998, pp. 1855–1909.

[36] M. Blaum, J. Brady, J. Bruck, and J. Menon, "Evenodd: An efficient scheme for tolerating double disk failures in raid architectures," *IEEE Transactions on computers*, vol. 44, no. 2, pp. 192–202, 1995.

[37] M. Blaum, J. Brady, J. Bruck, J. Menon, and A. Vardy, "The evenodd code and its generalization," *High Performance Mass Storage and Parallel I/O*, pp. 187–208, 2001.

[38] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, "Row-diagonal parity for double disk failure correction," in *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*, 2004, pp. 1–14.

[39] C. Huang, M. Chen, and J. Li, "Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems," in *Sixth IEEE International Symposium on Network Computing and Applications (NCA 2007)*, July 2007, pp. 79–86.

[40] J. Katz and L. Trevisan, "On the efficiency of local decoding procedures for error-correcting codes," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, 2000, pp. 80–86.

[41] S. Yekhanin, "Towards 3-query locally decodable codes of subexponential length," *Journal of the ACM (JACM)*, vol. 55, no. 1, pp. 1–16, 2008.

[42] K. Efremenko, "3-query locally decodable codes of subexponential length," *SIAM Journal on Computing*, vol. 41, no. 6, pp. 1694–1703, 2012.

[43] S. Kopparty, S. Saraf, and S. Yekhanin, "High-rate codes with sublinear-time decoding," *Journal of the ACM (JACM)*, vol. 61, no. 5, pp. 1–20, 2014.

[44] S. Yekhanin *et al.*, *Locally decodable codes*. Citeseer, 2012.

[45] S. Kadhe and A. Sprintson, "Codes with unequal locality," in *2016 IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 435–439.

[46] G. Kim and J. Lee, "Locally repairable codes with unequal local erasure correction," *IEEE Transactions on Information Theory*, vol. 64, no. 11, pp. 7137–7152, 2018.

[47] V. R. Cadambe and A. Mazumdar, "Bounds on the size of locally recoverable codes," *IEEE Transactions on Information Theory*, vol. 61, no. 11, pp. 5787–5794, Nov 2015.

[48] J. Han and L. A. Lastras-Montano, "Reliable memories with subline accesses," in *2007 IEEE International Symposium on Information Theory*, 2007, pp. 2531–2535.

[49] D. S. Papailiopoulos and A. G. Dimakis, "Locally repairable codes," *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 5843–5855, 2014.

[50] N. Silberstein, A. S. Rawat, and S. Vishwanath, "Error resilience in distributed storage via rank-metric codes," in *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2012, pp. 1150–1157.

[51] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 60, no. 1, pp. 212–236, Jan.2014.

[52] S. Goparaju and R. Calderbank, "Binary cyclic codes that are locally repairable," in *2014 IEEE International Symposium on Information Theory*, 2014, pp. 676–680.

[53] N. Silberstein and A. Zeh, "Optimal binary locally repairable codes via anticodes," in *2015 IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 1247–1251.

[54] A. Zeh and E. Yaakobi, "Optimal linear and cyclic locally repairable codes over small fields," in *2015 IEEE Information Theory Workshop (ITW)*, 2015, pp. 1–5.

[55] M. Forbes and S. Yekhanin, "On the locality of codeword symbols in non-linear codes," *Discrete mathematics*, vol. 324, pp. 78–84, 2014.

[56] E. M. Gabidulin, "Theory of codes with maximum rank distance," *Problemy Peredachi Informatsii*, vol. 21, no. 1, pp. 3–16, 1985.

[57] R. M. Roth, "Maximum-rank array codes and their application to criss-cross error correction," *IEEE transactions on Information Theory*, vol. 37, no. 2, pp. 328–336, 1991.

[58] E. M. Gabidulin and N. I. Pilipchuk, "Error and erasure correcting algorithms for rank codes," *Designs, codes and Cryptography*, vol. 49, no. 1-3, pp. 105–122, 2008.

[59] T. Westerbäck, R. Freij-Hollanti, and C. Hollanti, "Applications of polymatroid theory to distributed storage systems," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sep. 2015, pp. 231–237.

[60] S. Bhadane and A. Thangaraj, "Unequal locality and recovery for locally recoverable codes with availability," in *2017 Twenty-third National Conference on Communications (NCC)*, March 2017, pp. 1–6.

[61] A. Zeh and E. Yaakobi, "Bounds and constructions of codes with multiple localities," in *2016 IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 640–644.

[62] A. Wang and Z. Zhang, "Repair locality with multiple erasure tolerance," *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 6979–6987, Nov 2014.

[63] L. Pamies-Juarez, H. D. Hollmann, and F. Oggier, "Locally repairable codes with multiple repair alternatives," in *2013 IEEE international symposium on information theory*, 2013, pp. 892–896.

[64] M. Asteris and A. G. Dimakis, "Repairable fountain codes," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 1037–1047, 2014.

[65] S. Kruglik and A. Frolov, "Bounds and constructions of codes with all-symbol locality and availability," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 1023–1027.

[66] S. Kruglik, K. Nazirkhanova, and A. Frolov, "On distance properties of(r, t, x)-lrc codes," in *2018 IEEE International Symposium on Information Theory (ISIT)*, June 2018, pp. 1336–1339.

[67] S. B. Balaji and P. V. Kumar, "Bounds on the rate and minimum distance of codes with availability," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 3155–3159.

[68] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[69] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK: Research perspectives and challenges for bitcoin and cryptocurrencies," in *2015 IEEE Symposium on Security and Privacy*, 2015, pp. 104–121.

[70] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and cryptocurrency technologies: a comprehensive introduction*. Princeton University Press, 2016.

[71] S. Underwood, "Blockchain beyond bitcoin," 2016.

[72] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "Medrec: Using blockchain for medical data access and permission management," in *2016 2nd International Conference on Open and Big Data (OBD)*, 2016, pp. 25–30.

[73] M. J. Casey and P. Wong, "Global supply chains are about to get better, thanks to blockchain," *Harvard business review*, vol. 13, pp. 1–6, 2017.

[74] M. Swan, *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

[75] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *2016 IEEE symposium on security and privacy (SP)*, 2016, pp. 839–858.

[76] M. Iansiti and K. R. Lakhani, "The truth about blockchain harvard business review," *Harvard University, hbr. org/2017/01/the-truth-about-blockchain, accessed date: February*, vol. 2, p. 2019, 2017.

[77] Y. Su, "On the construction of local parities for (r, t) -availability in distributed storage," *IEEE Transactions on Communications*, vol. 65, no. 6, pp. 2332–2344, June 2017.

[78] A. S. Rawat, Z. Song, A. G. Dimakis, and A. Gál, "Batch codes through dense graphs without short cycles," *IEEE Transactions on Information Theory*, vol. 62, no. 4, pp. 1592–1604, 2016.

List of Publications

Journals:

- U. D. Kadiyam and S. Das, "A Class of Codes With Availability and Multiple Local-Erasures Correction," in *IEEE Communications Letters*, vol. 24, no. 10, pp. 2087-2091, Oct. 2020, doi: 10.1109/LCOMM.2020.3004757. (from chapter 3).
- U. D. Kadiyam and S. Das, "Codes with Availability for Multiple Code Symbols," in *IEEE Communications Letters*, doi: 10.1109/LCOMM.2021.3079358 (from chapter 4).

Conferences:

- U. D. Kadiyam and S. Das, "Codes with Availability and different Localities," 2020 International Conference on Signal Processing and Communications (SPCOM), 2020, pp. 1-5, doi: 10.1109/SPCOM50965.2020.9179518 (from chapter 3).
- U. D. Kadiyam, "Locality and Availability with Multiple Erasure Correction," 2018 International Conference on Signal Processing and Communications (SPCOM), 2018, pp. 16-20, doi: 10.1109/SPCOM.2018.8724452 (from chapter 3).