On Heterogeneous Distributed Storage Systems: Bounds and Code Constructions

by

KRISHNA GOPAL
201221007

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

July, 2019

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Doctor of Philosophy at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

Krishna Gopal

Certificate

This is to certify that the thesis work entitled ON HETEROGENEOUS DISTRIBUTED STORAGE SYSTEMS: BOUNDS AND CODE CONSTRUCTIONS has been carried out by KRISHNA GOPAL for the degree of Doctor of Philosophy at *Dhirubhai Ambani Institute of Information and Communication Technology* under my supervision.

Professor Manish K Gupta
Thesis Supervisor

Acknowledgments

Vande Vissnnum Bhava-Bhaya-Haram Sarva-Loka-Eka-Naatham ||

It is my immense pleasure to express my sincere gratitude to my advisor Prof. Manish K Gupta for his endless support for my journey of Ph.D. His guidance helped me in all my research and writing of this thesis.

Next, I would like to thank the rest of my committee: Prof. Ashok Amin, Prof. V. Sunitha, Prof. Gagan Garg, Prof. Rahul Muthu, and Prof. Nabin Kumar Sahu for their insightful comments related to my research. I thank Prof. Sumn Mitra and other faculty members for academic support during the entire journey of the Ph.D. programme. My sincere thanks to DAIICT for supporting me to attend the conferences and paper presentations. I am also grateful to the staff members (specially Help Desk, and Resource Centre) for their support to provide me with resources throughout my research. I thank my friends and colleagues: Dr. Dixita Limbachiya and other for encouraging me spiritually during a hard time in the last few years.

Last but not the least, I would like to thank my father Mr. Dilip Kumar Benerjee, my mother late Shukla Benerjee, my aunt and uncle Mrs. & Mr. Doli Lalit Kumar Banerjee, and to my sister and brother-in-law Mrs. & Mr. Sujata Sameer Gauswami, and my cousin Mrs. & Mr. Sangita Devashish Banerjee for supporting me spiritually throughout this journey and for my life.

Krishna Gopal Benerjee

Contents

Abstract

In *Distributed Storage Systems* (DSSs), usually, data is stored using encoded packets on different chunk servers. In this thesis, we have considered heterogeneous DSSs in which each node may store a different number of packets and each having different repair bandwidth. In particular, a data collector can reconstruct the file at time t using some specific nodes in the system, and for arbitrary node failure, the system can be repaired by some set of arbitrary nodes. Using *min-cut* bound, we investigate the fundamental trade-off between storage and repair cost for our model of heterogeneous DSS. In particular, the problem is formulated as a bi-objective optimization linear programming problem. For an arbitrary DSS, it is shown that the calculated *min-cut* bound is tight.

For a DSS with symmetric parameters, a well known class of *Distributed Replication-based Simple Storage (DRESS)* codes is *Fractional Repetition* (FR) code. In such systems, the replicas of data packets encoded by Maximum Distance Separable (MDS) code, are stored on distributed nodes. Most of the available constructions for the FR codes are based on combinatorial designs and Graph theory. In this thesis, FR codes with generalized parameters (such as replication factor of each packet are not same and storage capacity of each node are also not same) are considered, and it is called as *Generalized Fractional Repetition* (GFR) code. For the GFR code, we propose an elegant sequence-based approach for the construction of the GFR code called *Flower codes*. Further, it is shown that any GFR code is equivalent to a Flower code. The condition for the universally good GFR code is given on such sequences. For some sequences, the universally good GFR codes are explored. In general, for the GFR codes with non-uniform parameters, bounds on the GFR code rate and DSS code rate are studied. Further, we have

shown that a GFR code corresponds to a hypergraph. Using the correspondence, properties and bounds of a hypergraph are directly mapped to the associated GFR code. In general, necessary and sufficient conditions for the existence of a GFR code is obtained using the correspondence. It is also shown that any GFR code associated with a linear hypergraph is universally good.

Motivation

The distributed storage systems have been developed to store huge amount of data such that it allows data accessibility, in a reliable manner, at any time and anywhere. Many companies such as Microsoft [47, 68], Amazon [8], Google [33], Facebook [93], etc. use such storage services through data centers which are distributed along a network. In storage systems like Google file system [33], multiple copies of data fragments are stored in a manner such that the system is reliable and the file can be retrieved from the system. At the same level of redundancy, efficient coding techniques can make the storage system more reliable [115]. In a particular coding technique, a source file is broken into k packets, and those packets are encoded into n packets. The packets are stored on n distinct nodes such that any k nodes out of the n nodes have sufficient information to retrieve the complete file. However, if some erasure codes, such as Reed-Solomon codes, are used to store data in such storage system then the complete file has to be reconstructed to repair a single node failure, which requires a huge amount of bandwidth and storage space. In [23], Dimakis *et al.* introduced *regenerating code* for Distributed Storage Systems (DSSs) which optimizes the repair bandwidth and the storage space. The parameters of the DSS considered in [23] are all symmetric and are known as homogeneous DSS. For the homogeneous DSS, the fundamental trade-off between the *node storage capacity* (the number of packets stored in a node) and the *repair bandwidth* (the total number of packets communicated during repair) is plotted in [23, 119]. Further, researchers have studied the trade-off for DSS with some asymmetric parameters [6, 7, 29, 83, 122], and constructed codes achieving the optimal points on the trade-off [26, 38, 88] with additional system properties.

Objectives

Objectives of the thesis are as follows.

- Calculating the Fundamental bound for a heterogeneous DSS.
- Using the Fundamental bound, finding the trade-off curve between the cost of node storage and the cost of repair bandwidth for the heterogeneous DSS.
- Using the Fundamental bound, finding bounds on the parameters for the existence of various specific DSSs.
- Construction of the universally good GFR codes for the heterogeneous DSSs.
- Finding bounds on the parameters of GFR codes defined on various heterogeneous DSSs.

List of Principal Symbols and Acronyms

$\mathbb{Z}$	Set of integers
i, j	Indexes
$\mathbb{F}_q$	Field with q symbols
M	Number of information packets
n	Number of nodes
U_i	Node with index i
α_i	Node storage capacity of the node U_i
α	Maximum storage capacity among all nodes
k	Maximum reconstruction degree
$M(k)$	Maximum number of distinct packets stored in any k nodes
$\mathcal{A}$	Reconstruction set
$\mathscr{A}$	Set of all reconstruction sets
d_i	Maximum repair degree of the node U_i
d	Maximum of all the repair degrees
$S_\ell(i)$	Surviving set with index ℓ and associated with node U_i
τ_i	Cardinality of the set $S_\ell(i)$
$\vec{u}$	Node sequence on the reconstruction set $\mathcal{A}$

$\mathscr{A}(\mathcal{A})$	Node sequence set for the reconstruction set $\mathcal{A}$
$\vec{s}$	Surviving sequence associated with the node sequence $\vec{u}$
$(\vec{u})$	Surviving sequence set associated with the node sequence $\vec{u}$
$\beta(i,j,\ell)$	Number of packets downloaded from $U_j \in S_\ell(i)$ to repair U_i
$\gamma(i,\ell)$	Repair bandwidth associated with $S_\ell(i)$ for the failed node U_i
$\vec{c}$	Storage cost vector for the failed node U_i
$\vec{r}$	Repair cost vector
$r(\beta_i)$	Node repair cost for the node U_i
$r(\vec{\beta}_i)$	Node repair cost vector
$\mathcal{G}$	Information Flow Graph
θ	Number of encoded Packets
P_j	Encoded packet indexed with j
ρ_j	Replication factor of the packet P_j
ρ	Maximum replication factor among all packets
$\vec{\rho}$	Replication vector of length θ on Packet replication factor
$\vec{\alpha}$	Node storage vector of length n on node storage capacity
$\mathscr{C}$	GFR code
$\mathscr{C} : (n,\theta,\vec{\alpha},\vec{\rho})$	GFR code with parameters $(n,\theta,\vec{\alpha},\vec{\rho})$
$\mathscr{C}^*$	Dual of GFR code $\mathscr{C}$
B	Node Packet Distribution Incidence Matrix of a GFR code
A	Node Adjacency Matrix of a GFR code
$\mathcal{R}_{\mathscr{C}}(k)$	Code rate of a GFR code with reconstruction degree k

$\mathcal{R}_{DSS}(k)$	Code rate of a DSS with reconstruction degree k
$\vec{x}$	Sequence
$\mathbf{x}$	Binary Sequence
$w_{\mathbf{x}}$	Entry sum of the elements of the finite binary sequence $\mathbf{x}$
$w_{\mathbf{x}}(r)$	Entry sum of the first r elements of the finite binary sequence $\mathbf{x}$
$\mathbf{xy}$	Concatenation of binary binary sequences $\mathbf{x}$ and $\mathbf{y}$
v_i	Hypervertex with index i
V	Hypervertex set
E	Hyperedge
$\mathcal{E}$	Set of all Hyperedges
$(V, \mathcal{E})$	Hypergraph with Hypervertex set V and Hyperedge set $\mathcal{E}$
$\mathscr{H}(\mathcal{V}; \mathcal{E})$	Graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$
ϕ	Bijection between Hypergraph and Fractional Repetition code
$\emptyset$	Empty set
$\square$	End of the proof

List of Tables

List of Figures

CHAPTER 1
Introduction

A message with content and clarity Has gotten to be quite a rarity. To combat the terror of serious error, Use bits of appropriate parity. -*S. W. Golomb* [72]

Today a common man cannot imagine life without phone, computers, etc. In all these devices, messages (data) are either stored or communicated among various objects. Message transmission (communication of a message between two objects or storage of message information in some object) is always subject to noise (additional unwanted messages). For example, storing data in a compact disk or communication between two cell phones is not free from error or noise. Therefore, it is quite essential to encode the transmitting data in such a way that the original information can be decoded and obtained from the noisy message. A general framework of transmitting a message along a noisy channel is shown in Figure 1.1. A message is encoded using a suitable encoding scheme, and then it is sent along the noisy channel. Because of this, the encoded message is distorted, and the receiver receives the changed message. The receiver decodes the message from the received message using the respective decoding scheme and gives it to the user. For example, the transmission of message YES or NO is illustrated in the Figure 1.2. In the example, the message YES is encoded by 000 and the message NO is encoded by 111. Suppose we transmit the message YES, the encoder sends 000 along the noisy channel and the receiver receives 100 in place of 000 because of noise in the channel. The received vector 100 is closer to 000 than the 111. Therefore,

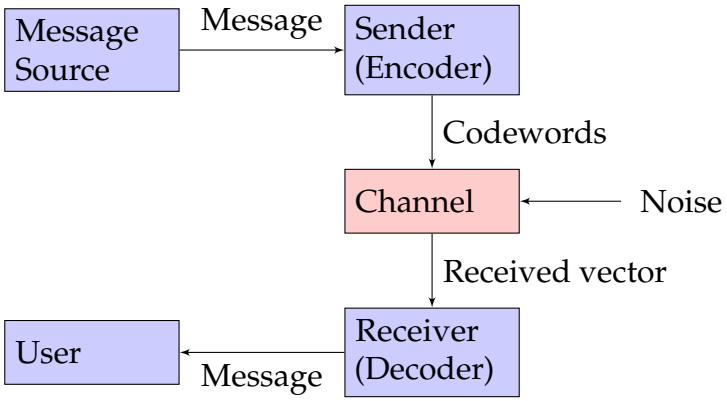

Figure 1.1: Message transmission along a noisy channel.

the receiver can conclude that the encoded message 000 was sent and hence, the message is YES. A code is the set of all encoded messages and the elements of the code are known codewords. In this example the code is $\{000, 111\}$, and 000 and 111 are all codewords. A code that detects errors in a received message is called an error detecting code. Also, if it can correct the error, it is called an error correcting code. In the example (Figure 1.2), at-most two-bit flip errors can be detected, and a one-bit flip error can be corrected. For more details about coding theory see [40].

In a seminal paper [39], Richard Hamming introduced basic technique of correcting single error (although a related code was known in 1945). In 1948, Claude Shannon published an article *A Mathematical Theory of Communication* in the Bell System Technical Journal [97]. The article discusses the maximum rate at which data information can be transmitted over a noisy communication channel with a specified bandwidth. In 1949, the binary Golay code was developed [34] which is a four error-detecting and three error-correcting code in a 24-bit length. The Hamming distance between two codewords (encoded messages) of the same length is the total number of positions at which symbols differ in both codewords. A message encoded using the Hamming code can detect 2 errors and correct 1 error. In Coding Theory, some fundamental bounds like Singleton bound, Sphere packing bound and Plotkin bound are obtained. These are the bounds on the number of codewords for given code length and minimum Hamming distance of a code. The Singleton bound was established by Richard Collom Singleton (1964) [106]. For an arbitrary code, the Singleton bound is the tight upper bound on the number of

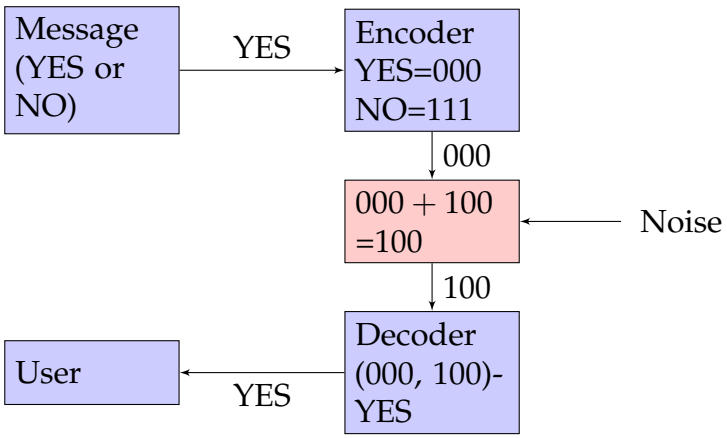

Figure 1.2: An example of transmission of the Message YES along a noisy channel.

distinct codewords with the codeword length and minimum Hamming distance of the code. A code, satisfying the Singleton bound with equality is known as the Maximum Distance Separable (MDS) code. For the MDS code, any k codewords out of the n codewords have sufficient information of the complete message. Various branches of coding theory have been developed such as Source coding (for data compression), Channel coding (for quick and efficient transmission of many valid codewords), Cryptographical coding (for secure communication), Line coding (for baseband transmission), and Network coding (for transmission among nodes in a network).

A network is represented by an acyclic directed connected graph, where each directed link has some weight (called capacity of the link). In a network, the maximum possible throughput from the source node to the destination node is called the maximum flow of the network. In 1956, L. R. Ford Jr. and D. R. Fulkerson proved the maximum flow is upper bounded by the minimum capacity of all cuts between these two nodes (source node and destination node) using max-flow-min-cut theorem, where the minimum capacity of a cut is the sum of the capacities of all the links on a cut [30]. For a multicast network (network in which group of data transmission is addressed to some destination nodes from a group of source nodes simultaneously), Ahlswede *et al.* (2000), in the article [2], proved that the network capacity could be achieved using network coding on the intermediate nodes. On an intermediate node, the network coding is combining some incoming packets into some outgoing packets. For a multicast linear network (network with two

source node and two destination node), the butterfly network [2] is a rapidly used example.

Over the last few decades, because of the explosive growth of information and data, storage and managing data have become a significant challenge. The storage capacity of many data centers exceeds the limit of the petabytes and exabytes. Any data center is limited by bandwidth cost (cost associated with the communication of information between various ends), processing power, system capacity cost, time management, and maintenance cost, etc. To overcome this, the industry has turned to a Distributed Storage System (DSS) which is running on low-cost common servers. Cloud storage is a DSS in which information is stored on distinct nodes as encoded packets in a redundant manner. One can retrieve the whole file by contacting some other nodes in the system. In a DSS, a failed node can be repaired using some other active nodes. In DSS, one has to optimize various parameters in the system such as storage capacity, repair bandwidth, availability, reliability, security, and scalability. DSS is used by many commercial systems like Facebook, Yahoo, IBM, Amazon, and Microsoft Windows Azure system [93, 47, 68, 8]. Using coding theory techniques such as erasure code, data reliability can be improved for the DSS keeping the same level of redundancy [116, 115]. For a system, redundancy is added using erasure codes to tolerate failures. Inspired by network coding, Dimakis *et al.* proposed a family of regenerating codes in [23], where storage cost and cost of repair bandwidth is optimum for the regenerating codes. Dimakis *et al.* considered a DSS model of n nodes in which the whole file can be recovered by downloading packets from any k (*reconstruction degree*) nodes, and a failed node can be repaired by replacing it with a new node, where the packets of the new node are downloaded from any d (*repair degree*) nodes. In this DSS model, for node failure, the repair bandwidth is the total number of packets downloaded from d nodes. In the Figure 1.3(a), an example of a DSS with 4 nodes is illustrated. In the DSS, a file is divided into 2 information packets x_1 and x_2 of same size. The two packets are then stored in 4 nodes U_1, U_2, U_3 and U_4 using repetition code. For the DSS, a data collector needs to connect some specific nodes (U_1 and U_3, or U_2 and U_4) to get the complete file information. Note that if any of the pair of

nodes out of the pairs (U_1, U_2) or (U_3, U_4) fails simultaneously, then the system can not be repaired anymore. Using coding theory, one can design a DSS with 4 nodes, which is more efficient than the DSS, given in the Figure 1.3(a). A DSS on 4 nodes is illustrated in the Figure 1.3(b) with both the reconstruction degree and the repair degree are 2. In the DSS, let a file be divided into 2 information packets x_1 and x_2 of the same size. A data collector can recover the complete file information by connecting any 2 nodes of the $(4, 2, 2)$ DSS. In the DSS, if any two nodes fail simultaneously, then the system can be repaired. So, the DSS (Figure 1.3(b)) is more efficient and reliable than the privious example Figure 1.3(a), at the same level of redundancy. In the DSS (Figure 1.3(b)), let the file of 4 MB data be stored in the DSS, where each packet is of size 2 MB. In the DSS, if a node fails, then it can be repaired by downloading 2 packets from any 2 helper nodes *i.e.*, the sum of 4 MB data needs to be downloaded to repair the node failure. If we allow network coding on nodes, then nodes can be repaired by downloading less data sum. In other words, one can design a DSS using network coding such that a node can be repaired by downloading data less than the 4 MB data from helper nodes. An example of such kind of DSS is given in the Fig 1.3(c). In the DSS, a file is divided into 4 packets y_1, y_2, y_3 and y_4 of same size, *i.e.*, a file of size 4 MB is divided into 4 packets and the size of each packet is 1 MB. In the DSS, a data collector can get the complete file information by connecting any 2 nodes. In the DSS, if a node fails, then the node can be repaired by downloading packets from any 2 helper nodes. For example, if node U_4 fails, then it can be repaired by replacing a new node U_4' with the same data information. The packet for the new node U_4' can be downloaded from the remaining 3 nodes. Since the computation on nodes is allowed, the packet y_1 from the node U_1, the one packet $y_3 + y_4$ from the node U_2 and the packet $y_2 + y_4$ from the node U_3 are downloaded for the new node U_4'.

The DSS model considered by Dimakis *et al.* has symmetric parameters. For such DSS, a family of regenerating codes with minimum cost of repair bandwidth is introduced in [26], and it is called Fractional Repetition (FR) codes. However, DSSs with asymmetric parameters are more close to the real-world scenarios. In this thesis, for such heterogeneous DSS, the fundamental bound on the file size is

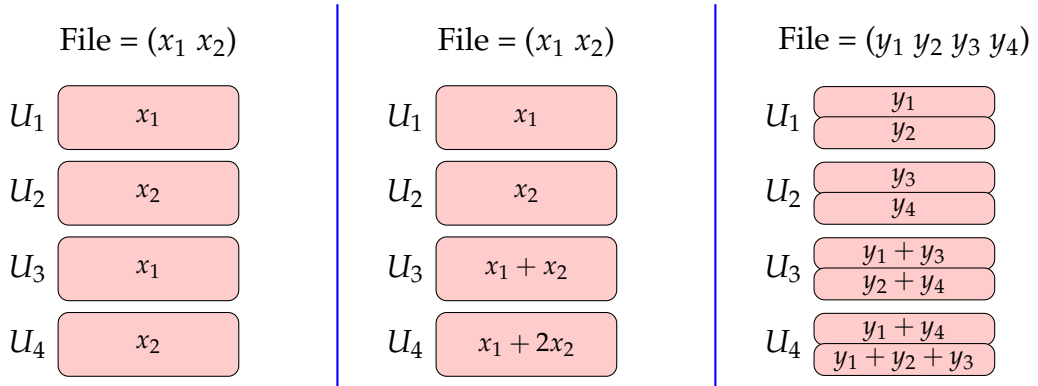

Figure 1.3(a): a DSS with Repetition code

Figure 1.3(b): DSS with $(4, 2)$ MDS code

Figure 1.3(c): DSS with Regenerating code

Figure 1.3: Examples of various DSSs on 4 nodes.

calculated. From the fundamental bound trade-off curve between the cost of node storage and the cost of repair bandwidth are obtained. For such heterogeneous DSS, FR codes with general settings on parameters are called Generalized Fractional Repetition (GFR) codes, and different bounds and properties of the GFR code are analyzed. A brief contribution of the thesis is following.

Chapter 2 includes the preliminaries for sequences, DSSs, information flow graph, GFR codes, and hypergraphs. In Chapter 3, the literature survey is included for regenerating codes and FR codes.

In Chapter 4, we have calculated the *min-cut* bound for the proposed heterogeneous DSS. For such heterogeneous DSS, we have established a bi-objective optimization linear programming problem subject to the *min-cut* bound. The solutions to the LP problem such as a trade-off curve between the system storage and the repair cost are plotted. In a heterogeneous DSS, the system storage cost and the system repair cost are average costs to store and repair unit information data on a node respectively. We have plotted a trade-off curve and compared it with the trade-off curves for the heterogeneous DSS given in [122] and the homogeneous DSS (DSS with the symmetric parameters) as investigated in [119]. Some specific cases are investigated for the established bi-objective optimization problem for various heterogeneous DSSs. The computational complexity to calculate parameters, for the fundamental bound achieving codes, is very high. So, a particular case by choosing constant repair traffic and reconstruction degree are considered. Further,

we have calculated relations on parameters which achieve the fundamental bound.

In Chapter 5, GFR codes which are more general and realistic case for practical scenario are considered. The parameters of GFR codes such as reconstruction degree, node storage capacity, repair degree, repair bandwidth and packet replication factor are not uniform. A GFR code construction based on sequences is given in Chapter 5. In particular, construction of Flower code is studied in detail. The conditions for universally good GFR code (Flower code) are also obtained in the same chapter.

In Chapter 6, it is shown that a GFR code corresponds to a hypergraph, where the GFR code may have some asymmetric parameters. Bounds on specific GFR codes which are corresponding to some specific hypergraphs are also obtained. For given parameters of a GFR code with some specific properties, if the respective bound is not satisfied, then the GFR code will not exist. It is observed that a GFR code corresponding to a linear hypergraph is universally good. Bounds on parameters of GFR codes with additional system properties are obtained in the same chapter.

In Chapter 7, the code rate is analyzed for a GFR code. The *GFR code rate* for a GFR code is the ratio of file size to the total number of encoded packets stored in the GFR code. Bounds on the GFR code rate, the growth of the GFR code rate and the DSS code rate are obtained in this thesis. It is also shown that the GFR code with GFR code rate more than the DSS code rate, does not exist. Again it has been proved that the reliable GFR code with GFR code rate more than 0.5 does not exist. A reliable GFR code with the maximum code rate 0.5 has the lowest tolerance factor of 1. Chapter 8 concludes the thesis with future work.

CHAPTER 2

Preliminaries

Mathematicians have tried in vain to this day to discover some order in the sequence of prime numbers, and we have reason to believe that it is a mystery into which the human mind will never penetrate. -*Leonhard Euler* [1]

The preliminaries for sequences, heterogeneous DSSs, GFR codes, and Hypergraphs are discussed in this chapter.

2.1 Sequences

A *sequence* is a one-dimensional array defined on a set of q symbols, called alphabet. If the length of the array is finite, then the sequence is called a *finite sequence*. For an alphabet A, a finite sequence of length ℓ $(\in \mathbb{N})$ is denoted by $\vec{x} = (x_1\ x_2 \dots x_\ell)$, where $x_i \in A$ for $i = 1, 2, \dots, \ell$. If the alphabet is $A = \{0, 1\}$ then the sequence is called a *binary sequence*. For simplicity, a finite binary sequence of length ℓ $(\in \mathbb{N})$ is denoted by $\mathbf{x} = x_1 x_2 \dots x_\ell$, where $x_i \in \{0, 1\}$ for $i = 1, 2, \dots, \ell$. The *weight* of a finite binary sequence $\mathbf{x}$ is the sum of all the terms of the sequence and denoted by $w_{\mathbf{x}}$. The *weight* of the first s terms of a binary sequence $\mathbf{x}$ is the sum of the first s terms of the sequence and denoted by $w_{\mathbf{x}}(s)$. For example, if $\mathbf{x} = 101101$, then $w_{\mathbf{x}} = 4$ and $w_{\mathbf{x}}(3) = 2$. A sequence $\mathbf{x}$ of length ℓ is called *periodic sequence*, if there exists some positive integer τ such that $\mathbf{x}_r = \mathbf{x}_{r+\tau}$, for $\tau | \ell$ and $r = 1, 2, \dots, \ell - \tau$. For the periodic sequence, the parameter τ is called *period*. For example, the

sequence $\mathbf{x} = 101110111011$ is a periodic sequence with the period $\tau = 4$. For two sequences $\mathbf{x}$ and $\mathbf{y}$, the concatenation is denoted $\mathbf{xy}$. For a positive integer s, the sequence $\mathbf{x}^s$ denotes the concatenation of s copies of the sequence $\mathbf{x}$. For example, $(10)^2(1011)^3 0^4 = 10101011101110110000$.

2.2 Distributed Storage Systems

In a DSS, a data file is encoded in a certain number of packets of the same size and those packets are stored on n distinct nodes. To retrieve the complete data file information from the DSS, a data collector has to download packets from any k nodes. For a reliable system, a failed node has to be repaired in the DSS. For the repair, a failed node is replaced by a new active node with the same information. For the new node, the packets are downloaded from each node (called *helper node*) of a set of d active nodes. The set of helper nodes is called *surviving set* for the node. Formally, an (n, k, d) DSS is a storage system with n nodes, reconstruction degree k and repair degree d such that

- each node contains a fraction of data file information,
- any data collector can reconstruct the complete file information by downloading packets from any k $(< n)$ nodes, and
- a failed node is repaired by some d helper nodes.

For a DSS, if all the parameters (such as reconstruction degree, repair degree of each node, repair bandwidth of each node, node storage capacity) are individual symmetric or uniform, then the DSS is called *homogeneous DSS*. The $(4, 2, 2)$ DSS as given in Chapter 1 (Figure 1.3(c)) is an example of homogeneous DSS.

2.2.1 Homogeneous DSSs

Dimakis *et al.* introduced regenerating codes in the seminal work [22, 23]. Let a file be divided into M distinct packets, and the packets are stored on a (n, k, d) DSS, where the node storage capacity of each node is α. In the DSS, a failed node

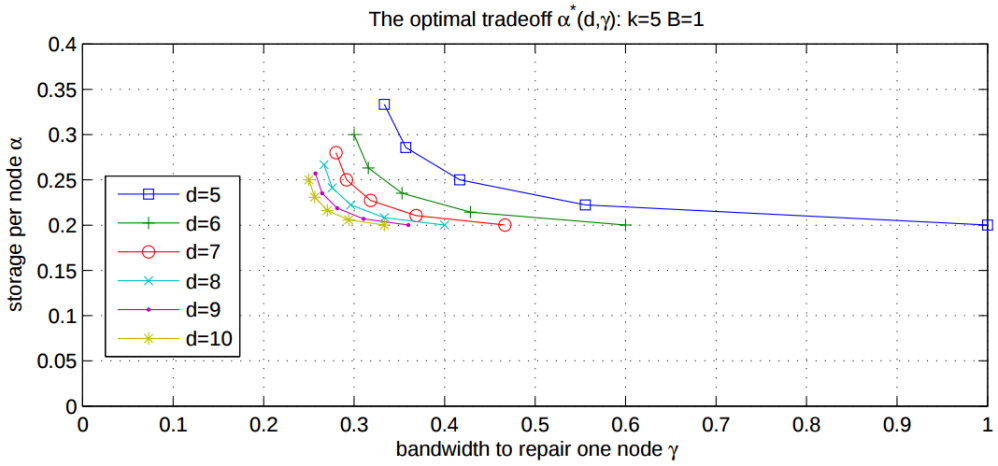

Figure 2.1: For $d = 5, 6, \ldots 10$, $k = 5$ and $M = 1$, optimal trade-off curves (as plotted in [119]) between storage and repair bandwidth.

is repaired by d helper nodes, and β packets are downloaded from each helper node for the repair process. Therefore, the repair bandwidth (the total number of downloaded packets for a node failure) is $\gamma = d\beta$. The parameters of the DSS must satisfy

$$M \leq \sum_{i=0}^{k-1} \min \left\{ \alpha, \left(1 - \frac{i}{d}\right) \gamma \right\}, \tag{2.1}$$

and the trade-off between the node storage capacity α and the repair bandwidth γ is plotted using Inequality (2.1) [23, 119]. For given $k = 5$, $M = 1$ and $d = 5, 6, \ldots 10$, the trade-off between storage capacity and repair bandwidth is simulated in [119] (see Figure 2.1 for the simulated trade-off). Note that the trade-off is asymptotic. For the trade-off curve, functional repair is considered, where the repaired node contains the function of the lost information.. Formally, the regenerating codes are defined as follows.

Definition 2.1. *For a $[(n, k, d), (\alpha, \beta, M)]$ DSS, regenerating codes are the class of codes such that*

- *the data file can be reconstructed by downloading packets from any k $(< n)$ nodes, and*
- *a failed node can be repaired by replacing it with a new node, where the packets for the new node is downloaded from any d existing nodes by downloading β packets from each.*

In the trade-off curve, by minimizing both parameters in different order, Minimum Bandwidth Regenerating (MBR) codes and Minimum Storage Regenerating (MSR) codes are obtained [23]. For the MSR code, the node storage capacity α_{MSR} and the repair bandwidth γ_{MSR} are given by the pair [23, 119]

$$(\alpha_{MSR}, \gamma_{MSR}) = \left(\frac{M}{k}, \frac{Md}{k(d-k+1)} \right). \tag{2.2}$$

MSR codes are constructed in [92, 88, 65]. For the MBR code, the the node storage capacity α_{MBR} and the repair bandwidth γ_{MBR} are given by the pair [23, 119]

$$(\alpha_{MBR}, \gamma_{MBR}) = \left(\frac{2Md}{k(2d-k+1)}, \frac{2Md}{k(2d-k+1)} \right). \tag{2.3}$$

MBR codes are constructed in [26, 88, 58, 88].

A DSS which is not homogeneous is the *heterogeneous DSS*. Heterogeneous DSSs are closer to the real-world scenarios, where the characterization of all storage nodes in various aspects are not necessarily uniform due to geographical environment and storage devices cost etc. Recently, heterogeneous distributed storage have been studied for various applications in storage systems such as hybrid storage systems [59], video-on-demand systems [124] and heterogeneous wireless networks [35].

2.2.2 Heterogeneous DSSs

Let a file be divided into M distinct packets, where each packet has the same length and is defined over the field $\mathbb{F}_q$. The packets are encoded and are stored on n distinct nodes U_i ($i = 1, 2, \ldots, n$) such that a data collector can recover the file by downloading packets from any k ($< n$) nodes. Let the α_i number of packets be stored on the node U_i for each $i = 1, 2, \ldots, n$. For a failed node, packets are downloaded from some d_i active nodes. Note that packets downloaded from two nodes in a surviving set may not be same.

An example of such heterogeneous DSS is considered in Figure 2.3. In this system, a file is divided into $M = 4$ massage information packets x_1, x_2, x_3 and x_4 on the field $\mathbb{F}_q$ with q symbols. The massage information packets are encoded

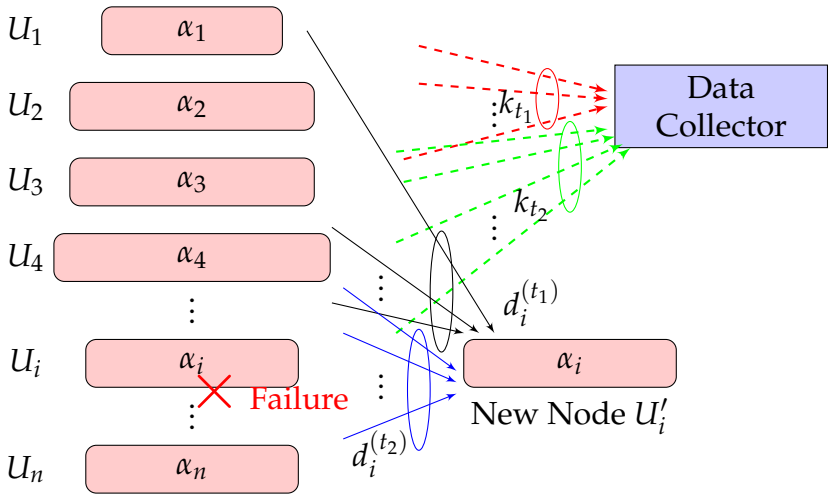

Figure 2.2: A model of heterogeneous DSS.

into 11 packets by taking linear combinations of massages information packets as $y_1 = x_1$, $y_2 = x_2$, $y_3 = x_3$, $y_4 = x_1 + x_2$, $y_5 = x_4$, $y_6 = x_1 + x_2$, $y_7 = x_1$, $y_8 = x_3$, $y_9 = x_2 + x_4$, $y_{10} = x_2$ and $y_{11} = x_1 + x_4$. The encoded packets y_m $(m = 1, 2, \ldots, 11)$ are distributed on the $n = 5$ nodes such that packets y_1 and y_2 are stored on node U_1, packets y_3 and y_4 are distributed on node U_2, packets y_5 and y_6 are on node U_3, packets y_7, y_8 and y_9 are on node U_4 and remaining two packets are on node U_5. Clearly the node storage capacity $\alpha_i = 2$ $(i = 1, 2, 3, 5)$ and $\alpha_4 = 3$. A data collector can download the complete file by connecting $k = 3$ or fewer nodes. In this example, if node U_5 fails, then it can be repaired by downloading packets y_7 and y_9 from node U_4. Since the recovered packets are function of lost packets so, it is functional repair. On the other hand, node U_5 can be repaired exactly by downloading packets y_1, y_2 and y_5 from nodes U_1 and U_3 and solving $y_{10} = y_2$ and $y_{11} = y_1 + y_5$.

In this thesis, we focus our attention to the heterogeneous DSS, the more general settings on the homogeneous DSS, for which at different time instants, reconstruction degrees are not uniform and repair degree are also not same. At any time t, one can define a *reconstruction set* $\mathcal{A}$ as a collection of the nodes having sufficient packets collectively to reconstruct the file *i.e.*,

$$\mathcal{A} = \{U_i : |\cup_{i \in I} U_i| \geq M, I \subset \{1, 2, \ldots, n\}\}.$$

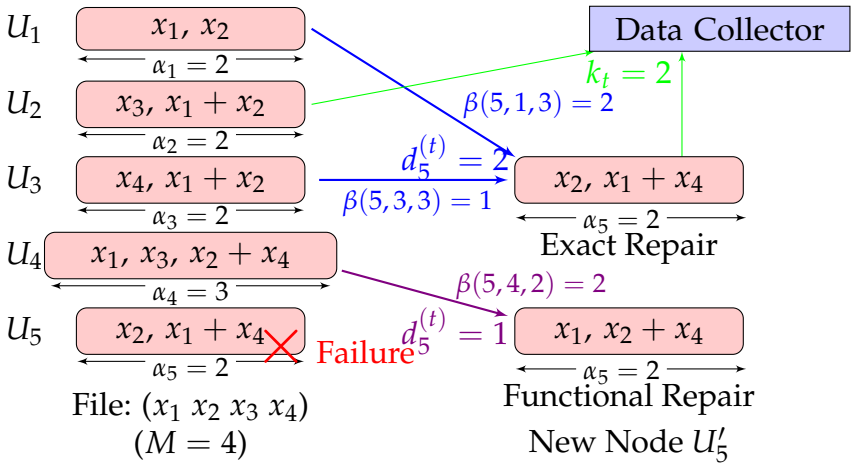

Figure 2.3: An example of a $(5,3)$ DSS.

Clearly $|I| = |\mathcal{A}| = k_t$ the reconstruction degree at time t, and intersection of any two reconstruction set may be non-empty. Define $\mathscr{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_t, \ldots\}$ as a set of all reconstruction sets. Note that the set $\mathscr{A}$ will be finite if all reconstruction sets $\mathcal{A} \in \mathscr{A}$ are distinct. Therefore, the vector $(k_1\, k_2 \ldots k_{|\mathscr{A}|})$ is called reconstruction vector and denoted by $\vec{k}$. For example as considered in Figure 2.3, $\mathscr{A} = \{\mathcal{A}_i : i = 1, 2, \ldots, 7\}$ and $\vec{k} = (3\ 3\ 2\ 2\ 2\ 2\ 2)$, where reconstruction set $\mathcal{A}_i$ is given in the Table 2.1 for $i = 1, 2, \ldots, 7$.

Table 2.1: Reconstruction sets for the DSS as considered in Figure 2.3.

Reconstruction set $\mathcal{A}$	Reconstruction degree $\lvert\mathcal{A}\rvert$
$\mathcal{A}_1 = \{U_1, U_2, U_3\}$	3
$\mathcal{A}_2 = \{U_1, U_2, U_5\}$	3
$\mathcal{A}_3 = \{U_1, U_4\}$	2
$\mathcal{A}_4 = \{U_2, U_4\}$	2
$\mathcal{A}_5 = \{U_2, U_5\}$	2
$\mathcal{A}_6 = \{U_3, U_4\}$	2
$\mathcal{A}_7 = \{U_4, U_5\}$	2

In the heterogeneous DSS, at time t, if a node U_i $(i = 1, 2, \ldots, n)$ fails then certain active nodes called helper nodes, download packets and generate a new node say U_i'. The downloaded packets for the new node, have sufficient information which is lost in the failed node. The new node U_i' takes place of the failed node U_i and the system is repaired. In particular, a set of those helper nodes is called *surviving set* for the failed node U_i and is denoted by $S(i)$. For a node U_i, at the

time instant t, the surviving set is

$$S(i) := \{U_j : j \in J \subset \{1, 2, \ldots, n\} \backslash \{i\}\}.$$

For a node U_i, if τ_i distinct surviving sets exist then the surviving sets are denoted $S_\ell(i)$ for $\ell = 1, 2, \ldots, \tau_i$. If a failed node U_i repaired by nodes of surviving set $S(i)$ then the repair degree at the time t, is $d_i^{(t)} = |J| = |S(i)|$. For the node U_i, $d_i := \max\{|S_\ell(i)| : \ell = 1, 2, \ldots, \tau_i\}$. For a given DSS, we denote $\vec{d} := (d_1\ d_2\ \ldots d_n)$ as repair degree vector. Surviving sets are listed in Table 2.2 for the heterogeneous DSS considered in Figure 2.3. In this example, one can see that if a node U_4 fails, then it can be repaired by connecting nodes U_2 and U_3 or nodes U_2 and U_5. Hence surviving sets for the node U_4 are $S_1(4)$ and $S_2(4)$. In Table 2.2, for a given i $(i = 1, 2, \ldots, 5)$, $|S(i)|$ is identical for all $\ell = 1, 2, \ldots, \tau_i$. In general, it may not be accurate. Also note that, in the table, we have chosen those surviving sets which are not the superset of some other surviving set for the same node failure. In particular, the condition ensures the active participation of each node of an arbitrary surviving set during failed node repair process.

Table 2.2: Surviving sets for nodes in the DSS as considered in Figure 2.3.

Nodes U_i	Surviving sets $S(i)$	Number of surviving sets τ_i
U_1	$S_1(1) = \{U_2, U_4\}$, $S_2(1) = \{U_2, U_5\}$, $S_3(1) = \{U_3, U_4\}$, $S_4(1) = \{U_3, U_5\}$, $S_5(1) = \{U_4, U_5\}$.	5
U_2	$S_1(2) = \{U_1, U_4\}$, $S_2(2) = \{U_3, U_4\}$, $S_3(2) = \{U_4, U_5\}$.	3
U_3	$S_1(3) = \{U_4\}$, $S_2(3) = \{U_5\}$.	2
U_4	$S_1(4) = \{U_2, U_3\}$, $S_2(4) = \{U_2, U_5\}$.	2
U_5	$S_1(5) = \{U_3\}$, $S_2(5) = \{U_4\}$.	2

For a failed node U_i, if system is repaired by nodes of specific surviving set $S_\ell(i)$ then the number of information packets downloaded by node $U_j \in S_\ell(i)$ will

be denoted by $\beta(i,j,\ell) > 0$. For example, if node U_4 fails in DSS as shown in Figure 2.3 then all the two packets from node $U_5 \in S_2(4)$ and packet $y_3 = x_3$ from node $U_2 \in S_2(4)$ is downloaded to repair the failed node U_4. Then $\beta(4,5,2) = 2$ and $\beta(4,2,2) = 1$.

If a failed node U_i ($i \in \{1,2,\ldots,n\}$) is repaired by nodes of surviving set $S_\ell(i)$ then *repair bandwidth* (denoted by $\gamma(i,\ell)$) for the node U_i, is the total number of packets downloaded by every nodes of the surviving set $S_\ell(i)$. Mathematically,

$$\gamma(i,\ell) = \sum_{\substack{j \\ U_j \in S_\ell(i)}} \beta(i,j,\ell). \tag{2.4}$$

In the heterogeneous DSS as shown in the Figure 2.3, if node U_5 fails and it is repaired by nodes of surviving set $S_3(5) = \{U_1, U_3\}$ (not considered in Table 2.2 since $S_1(5) \subsetneqq S_3(5)$) then $\gamma(5,3) = \beta(5,1,3) + \beta(5,3,3) = 2 + 1 = 3$ units.

The DSS with the general setting of parameters can be called *Generalized Distributed Storage System* (Genralized DSS) with the parameters $(n, \vec{k}, \vec{d})$. In this thesis, at time instant t, single node failure is considered because simultaneously multi-node failures can be assumed as a sequence of single node failures in a small time interval.

In a homogeneous DSS, Wu *et al.* [119] solved an optimization problem with the constraint of *min-cut* bound between the parameters and they plotted the trade-off curve between node storage capacity α and node repair bandwidth $d\beta$. The fundamental bound as given in Inequality 2.1 is obtained by the information flow graph for the homogeneous DSS [119]. Similarly, one can plot the trade-off curve for the heterogeneous DSS.

In this chapter, we are defining some sequences of distinct nodes and corresponding surviving sets for our model to analyze information flow. The definitions are as follows.

Definition 2.2. *(Node Sequence): For a reconstruction set $\mathcal{A}$, node sequence $\vec{u} = (U_1\ U_2\ \ldots\ U_{|\mathcal{A}|})$ is a sequence of nodes such that each node in the set $\mathcal{A}$ has appeared exactly ones in the sequence.*

Definition 2.3. *(Node Sequence Set): For a reconstruction set $\mathcal{A}$, a set of all node se-*

quences is called node sequence set and is denoted by $\mathscr{A}(\mathcal{A})$. Clearly $|\mathscr{A}(\mathcal{A})| = |\mathcal{A}|!$.

For the DSS as considered in the Figure 2.3, node sequences and node sequence sets, defined on various reconstruction sets, are listed in the Table 2.3.

Table 2.3: Node sequences for the DSS as considered in Figure 2.3.

Reconstruction set $\mathcal{A}$	Node sequences $\vec{u}$	Node sequence set $\mathscr{A}(\mathcal{A})$
$\mathcal{A}_1 = \{U_1, U_2, U_3\}$	$\vec{u}_1 = (U_1\ U_2\ U_3)$ $\vec{u}_2 = (U_1\ U_3\ U_2)$ $\vec{u}_3 = (U_2\ U_1\ U_3)$ $\vec{u}_4 = (U_2\ U_3\ U_1)$ $\vec{u}_5 = (U_3\ U_1\ U_2)$ $\vec{u}_6 = (U_3\ U_2\ U_1)$	$\mathscr{A}(\mathcal{A}_1) = \{\vec{u}_1, \vec{u}_2, \vec{u}_3, \vec{u}_4, \vec{u}_5, \vec{u}_6\}$
$\mathcal{A}_2 = \{U_1, U_2, U_5\}$	$\vec{u}_1 = (U_1\ U_2\ U_5)$ $\vec{u}_2 = (U_1\ U_5\ U_2)$ $\vec{u}_3 = (U_2\ U_1\ U_5)$ $\vec{u}_4 = (U_2\ U_5\ U_1)$ $\vec{u}_5 = (U_5\ U_1\ U_2)$ $\vec{u}_6 = (U_5\ U_2\ U_1)$	$\mathscr{A}(\mathcal{A}_2) = \{\vec{u}_1, \vec{u}_2, \vec{u}_3, \vec{u}_4, \vec{u}_5, \vec{u}_6\}$
$\mathcal{A}_3 = \{U_1, U_4\}$	$\vec{u}_1 = (U_1\ U_4)$ $\vec{u}_2 = (U_4\ U_1)$	$\mathscr{A}(\mathcal{A}_3) = \{\vec{u}_1, \vec{u}_2\}$
$\mathcal{A}_4 = \{U_2, U_4\}$	$\vec{u}_1 = (U_2\ U_4)$ $\vec{u}_2 = (U_4\ U_2)$	$\mathscr{A}(\mathcal{A}_4) = \{\vec{u}_1, \vec{u}_2\}$
$\mathcal{A}_5 = \{U_2, U_5\}$	$\vec{u}_1 = (U_2\ U_5)$ $\vec{u}_2 = (U_5\ U_2)$	$\mathscr{A}(\mathcal{A}_5) = \{\vec{u}_1, \vec{u}_2\}$
$\mathcal{A}_6 = \{U_3, U_4\}$	$\vec{u}_1 = (U_3\ U_4)$ $\vec{u}_2 = (U_4\ U_3)$	$\mathscr{A}(\mathcal{A}_6) = \{\vec{u}_1, \vec{u}_2\}$
$\mathcal{A}_7 = \{U_4, U_5\}$	$\vec{u}_1 = (U_4\ U_5)$ $\vec{u}_2 = (U_5\ U_4)$	$\mathscr{A}(\mathcal{A}_7) = \{\vec{u}_1, \vec{u}_2\}$

Definition 2.4. *(Surviving Sequence): For a node sequence $\vec{u} = (U_{\lambda_1} U_{\lambda_2} \dots U_{\lambda_{|\mathcal{A}|}})$ on a reconstruction set $\mathcal{A} \in \mathscr{A}$, the corresponding sequences of surviving sets is $\vec{s} = \left(S_{\ell_1}(\lambda_1)\ S_{\ell_2}(\lambda_2) \dots S_{\ell_{|\mathcal{A}|}}(\lambda_{|\mathcal{A}|})\right)$, where $S_{\ell_i}(\lambda_i)$ is the surviving set of the node U_{λ_i} and $\ell_i \in \{1, 2, \dots, \tau_{\lambda_i}\}$.*

Definition 2.5. *(Surviving Sequences Set): For a node sequence $\vec{u}$ on a reconstruction set $\mathcal{A}$, the set of all distinct surviving sequences is called the surviving sequence set and is denoted by $\mathscr{S}(\vec{u})$. Clearly $|\mathscr{S}(\vec{u})| = \left(\prod_{i=1}^{|\mathcal{A}|} \tau_{\lambda_i}\right)$.*

For the DSS as given in the Figure 2.3, surviving sequences and node sequence set for the node sequence $\vec{u} = (U_1\ U_4)$ are listed in the Table 2.4.

In [122], Quan *et al*. have given trade-off curve between system storage cost and system repair cost for a heterogeneous DSS, having uniform reconstruction degree. Similarly one can give a trade-off curve between system storage cost and system repair cost for a heterogeneous DSS model considered in this chapter. For the heterogeneous DSS, we define system storage cost, cost of node storage and system repair cost as follows.

Table 2.4: Surviving sequences for the DSS as considered in Figure 2.3.

Node sequence $\vec{u}$	Surviving sequences $\vec{s}$	Surviving sequence set $\mathscr{S}(\vec{u})$
$\vec{u} = (U_1\ U_4)$	$\vec{s}_1 = (S_1(1)\ S_1(4))$ $\vec{s}_2 = (S_1(1)\ S_2(4))$ $\vec{s}_3 = (S_2(1)\ S_1(4))$ $\vec{s}_4 = (S_2(1)\ S_2(4))$ $\vec{s}_5 = (S_3(1)\ S_1(4))$ $\vec{s}_6 = (S_3(1)\ S_2(4))$ $\vec{s}_7 = (S_4(1)\ S_1(4))$ $\vec{s}_8 = (S_4(1)\ S_2(4))$ $\vec{s}_9 = (S_5(1)\ S_1(4))$ $\vec{s}_{10} = (S_5(1)\ S_2(4))$	$\mathscr{S}(\vec{u}) = \{\vec{s}_i : i = 1, 2, \ldots, 10\}$

Definition 2.6. *(System Storage Cost): Total amount of cost $C_s(\vec{\alpha})$ to store unit data in heterogeneous DSS$(n, \vec{k}, \vec{d})$ is called system storage cost, where storage amount vector $\vec{\alpha} := (\alpha_1\ \alpha_2\ \ldots \alpha_n)$, storage cost vector $\vec{c} := (c_1\ c_2 \ldots c_n)$, α_i is storage capacity of node U_i and c_i is the cost to store unit information data in node U_i for $i = 1, 2, \ldots, n$. Clearly*

$$C_s(\vec{\alpha}) = \frac{1}{M} \sum_{j=1}^{n} c_j \alpha_j$$

System storage cost $C_s(\vec{\alpha})$ is 68 cost units for the example considered in Figure 2.3 with $\vec{c} = (100\ 10\ 10\ 10\ 1)$, where the storage cost for the node U_1 is $c_1 = 100$, for node U_5 is $c_5 = 1$ and for node $U_i = 10$ $(i = 2, 3, 4)$.

Definition 2.7. *(Node Repair Cost): The average amount of cost to repair a node U_i $(i \in \{1, 2, \ldots, n\})$ in heterogeneous DSS$(n, \vec{k}, \vec{d})$, is called node repair cost $r(\beta_i)$ associated*

with repair cost vector $\vec{r} := (r_1\ r_2 \ldots r_n)$ *s.t.*

$$r(\beta_i) = \frac{1}{M\tau_i} \sum_{\ell=1}^{\tau_i} \sum_{\substack{j \\ U_j \in S_\ell(i)}} r_j \beta(i, j, \ell), \tag{2.5}$$

where r_j *is cost to download unit amount of data from node* U_j *during the repair process. Clearly node repair vector* $r(\vec{\beta}) := (r(\beta_1)\ r(\beta_2)\ \ldots\ r(\beta_n))$.

In the example considered in Figure 2.3, if $\vec{r} = (10\ 1\ 1\ 1\ 1)$ then node repair cost vector $r(\vec{\beta}) = (r(\beta_1)\ r(\beta_2)\ r(\beta_3)\ r(\beta_4)\ r(\beta_5))$ = $(\frac{1}{2}\ \frac{4}{3}\ \frac{1}{2}\ \frac{3}{4}\ \frac{1}{2})$.

Definition 2.8. *(System Repair Cost): System repair cost is total amount of cost to repair all nodes in heterogeneous DSS*$(n, \vec{k}, \vec{d})$ *and is denoted by* $C_r(\vec{\beta})$. *Mathematically*

$$C_r(\vec{\beta}) = \sum_{j=1}^{n} r(\beta_j).$$

Clearly $C_r(\vec{\beta}) = \frac{43}{12}$ unit for $\vec{r} = (10\ 1\ 1\ 1\ 1)$ in the example considered in Figure 2.3.

2.3 Generalized Fractional Repetition Codes

The MSR codes are more efficient for archival purpose and the MBR codes are efficient for Internet applications [24, 26]. To optimize disk I/O, a class of MBR codes are known as *Distributed Replication-based Exact Simple Storage (DRESS) codes* were introduced and studied by researchers [26, 96, 89, 134]. The repair mechanism for these codes are known as *repair by transfer*, *uncoded repair* or *table based repair* (to repair a failed node, packets are copied from helper nodes and are forwarded to the newcomer node without any computation). The repair reduces the computation cost. As shown in Figure 2.4, the DRESS code is a two-layer code, consists of the inner code, called *Fractional Repetition* (FR) code along with the MDS code as outer code [26]. For a positive integer ρ, Fractional Repetition code is a collection of n subsets of a set $\{P_j : j = 1, 2, \ldots, \theta\}$ such that each subset has the same cardinality α and ρ copies of each packet P_j is in the collection [26].

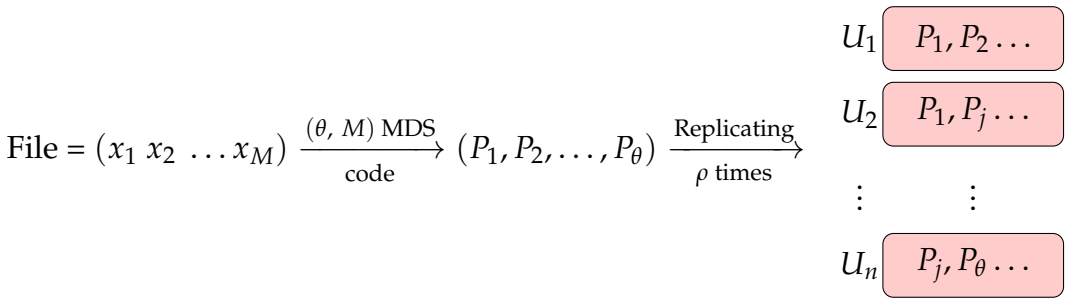

Figure 2.4: DRESS Code

For the more general setting on parameters of FR code, we define Generalized fractional Repetition (GFR) code on a heterogeneous DSS $(n, \vec{k}, \vec{d})$. Let a file be divided into M distinct information packets, where information packet symbols are associated with a finite field $\mathbb{F}_q$. By using (θ, M) MDS code on the information packets, the M information packets are encoded into θ distinct packets P_1, P_2, …, P_θ. Because of the property of the MDS code, the complete file information can be extracted from any M packets. Each packet P_j $(1 \leq j \leq \theta)$ is replicated ρ_j times in the system. All the packets are distributed on n distinct nodes U_1, U_2, …, U_n such that each node U_i $(i = 1, 2, \ldots, n)$ contains α_i packets. It is easy to observe that the code is the generalization of the FR code on multiple parameters. For the GFR code, the maximum node storage capacity and the maximum replication factor are given by $\alpha = \max\{\alpha_i : i = 1, 2, \ldots, n\}$ and $\rho = \max\{\rho_j : j = 1, 2, \ldots, \theta\}$ respectively. The GFR code is denoted by $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, where $\vec{\alpha} = (\alpha_1\ \alpha_2 \ldots \alpha_n)$ and $\vec{\rho} = (\rho_1\ \rho_2 \ldots \rho_\theta)$.

Example 2.1. *Let a file be split into* 5 $(= M)$ *information packets and encoded into* 6 $(= \theta)$ *distinct packets* P_1, P_2, P_3, P_4, P_5 *and* P_6 *using a* $(6, 5)$ *MDS code. The* 13 *replicas of the* 6 $(= \theta)$ *distinct packets are stored in the* 5 *distinct nodes* U_1, U_2, U_3, U_4 *and* U_5 *such that* $U_1 = \{P_1, P_2, P_3, P_4\}$, $U_2 = \{P_1, P_5, P_6\}$, $U_3 = \{P_2, P_5\}$, $U_4 = \{P_3, P_6\}$ *and* $U_5 = \{P_4, P_6\}$. *The node packet distribution of the GFR code is illustrated in Table 2.5. Hence,* $\alpha_1 = |U_1| = 4$, $\alpha_2 = |U_2| = 3$, $\alpha_i = |U_i| = 2$ $(i = 3, 4, 5)$. *For the GFR code, the node storage vector* $\vec{\alpha} = (4\ 3\ 2\ 2\ 2)$ *and the replication vector* $\vec{\rho} = (2\ 2\ 2\ 2\ 2\ 3)$. *In the GFR code,* $\rho = \max\{\rho_j : j = 1, 2, \ldots, 6\} = 3$ *and* $\alpha = \max\{\alpha_i : i = 1, 2, \ldots, 5\} = 4$. *Hence, the GFR code is denoted by* $\mathscr{C} : (n = 5, \theta = 6, \vec{\alpha} = (4\ 3\ 2\ 2\ 2), \vec{\rho} = (2\ 2\ 2\ 2\ 2\ 3))$.

For a GFR code, if a data collector connects any k nodes and downloads M

Table 2.5: The node packet distribution of the GFR code as considered in Example 2.1.

U_1	P_1, P_2, P_3, P_4
U_2	P_1, P_5, P_6
U_3	P_2, P_5
U_4	P_3, P_6
U_5	P_4, P_6

distinct packets, then the file can be retrieved by using the MDS property, *i.e.*, message information can be obtained from any M packets out of θ packets. The parameter k is known as the reconstruction degree of the GFR code. Note that the reconstruction degree of the GFR code (Example 2.1) is $k = 3$. In a GFR code, a failed node U_i can be repaired by replacing a new node U_i', where both the nodes U_i and U_i' carry the same information. The repair degree of a failed node U_i is the cardinality of the set of the helper nodes. For a node U_i in a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, more then one such helper node sets exist with different repair degree. For the node U_i, the maximum repair degree is d_i and $d = \max\{d_i : i = 1, 2, \ldots, n\}$. In the GFR code (Example 2.1), if node U_2 fails then the new node U_2' can be created by downloading packets from nodes of set $\{U_1, U_3, U_4\}$ or set $\{U_1, U_3, U_5\}$. Hence, $d_2 = 3$ and $d = \max\{d_i : i = 1, 2, 3, 4, 5\} = 4$. Now, one can define a GFR code formally as follows.

Definition 2.9. *(Generalized Fractional Repetition Code): On a Distributed Storage System with n nodes (denoted by U_i for $i = 1, 2, \ldots, n$) and θ packets (denoted by P_j for $j = 1, 2, \ldots, \theta$), one can define a GFR code as a collection $\mathscr{C}$ of n subsets U_i ($i = 1, 2, \ldots, n$) of a set $\{P_j : j = 1, 2, \ldots, \theta\}$, such that an arbitrary packet P_j is distributed on ρ_j ($\in \mathbb{N}$) distinct subsets in the collection $\mathscr{C}$. The GFR code is denoted by $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, where $|U_i| = \alpha_i$, $\vec{\alpha} = (\alpha_1\ \alpha_2 \ldots \alpha_n)$ and $\vec{\rho} = (\rho_1\ \rho_2 \ldots \rho_\theta)$.*

For identical replication factor *i.e.*, $\rho_j = \rho$ for $j = 1, 2, \ldots, \theta$, the notation of GFR code can be simplified by $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$. Similarly, for $\alpha_i = \alpha$ ($i = 1, 2, \ldots, n$), the GFR code can be denoted by $\mathscr{C} : (n, \theta, \alpha, \vec{\rho})$. Again, for a GFR code with $\alpha_i = \alpha$ ($i = 1, 2, \ldots, n$) and $\rho_j = \rho$ ($j = 1, 2, \ldots, \theta$), can be represented by $\mathscr{C} : (n, \theta, \alpha, \rho)$ (the FR code as introduced in [26]).

One can represent a GFR code with matrices called *Node Adjacency Matrix* and

Node Packet Distribution incidence Matrix. For a GFR code, both the matrices are defined as follows.

Definition 2.10. *(Node Adjacency Matrix): For a GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$*, the node adjacency matrix* $A = [a_{i,j}]_{n\times n}$ *is a square matrix over* $\mathbb{N} \cup \{0\}$ *with size* $n \times n$*, where the matrix element*

$$a_{i,j} = \begin{cases} |U_i \cap U_j|, & \text{if } i \neq j; \\ 0, & \text{if } i = j. \end{cases} \tag{2.6}$$

Note that the node adjacency matrix is symmetric.

The node adjacency matrix of the GFR code $\mathscr{C} : (5, 6, (4\ 3\ 2\ 2\ 2), (2\ 2\ 2\ 2\ 2\ 3))$ (Example 2.1) is

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

Definition 2.11. *(Node Packet Distribution Incidence Matrix): For a GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$*, a matrix* $B = [b_{i,j}]_{n\times\theta}$ *over* $\{0, 1\}$ *is called a node packet distribution incidence matrix (NPDI Matrix) if*

$$b_{i,j} = \begin{cases} 1 & : \text{if the packet } P_j \text{ is stored in the node } U_i; \\ 0 & : \text{if the packet } P_j \text{ is not stored in the node } U_i. \end{cases}$$

The NPDI matrix of the GFR code $\mathscr{C} : (5, 6, (4\ 3\ 2\ 2\ 2), (2\ 2\ 2\ 2\ 2\ 3))$ (Example 2.1) is

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

Some properties of the NPDI Matrix for a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ are observed as.

1. For a GFR code, NPDI Matrix exists, and it is unique up to the permutation.

2. In the NPDI Matrix of a GFR code, there is no column with entry sum zero.

3. In the NPDI Matrix of a GFR code, there is no row with entry sum zero.

4. The entry sum of j^{th} column in the NPDI Matrix of a GFR code is ρ_j (replication factor of packet P_j, for each $j = 1, 2, \ldots \theta$).

5. The entry sum of i^{th} row in the NPDI Matrix of a GFR code is α_i (node storage capacity of node U_i, for each $i = 1, 2, \ldots n$).

6. The column support of j^{th} column for the NPDI Matrix of a GFR code is the set of indexes of those nodes on which packet P_j is distributed.

7. The row support of i^{th} row for NPDI Matrix of a GFR code is the set of indexes of those packets which are distributed on node U_i.

Hence, one can observe the following two remarks.

Remark 2.1. *For each binary matrix $B_{n\times\theta}$ with nonzero entry sum of each row and each column, a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ exists.*

Remark 2.2. *For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ with the NPDI Matrix B, the adjacency matrix $A = BB^t$, where B^t is the transpose of the matrix B.*

In a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, if $M(k)$ is the maximum number of distinct packets that guarantee to deliver to any user connected with any k nodes of the GFR code then

$$M(k) := \min\left\{\left|\bigcup_{i\in I} U_i\right| : |I| = k, I \subset \{1, 2, \ldots, n\}\right\}. \tag{2.7}$$

Note that $M \leq M(k)$. For the GFR code (as given in Example 2.1), one can find that $M(3) = 5$.

Consider a GFR code $\mathscr{C} : (n, \theta, \alpha, \rho)$ with $\alpha_i = \alpha$ for each $i = 1, 2, \ldots, n$ and $\rho_j = \rho$ for each $j = 1, 2, \ldots, \theta$. The GFR code is called *universally good* [26, 105, 128] if, for $k = 1, 2, \ldots \alpha$, the maximum file size $M(k)$ achieves the MBR capacity [22] *i.e.*,

$$M(k) \geq k\alpha - \binom{k}{2}, \tag{2.8}$$

for $k = 1, 2, \ldots \alpha$. For MBR codes repair bandwidth is equal to the node storage capacity [23]. Hence, for MBR codes, precisely one packet is communicated from one helper node for node failure. This motives to define universally good GFR codes with asymmetric parameters. In this thesis, a GFR code in which any two nodes can share at-most one packet is called *universally good Generalized Fractional Repetition code* (universally good GFR code). For such GFR code with the reconstruction degree k, the maximum file size is

$$M(k) \geq \sum_{i=1}^{k} \alpha_i - \binom{k}{2}, \tag{2.9}$$

where $\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n$. The GFR code as considered in Example 2.1 is a universally good GFR code.

The efficiency of any GFR code can be measure by code rate. Now we define code rate for a GFR code and a DSS. For a given $n \in \mathbb{N}$, the code rate of a DSS (DSS with n distinct nodes and the maximum reconstruction degree $k < n$) is

$$\mathcal{R}_{DSS}(k) := \frac{k}{n}.$$

Similarly, for a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ defined on a DSS (DSS with n distinct nodes and the maximum reconstruction degree $k < n$), the *FR code rate* is given by

$$\mathcal{R}_{\mathscr{C}}(k) := \frac{M(k)}{\sum_{j=1}^{\theta} \rho_j}. \tag{2.10}$$

For $k = 3$ in the Example 2.1, the code rate of the $(5, 3)$ DSS is $\mathcal{R}_{DSS}(3) = \frac{3}{5}$ and GFR code rate of the GFR code is $\mathcal{R}_{\mathscr{C}}(3) = \frac{5}{13}$.

Consider two GFR codes $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ and $\mathscr{C}^* : (n^*, \theta^*, \vec{\alpha}^*, \vec{\rho}^*)$ such that, in the GFR code $\mathscr{C}$, the packet P_j is stored on the node U_i if and only if, in the GFR code $\mathscr{C}^*$, the packet P_i^* is stored on the node U_j^*, where $n^* = \theta$, $\theta^* = n$, $\alpha_j^* = \rho_j$ and $\rho_i^* = \alpha_j$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, \theta$. Essentially, the roles of packets and nodes are exchanged between $\mathscr{C}$ and $\mathscr{C}^*$. Hence, the GFR code $\mathscr{C}^*$ is called the dual of the GFR code $\mathscr{C}$. In [135], the authors studied the dual GFR code with symmetric parameters. A formal definition of dual GFR code is following.

Definition 2.12. *(Dual Generalized Fractional Repetition Code): Let $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ be a GFR code. A dual $\mathscr{C}^* : (n^*, \theta^*, \vec{\alpha}^*, \vec{\rho}^*)$ of the GFR code $\mathscr{C}$ is a GFR code such that the node $U_j^* = \{f(U_i) = P_i^* : P_j \in U_i\}$, where the bijection f is defined between the set of packets $\{P_1^*, P_2^*, \ldots, P_{\theta^*}^*\}$ and the set of nodes $\{U_1, U_2, \ldots, U_n\}$. The parameters of the dual GFR code are $n^* = \theta$, $\theta^* = n$, $\alpha_j^* = \rho_j$ and $\rho_i^* = \alpha_i$, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, \theta$.*

For the GFR code $\mathscr{C} : (5, 6, (4\ 3\ 2\ 2\ 2), (2\ 2\ 2\ 2\ 2\ 3))$ (as given in the Example 2.1), the dual GFR code $\mathscr{C}^* : (5, 4, (2\ 2\ 2\ 2\ 2\ 3), (4\ 3\ 2\ 2\ 2))$ is $U_1^* = \{P_1^*, P_2^*\}$, $U_2^* = \{P_1^*, P_3^*\}$, $U_3^* = \{P_1^*, P_4^*\}$, $U_4^* = \{P_1^*, P_5^*\}$, $U_5^* = \{P_2^*, P_3^*\}$ and $U_6^* = \{P_2^*, P_4^*, P_5^*\}$.

For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ with NPDI Matrix B, the NPDI Matrix B^* of the dual GFR code $\mathscr{C}^* : (n^*, \theta^*, \vec{\alpha}^*, \vec{\rho}^*)$ is the transpose matrix of the NPDI Matrix B *i.e.*, $B^* = B^t$. The NPDI Matrix of the GFR code $\mathscr{C}^* : (5, 4, (2\ 2\ 2\ 2\ 2\ 3), (4\ 3\ 2\ 2\ 2))$ is

$$B_{6\times 5}^* = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

By the property of transpose of a matrix, one can observe the following two remarks.

Remark 2.3. *A GFR code is corresponding to the dual GFR code if and only if the NPDI Matrix of the GFR code is a symmetric matrix.*

Remark 2.4. *Dual of a dual GFR code is corresponding to the GFR code itself.*

Two GFR codes are equivalent codes if their parameters are identical. In other words, two GFR codes with same number of nodes and same number of packets, are equivalent GFR codes if both the GFR codes have same distribution of nodes and packets.

Definition 2.13. *(Equivalent Generalized Fractional Repetition Codes): For given positive integers n and θ, let $f : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ and $g : \{1, 2, \ldots, \theta\} \to$*

$\{1, 2, \ldots, \theta\}$ *be two bijective functions. Two GFR codes* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ *and* $\mathscr{C}' : (n, \theta, \vec{\alpha}', \vec{\rho}')$ *on n nodes and* θ *packets are called equivalent GFR codes (denoted by* $\mathscr{C} \equiv \mathscr{C}'$*) only if the node packet distribution in both GFR codes are identical i.e., for the GFR code* $\mathscr{C}$*, packet* P_j $(j = 1, 2, \ldots, \theta)$ *is stored in node* U_i $(i = 1, 2, \ldots, n)$ *if and only if, for the GFR code* $\mathscr{C}'$*, packet* $P_{g(j)}$ *is stored in node* $U_{f(i)}$*.*

Consider two GFR codes $\mathscr{C} : (4, 5, (3\ 3\ 2\ 2), 2)$ and $\mathscr{C}' : (4, 5, (3\ 2\ 3\ 2), 2)$, where the node packet distribution of GFR code $\mathscr{C}$ are $U_1 = \{P_1, P_2, P_3\}$, $U_2 = \{P_1, P_4, P_5\}$, $U_3 = \{P_2, P_4\}$ and $U_4 = \{P_3, P_5\}$, and the node packet distribution of GFR code $\mathscr{C}'$ are $U_1' = \{P_1', P_2', P_3'\}$, $U_2' = \{P_2', P_3'\}$, $U_3' = \{P_1', P_3', P_5'\}$ and $U_4' = \{P_4', P_5'\}$. Observe that there exists bijective functions $f : \{1, 2, 3, 4\} \to \{1, 2, 3, 4\}$ and $g : \{1, 2, 3, 4, 5\} \to \{1, 2, 3, 4, 5\}$ such that $g(3) = 4$, $g(4) = 3$ and $g(j) = j$ for $j = 1, 2, 5$, and $f(2) = 3$, $f(3) = 2$ and $f(i) = i$ for $i = 1, 4$.

For a given GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, one can find the equivalent GFR code $\mathscr{C}' : (n, \theta, \vec{\alpha}', \vec{\rho}')$ by permuting node indexes and/or permuting packet indexes. So, GFR codes, for such permutation matrices, are equivalent. Hence, one can observe the following remark.

Remark 2.5. *Equivalent GFR codes have identical permutation on NPDI Matrices.*

2.4 Hypergraphs

For a finite set $V = \{v_1, v_2, \ldots, v_n\}$, the pair $(V, \mathcal{E})$ is called a *hypergraph* with hypervertex set V and hyperedge set $\mathcal{E}$, where $\mathcal{E} \subset \{E : E \subseteq V\}$ is a family of some subsets of V [17, 20]. For a hypergraph $(V, \mathcal{E})$, $|E|$ is called the *size of the hyperedge* $E \in \mathcal{E}$ and $|\mathcal{E}(v)|$ is the *degree of the hypervertex* $v \in V$, where $\mathcal{E}(v) = \{E : v \in E\}$. An example of such hypergraph is shown in Figure 2.5, where $V = \{v_i : i = 1, 2, \ldots, 7\}$ and $\mathcal{E} = \{E_1 = \{v_1, v_4, v_5\}, E_2 = \{v_4, v_5\}, E_3 = \{v_3, v_5, v_6\}, E_4 = \{v_7\}\}$. In this hypergraph $(V, \mathcal{E})$, hypervertices are shown by simply bold dots, and the hyperedges are represented by the covered area which contains some hypervertices. In the particular example, there are 7 hypervertices and 4 hyperedges such that $\mathcal{E}(v_1) = \{E_1\}$, $\mathcal{E}(v_2) = \emptyset$, $\mathcal{E}(v_3) = \{E_3\}$, $\mathcal{E}(v_4) = \{E_1, E_2\}$, $\mathcal{E}(v_5) = \{E_1, E_2, E_3\}$, $\mathcal{E}(v_6) = \{E_3\}$ and $\mathcal{E}(v_7) = \{E_4\}$.

In a hypergraph $(V, \mathcal{E})$, a hypervertex $v \in V$ is called *isolated hypervertex* if $\mathcal{E}(v) = \emptyset$. A hypergraph $(V, \mathcal{E})$ does not have any isolated hypervertex if $V = \bigcup_{E \in \mathcal{E}} E$. In a hypergraph $(V, \mathcal{E})$, a *loop* is a hyperedge $E \in \mathcal{E}$ with $|E| = 1$. If two distinct hyperedges share same hypervertices, then those hyperedges are called *parallel hyperedges* for the hypergraph. Similarly, if same hyperedges share two distinct hypervertices, then those hypervertices are called *parallel hypervertices*.

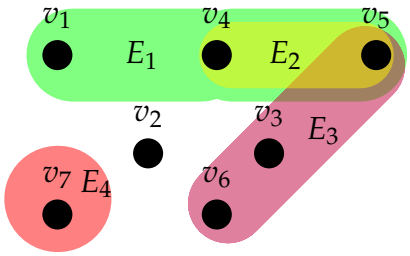

Figure 2.5: An example of a Hypergraph.

For a hypergraph $(V, \mathcal{E})$, a subset of V is called a *hypervertex cover* if each hyperedge is incident to at least one hypervertex of the subset. For an example, the set $\{v_4, v_6, v_7\}$ is a hypervertex cover of the hypergraph given in Figure 2.5. Note that the hypervertex cover of any hypergraph exists, but it is not unique for some hypergraph.

Now we collect some properties of hypergraphs in the following three lemmas.

Lemma 2.1. *([17, Theorem 1, Chapter 1]) Given m integers $r_1, r_2, \ldots, r_m$ and n-tuple of integers $d_1 \geq d_2 \geq \ldots \geq d_n$, there exist a hypergraph $(V,\mathcal{E})$ with $V = \{v_1, v_2, \ldots, v_n\}$ and $\mathcal{E} = \{E_1, E_2, \ldots, E_m\}$ such that $deg(v_i) = d_i$ for $i \leq n$ and $|E_j| = r_j$ for $j \leq m$ if and only if*

1. $\sum_{j=1}^{m} min\{|E_j|, k\} \geq \sum_{i=1}^{k} deg(v_i)$ *(for $k < n$), and*
2. $\sum_{j=1}^{m} |E_j| = \sum_{i=1}^{n} deg(v_i)$.

Lemma 2.2. *([17, Theorem 4, Chapter 1]) Every hypergraph $(V,\mathcal{E})$ with no repeated hyperedge satisfies*

$$\max\{deg(v_i) : i = 1, 2, \ldots, |V|\} \leq 2^{|V|-1}.$$

Lemma 2.3. *([17, Theorem 6, Chapter 1]) For $\mathcal{E} = \{E_1, E_2 \ldots, E_m, F_1, F_2, \ldots, F_m\}$, let $(V,\mathcal{E})$ be a hypergraph of order n with $2m$ hyperedges such that $E_i \cap F_j = \emptyset$ if and only if $i = j$. Then*

$$\sum_{j=1}^{m} \binom{|E_j| + |F_j|}{|F_j|}^{-1} \leq 1.$$

A hypergraph $(V', \mathcal{E}')$ is called a *sub-hypergraph* of a hypergraph $(V, \mathcal{E})$ if $V' \subseteq V$ and $\mathcal{E}' \subseteq \{E' \neq \emptyset : E' \subseteq E \cap V', E \in \mathcal{E}\}$. For an example, a hypergraph $(V', \mathcal{E}')$ with $V' = \{v_1, v_3, v_4, v_5, v_6\}$ and $\mathcal{E}' = \{E'_1 = \{v_1, v_4, v_5\}, E'_2 = \{v_4\}, E'_3 = \{v_3, v_5\}\}$, is a sub-hypergraph of the hypergraph $(V, \mathcal{E})$ (as shown in Figure 2.5). For a hypergraph $(V, \mathcal{E})$, hypergraph $(V', \mathcal{E}')$ is called sub-hypergraph induced by the set $V' \subset V$ if $\mathcal{E}' = \{E' \neq \emptyset : E' = E \cap V', E \in \mathcal{E}\}$. A hypergraph is called *connected hypergraph* if there exists a sub-hypergraph with the same hypervertex set such that the sub-hypergraph is isomorphic to a connected graph. The hypergraph $(V, \mathcal{E})$ (Figure 2.5) is not a connected hypergraph since there is an isolated hypervertex.

Consider a hypergraph $(V, \mathcal{E})$ without isolated hypervertices. For a finite set $V^* = \{v_1^*, v_2^*, \ldots, v_{|\mathcal{E}|}^*\}$, a hypergraph $(V^*, \mathcal{E}^*)$ is called the *dual* of the hypergraph $(V, \mathcal{E})$ if their exist a bijection $g : \mathcal{E} \to V^*$ such that $E_j = \{g(E_i) = v_i^* : v_j \in E_i\}$, where $j = 1, 2, \ldots, |V|$. For an example, a hypergraph $(V^*, \mathcal{E}^*)$ with $V^* = \{v_1^*, v_2^*, v_3^*, v_4^*\}$ and $\mathcal{E}^* = \{E_1^* = \{v_1^*\}, E_2^* = \{v_3^*\}, E_3^* = \{v_1^*, v_2^*\}, E_4^* = \{v_1^*, v_2^*, v_3^*\}, E_5^* = \{v_4^*\}\}$, is the dual of a hypergraph $(V, \mathcal{E})$ with $V = \{v_1, v_2, v_3, v_4, v_5\}$ and $\mathcal{E} = \{E_1 = \{v_1, v_3, v_4\}, E_2 = \{v_3, v_4\}, E_3 = \{v_2, v_4\}, E_4 = \{v_5\}\}$, where $g(E_j) = v_j^*$ and $i = 1, 2, \ldots, 5$.

In a hypergraph, if two hyperedges share one node maximum, then the hypergraph is called linear hypergraph. For example, a hypergraph $(V,\mathcal{E})$ is a *linear hypergraph*, where $V = \{v_1, v_2, v_3, v_4\}$ and $\mathcal{E}$ = $\{\{v_1, v_2.v_3\}, \{v_1, v_4\}, \{v_4\}\}$. For such hypergraph, one can find bound on the number of hyperedges as given in the following Lemma.

Lemma 2.4. *For a linear hypergraph* $(V, \mathcal{E})$*,*

$$|\mathcal{E}| \geq \sum_{v \in V} |\mathcal{E}(v)| - \binom{|V|}{2}. \tag{2.11}$$

Proof. In a linear hypergraph $(V, \mathcal{E})$, at most $\binom{|V|}{2}$ hyperedges are repeated in the hyperedge count $\sum_{v \in V} |\mathcal{E}(v)|$. So, the total number of hyperedges of $\mathscr{H}$ is bounded by $\sum_{v \in V} |\mathcal{E}(v)| - \binom{|V|}{2}$. □

A sub-hypergraph of a linear hypergraph is also a linear hypergraph. For a positive integer k (not more than the size of the hypergraph), one can find a bound on the size of sub-hypergraph induced by any k-element subset. Formally, the bound is discussed in the following lemma.

Lemma 2.5. *For* $V = \{v_i : i = 1, 2, \ldots, n\}$*, consider a linear hypergraph* $(V, \mathcal{E})$ *with* $|\mathcal{E}(v_i)| \leq |\mathcal{E}(v_j)|$*, for* $i < j$ *and* $i, j = 1, 2, \ldots, n$*. For a given positive integer* $k < n$*, let* $(V' \subset V, \mathcal{E}')$ *be a induced sub-hypergraph of the hypergraph* $(V, \mathcal{E})$ *with* $\mathcal{E}' = \{E' \neq \emptyset : E' = E \cap V', E \in \mathcal{E}\}$ *and* $|V'| = k$*. Then*

$$\min\{|\mathcal{E}'| : V' \subset V, |V'| = k\} \geq \sum_{i=1}^{k} |\mathcal{E}(v_i)| - \binom{k}{2}. \tag{2.12}$$

Proof. For $V = \{v_i : i = 1, 2, \ldots, n\}$, let $(V, \mathcal{E})$ be a linear conditional hypergraph with $|\mathcal{E}(v_i)| \leq |\mathcal{E}(v_j)|$, for $1 \leq i < j \leq n$. For a given positive integer $k < n$, let V' be an arbitrary k-element subset of the set V. Consider a hypergraph $(V', \mathcal{E}')$, where $\mathcal{E}' = \{E' \neq \emptyset : E' = E \cap V', E \in \mathcal{E}\}$. By definition of the degree of a hypervertex,

$$\begin{aligned} \sum_{v \in V'} |\mathcal{E}(v)| &\geq \sum_{i=1}^{k} |\mathcal{E}(v_i)| \\ \Rightarrow \sum_{v \in V'} |\mathcal{E}(v)| - \binom{|V'|}{2} &\geq \sum_{i=1}^{k} |\mathcal{E}(v_i)| - \binom{k}{2}. \end{aligned}$$

A sub-hypergraph of a linear hypergraph is also linear. So, using Lemma 2.4, one can conclude that

$$|\mathcal{E}'| \geq \sum_{v \in V'} |\mathcal{E}(v)| - \binom{|V'|}{2} \geq \sum_{i=1}^{k} |\mathcal{E}(v_i)| - \binom{k}{2}.$$

The chosen k-element set V' is an arbitrary subset of set V, so

$$\min\{|\mathcal{E}'| : V' \subset V, |V'| = k\} \geq \sum_{i=1}^{k} |\mathcal{E}(v_i)| - \binom{k}{2}.$$

Hence, the lemma is proved. □

Now we collect following two lemmas for linear hypergraphs.

Lemma 2.6. *([17, Theorem 3, Chapter 1]) For every linear hypergraph $(V,\mathcal{E})$, we have*

$$\sum_{E \in \mathcal{E}} \binom{|E|}{2} \leq \binom{|V|}{2}.$$

Lemma 2.7. *([17, Chapter 1]) For a linear hypergraph, the dual is linear.*

A hypergraph $(V, \mathcal{E})$ is called a *simple hypergraph* if none of the subset in $\mathcal{E}$ is a proper subset of another. For example, the hypergraph $(V, \mathcal{E})$ with $V = \{v_i : i = 1, 2, 3, 4\}$ and $\mathcal{E} = \{E_j : j = 1, 2, 3\}$ is a simple hypergraph, where $E_1 = \{v_1, v_2, v_3\}$, $E_2 = \{v_1, v_3, v_4\}$ and $E_3 = \{v_2, v_4\}$. A bound for simple hypergraph is given in following lemma.

Lemma 2.8. *([17, Theorem 3, Chapter 1]) Every simple hypergraph $(V, \mathcal{E})$ satisfies*

1. $\sum_{E \in \mathcal{E}} \leq \binom{|V|}{|E|}^{-1} \leq 1$,
2. $|\mathcal{E}| \leq \binom{|V|}{|E|}^{-1} \leq 1$.

For a positive integer r, a hypergraph $(V, \mathcal{E})$ is called a *r-uniform hypergraph*, if $|E| = r$ for each $E \in \mathcal{E}$. For example, the hypergraph $(V, \mathcal{E})$ with $V = \{v_i : i = 1, 2, 3, 4\}$ and $\mathcal{E} = \{E_j : j = 1, 2, 3\}$ is 3-uniform hypergraph, where $E_1 = \{v_1, v_2, v_3\}$, $E_2 = \{v_1, v_3, v_4\}$ and $E_3 = \{v_1, v_2, v_4\}$. For an r-uniform linear hypergraph, a bound is given in following lemma.

Lemma 2.9. *([17, Theorem 3, Chapter 1]) For every r-uniform linear hypergraph $(V,\mathcal{E})$, we have*

$$|\mathcal{E}| \leq \frac{n(n-1)}{r(r-1)}.$$

Lemma 2.10. *([17, Proposition 2, Chapter 1]) An n-tuple of integers $d_1 \geq d_2 \geq \ldots \geq d_n$, is a degree sequence of a connected uniform hypergraph of rank r if and only if*

1. $\sum_{i=1}^{n} d_i$ *is a multiple of r,*
2. $d_i \geq 1$ $(i = 1, 2, \ldots, n)$*,*
3. $\sum_{i=1}^{n} d_i \geq \frac{r(n-1)}{r-1}$*, and*
4. $d_1 \leq m = \frac{\sum_{i=1}^{n} d_i}{r}$.

For a positive integer r, a hypergraph $(V, \mathcal{E})$ is called a *r-regular hypergraph*, if $deg(v) = r$ for each $v \in V$. For example, the hypergraph $(V, \mathcal{E})$ with $V = \{v_i : i = 1, 2, 3, 4\}$ and $\mathcal{E} = \{E_j : j = 1, 2, 3\}$ is 2-regular hypergraph, where $E_1 = \{v_1, v_2, v_3\}$, $E_2 = \{v_1, v_3, v_4\}$ and $E_3 = \{v_2, v_4\}$.

Lemma 2.11. *([17, Chapter 1]) For a r-regular hypergraph, the dual is a r-uniform hypergraph.*

Lemma 2.12. *([17, Chapter 1]) For a r-uniform hypergraph, the dual is a r-regular hypergraph.*

CHAPTER 3

Regenerating Codes and Fractional Repetition Codes

Any idea, plan, or purpose may be placed in the mind through repetition of thought. -*Napoleon Hill* [1]

In this chapter, we give a brief overview of regenerating codes and FR codes.

3.1 Regenerating Codes

In Chapter 2, DSS with homogeneous parameters is formally defined. For such homogeneous DSS, regenerating code is introduced in the same chapter. Further, a fundamental bound (Inequality 2.1) on file size is also discussed for the homogeneous DSS. Further, a trade-off between storage and repair bandwidth is also discussed in the chapter. All the points on the trade-off curve can be obtained by linear network codes over finite fields, where the size of the field is sufficiently large [117, 118]. The optimal points (MBR code and MSR code) on the trade-off curve are discussed in the same chapter. The trade-off between storage and repair bandwidth for exact repair is studied in [37]. In [95], Shah *et al.* calculated cut-set lower bound on repair bandwidth for a special flexible setting for homogeneous DSS. In a nice survey [24], an overview of some existing results and repair models on DSS are explored. In [87], the trade-off between storage capacity and repair bandwidth

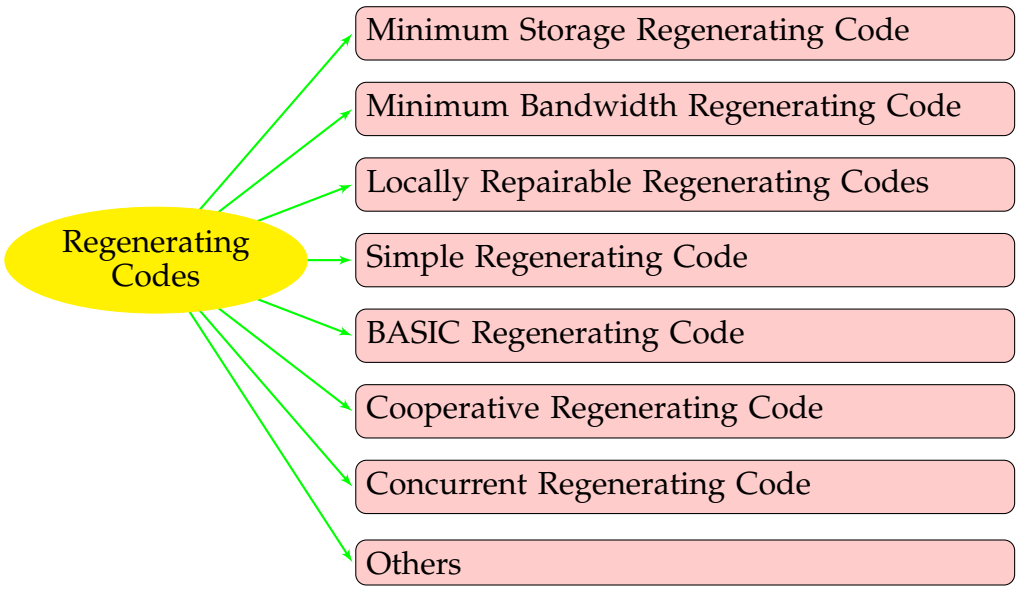

Figure 3.1: Various regenerating codes.

is investigated for exact repair linear regenerating codes for $k = d = n - 1$. In [109, 110], for functional repair regenerating codes and exact repair regenerating codes, the existence of a gap between the storage-bandwidth trade-off is discussed. For the trade-off between storage and repair bandwidth, the outer bound is studied in [91]. In [61], the outer bound for (n, k, d) DSS is calculated and the bound gets tighter as k approaches to n. For the $(5, 4, 4)$ exact-repair regenerating codes, the fundamental trade-off between storage and repair bandwidth is classified in [111].

3.1.1 Regenerating codes with additional system properties

In this section a brief overview on regenerating codes (see Figure 3.1) with additional system properties such as locality and multiple node failure are discussed.

1. *Locally Repairable codes*: The bandwidth requirement of the regenerating codes are significantly less than the bandwidth requirement of erasure codes by allowing to communicate with more than k helper nodes for a failed node. For such systems, using these codes, it seems to be the high I/O traffic of those nodes. In the systems, the I/O traffic is proportional to the number of helper nodes r communicated in the repair process [78]. In other words, an erasure code has locality r if any failed node can be repaired by communicating r helper nodes. In [36], Gopalan *et al.* studied the codewords locality and

obtained bound in terms of the code-word length n, the message length k, the minimum distance d, and the code locality r. In [36], for a linear erasure code, the number of parity symbols is bounded as,

$$n - k \geq \lceil k/r \rceil + d - 2. \tag{3.1}$$

To satisfy both the cases the maximum reliability $r = k$, and the minimum cost, $r << k$, in [78], it has been shown that each storage node in the system must store some additional information such that $\alpha = (1+\epsilon)M/k$, where $\epsilon \geq 0$. Hence, the Inequality 3.1, reduces to

$$\begin{aligned} d &\leq n - \left\lceil \frac{M}{\alpha} \right\rceil - \left\lceil \frac{M}{r\alpha} \right\rceil + 2 \\ &= n - \left\lceil \frac{k}{1+\epsilon} \right\rceil - \left\lceil \frac{k}{r(1+\epsilon)} \right\rceil + 2. \end{aligned} \tag{3.2}$$

Notice that the bound 3.1 can be applied also to nonlinear codes. Finding ϵ is an interesting optimization problem to minimize the overhead storage [78]. A regenerating code with locality $r = d < k$, $\beta < \alpha$, and functional repair is called *locally repairable regenerating codes* [3, 5, 41].

2. *Simple Regenerating Code*: In [80] the Simple Regenerating code is studied. A Simple Regenerating code is a (n, k) Regenerating code with rate $\frac{2k}{3n}$, where the rate can be made arbitrarily close to $\frac{2}{3}$, for given erasure resiliency.

3. *BASIC Regenerating Code*: Motivated by [48, 45], BASIC (Binary Addition and Shift Implementable Cyclicconvolutional) coding framework was given in [44, 100]. In BASIC regenerating code, the coding is enabled such that a failed node is repaired by XOR operation and properties of bit-wise cyclic-shift. Using XOR operations and bit-wise acyclic-shifts, a class of regenerating codes is given in [43]. The same line of work on computational complexity for Network coding problem is done in [54, 53, 50]. In [42], an overview for BASIC codes is given, and it discusses some open problems related to them.

4. *Cooperative Regenerating Code*: The regenerating code which can tolerate mul-

tiple node failure simultaneously, is called Cooperative Regenerating Code. For the regenerating code, the failed nodes are repaired with the helper nodes and newly generated nodes [46, 52, 99, 101]. In [46], the trade-off between the node storage capacity and repair bandwidth are discussed. In [114], a multiple-loss flexible repair mechanism is proposed on the minimal-storage nodes to repair multiple failures. On the trade-off curve between node storage and repair bandwidth for the cooperative regenerating codes, the two extreme points correspond to minimal-storage cooperative regenerating (MSCR) codes and minimal-bandwidth cooperative regenerating (MBCR) codes. In [113, 64], constructions based on product matrix are proposed for exact MBCR and MSCR codes. the interference alignment is introduced in [21]. As a family of RGC, fractional repetition [67] codes are also considered [49]. In [139], the centralized exact repair of multiple failures in DSS is discussed. For the DSS, a trade-off between the normalized storage and repair bandwidth is also investigated in [139]. In [60], an outer bound on the trade-off between storage repair bandwidth of linear cooperative regenerating codes for exact repair with $d = k = n - r$ is given, where the linear cooperative regenerating code can tolerate maximum r node failure simultaneously. For exact-repair linear cooperative regenerating codes, an outer bound on the storage-bandwidth trade-off is given in [62].

5. *Concurrent regenerating codes*: The concurrent repair framework for both single and multiple node failures have been studied in [125], and the codes are simpler and more stable than the cooperative Regenerating codes.

6. *Others*: A generalization of regenerating codes as “cloud of clouds" framework is considered in [86]. In this paper, a DSS model of n clusters is considered such that each cluster contains m nodes. The system is fully connected *i.e.*, nodes in a cluster are connected through intra-cluster links, and clusters are connected through inter-cluster links. For the DSS model, the trade-off between inter-cluster repair bandwidth and storage overhead is discussed in [86].

3.1.2 Regenerating codes with general parameters

For heterogeneous DSS, repair problem with data allocation is studied in [35, 63, 66, 73, 121]. In [7], a DSS is studied in which storage nodes are divided into two sets such that communication costs of any two storage nodes from a different group are different and all nodes in the same group have same communication cost. In [31] (Extended version [31]), two rack model of a DSS has been studied. In the two rack model, the communication cost between the nodes in the same rack is significantly smaller than the communication cast between nodes in the two different racks. Hence, for a node failure, transmitting more data from the same rack is reducing communication cost. Using the information flow graph for the two rack DSS model, the fundamental trade-off is plotted between storage cost and repair cost in [7] and [31]. A non-homogeneous DSS is considered in [112], where a supernode exists in the DSS such that the storage capacity, reliability, and availability probability are more than the other nodes in the system. It has been shown that this model can achieve the optimal bandwidth-storage trade-off bound in [22, 23] with a smaller file and alphabet size than the traditional homogeneous storage network in [79]. For heterogeneous DSS, repair cost can be reduced by allowing helper nodes to encode the codewords of other nodes [32]. In [6, 7, 31, 83, 121], authors investigated the trade-off between storage capacity and repair bandwidth for the generalized regenerating codes and shown that each point on the curve is achievable. In the generalized regenerating code, set of all nodes is divided into two partitions such that every node in each partition has uniform parameters (α_i, d_i, β_i) $(\forall i \in \{1, 2\})$. In [29], Ernvall *et al.* calculated the capacity bounds of a heterogeneous DSS having dynamic repair bandwidth and constant repair degree. In [14], capacity bound is calculated for heterogeneous DSSs with dynamic repair bandwidth, where node repair is done by some specific helper nodes. In [122], the trade-off between system storage cost and system repair cost is investigated for heterogeneous DSSs with dynamic storage and repair cost. In [55], selective regenerating codes are proposed, and the trade-off between storage per node and repair bandwidth is plotted for the regenerating codes, where selective regenerating codes are those codes in which helper nodes of a failed node are chosen in such a smart way that it reduces repair

bandwidth. In [94, 91], the trade-off between node storage capacity and bandwidth is analyzed for exact repair. In [90], authors gave constructions for interior points of the normalized trade-off. In [25], Duursma improved bounds on exact repair for regenerating codes. In [4], authors talked about the improvement of regenerating codes by a smart selection of helper nodes for arbitrary node failure. Motivated by this, in Chapter 4, the fundamental bound is calculated for the generalized DSS, and, using the bound, the trade-off between the storage cost and cost of repair bandwidth is obtained. For more details, see [9, 12, 24].

3.2 On Fractional Repetition Codes

A DRESS code with parameters $[(\theta, M(k), k, (n, \alpha, \rho))]$ is a two layer code with an outer layer is (θ, M) MDS code and an inner code is an FR code $\mathscr{C} : (n, \theta, \alpha, \rho)$. To store a file on a DSS, the file with M packets (each has equal size) is first encoded by using (θ, M) MDS; next, the replicas of those θ packets are placed on the n nodes such that node U_i ($i \in \{1, 2, \ldots, n\}$) of the DSS stores α packets and ρ copies of a packet are stored in the system. A data collector can retrieve the file by downloading packets from any set of k nodes. If a node U_j fails, then it can be repaired by using a set of d other active nodes. In the Figure 2.4, the encoding scheme is shown for an FR code defined on homogeneous DSS.

1. *Supported file size*: For an FR code, the maximum file with size $M(k)$ is bounded by the Inequality 2.7, where each has same node storage capacity and same replication factors. An FR code $\mathscr{C} : (n, \theta, \alpha, \rho)$ is called *universally good* [26] or *optimal* [105] if the DRESS code with parameters $[(\theta, M(k), k, (n, \alpha, \rho))]$ satisfies the Inequality 2.8 for any $k \leq \alpha$, where the right hand side of the Inequality 2.8 is the maximum file size that can be stored using an MBR code in the DSS, and it is known the MBR capacity [23]. It is interesting to consider those codes which allow storing larger files in MBR codes. To satisfy 2.8 the FR code should satisfy the Inequality 2.8. Note that if an FR code $\mathscr{C}$ satisfies 2.8 then $|U_i \cap U_j| \leq 1$, for $U_i, U_j \in \mathscr{C}$, $i \neq j \in \{1, 2, \ldots, n\}$ [89].

   For a given DSS with exact and uncoded repair, the maximum file that can be

stored the DSS is called FR capacity and is denoted by $A(n,k,\alpha,\rho)$ [26]. Two upper bounds on the FR capacity $A(n,k,\alpha,\rho)$ for a (n,k,d) DSS are calculated:

$$A(n,k,\alpha,\rho) \leq \left\lfloor \frac{n\alpha}{\rho}\left(1-\frac{\binom{n-\rho}{k}}{\binom{n}{k}}\right)\right\rfloor; \tag{3.3}$$

$$A(n,k,\alpha,\rho) \leq \Phi(k), \tag{3.4}$$

where $\Phi(1)=\alpha, \Phi(k+1)=\Phi(k)+\alpha-\left\lceil\frac{\rho\Phi(k)-k\alpha}{n-k}\right\rceil$. The bound given in (3.4) is more tighter the bound given in (3.3). For given k, the FR code is called *optimal* if it satisfies

$$\min\{|\cup_{i\in i} U_i| : |I|=k, I\subset\{1,2,\ldots,n\}\} = A(n,k,\alpha,\rho) \tag{3.5}$$

2. *Minimum distance*: For any FR code, some nodes fail then the remaining nodes do not have the sufficient packets to recover the stored file. Therefore, in [127], the *minimum distance* of FR code is defined as the minimum number of nodes such that if these nodes fail then the active nodes can not recover the complete file. More precisely, the minimum distance of an FR code $\mathscr{C}:(n,\theta,\alpha,\rho)$ on an (n,k,d)-DSS, denoted by d_{min}, is the size of the smallest subset $\mathcal{U}$ of $\{U_i : i=1,2,\ldots,n\}$ such that the number of distinct packets in $\{U_i : i=1,2,\ldots,n\}\backslash\mathcal{U}$ is less than the file size $M(k)$. In [127, Lemma 4], an alternate definition of the minimum distance of an FR code is given.

Lemma 3.1. *[127, Lemma 4] For an FR code $\mathscr{C}:(n,\theta,\alpha,\rho)$ on an (n,k,d)-DSS with maximum file size $M(k)=\theta-\delta+1$, the minimum distance*

$$d_{min} = \min\left\{|\{i : U_i\cap S\neq\emptyset\}| : |S|=\delta, S\subset\{P_1,P_2,\ldots,P_\theta\}\right\}, \tag{3.6}$$

where $\delta\in\{1,2,\ldots,\theta\}$.

The FR code follows a Singleton-like bound on the minimum distance [77].

For an FR code $\mathscr{C} : (n, \theta, \alpha, \rho)$ on an (n, k, d)-DSS, the minimum distance

$$d_{min} \leq n - \left\lceil \frac{M(k)}{\alpha} \right\rceil + 1. \tag{3.7}$$

3.2.1 Classification of constructions of FR codes on homogeneous DSSs

The classification of constructions of FR codes (see Figure 3.2) defined on homogeneous DSSs as follows.

- *Algebraic Constructions*: Using partially ordered sets, FR codes are constructed in [11]. In [81, 56], FR codes are constructed using difference sets. In [56], FR codes are constructed using relative difference sets with $\lambda = 1$, cyclic relative difference sets and non-cyclic relative difference sets. FR codes are constructed from quasi-perfect difference families and perfect difference families in [81]. In the paper [108], two constructions of FR codes are proposed which are based on affine permutation matrices and circulant permutation matrices. In [126], FR code, called *Cyclic repetition erasure code*, is constructed using circulant permutation matrices.

- *Algorithmic Constructions*: In the paper [10], an algorithm is presented for constructing the node-packet distribution matrix of an FR code, where the node-packet distribution matrix is the matrix representation of the FR code.

- *Constructions using Combinatorics*: In [26], constructions of FR codes are given using steiner systems, Fano plane and their dual designs. In [74, 75] and [77], FR codes are constructed using Affine resolvable designs, mutually orthogonal Latin squares and Kronecker product technique. In [120], the construction of the optimal FR codes is proposed by t-designs. In [138], the FR codes are constructed using uniform group divisible designs. The FR codes are constructed using regular packing designs in [128].

- *Constructions using Finite Geometry*: In [57], FR codes are constructed using finite geometries and corresponding bipartite cage graphs.

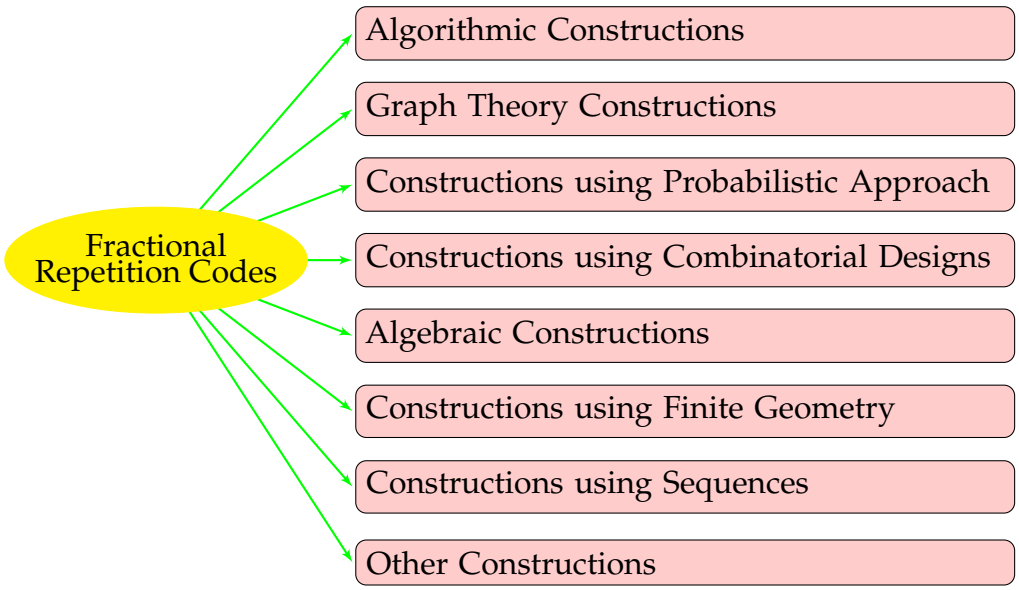

Figure 3.2: Constructions of FR codes.

- *Constructions using Probabilistic Approach*: Randomized FR codes are constructed in [82] using the balls-and-bins model.

- *Graph Theory Constructions*: In [26, 57, 105, 76, 120], FR codes are constructed using graphs, where the distribution of packets on nodes in an FR code are associated with adjacency of edges with vertices in respective graph. In [26], FR codes are constructed from complete graphs and regular graphs. For given parameters and the fewest number of storage nodes, FR codes are constructed using bipartite cage graphs in [57]. In [105], FR codes are constructed using Turàn graph and cage graphs. Using regular graphs with a given girth, FR codes are constructed in [76]. In [120], Xu *et al.* have looked FR code as a biregular graph, and shown that finding FR code with k fault-tolerance and the minimum redundancy, is the same as the *Zarankiewicz* problem in graph theory.

- *Other Constructions*: In [85], FR codes are constructed using ring construction. In the construction, nodes are placed in a circle, and all the replicated packets are distributed to the nodes in specific order. In [98], FR codes are defined as hypergraphs and some bounds are calculated.

In [10], FR codes are enumerated for given number of nodes. Algorithms

are presented in [16] which compute the repair degree d of the FR code and the reconstruction degree k. For an FR code, the dual bound on file size is investigated in [135] and improved the bound is in [136].

3.2.2 FR code constructions with additional system properties

Some FR codes with an additional system property (see Figure 3.3), are also studied in the literature. We summarise these results as follows.

1. In [76], the minimum distance of FR code is defined as the minimum number of nodes such that if these nodes fail then the active nodes cannot recover the complete file. The Singleton like bound (Inequality 3.7) on the minimum distance is established in [76]. FR codes are constructed in [127] using t-design.

2. In [129], *Adaptive* FR code is constructed, where the adaptive FR code can adapt the system changes such as disk failure. The Adaptive FR code is obtained from symmetric design by removing some points from the blocks.

3. In [102, 103, 104] and [105] Fractional Repetition Batch (FRB) code is studied which provides uncoded repairs and parallel reads of subsets of stored packets. An FR code is an FRB code if any batch of t information packets can be decoded by reading at most one symbol from each node. The FRB codes are constructed using a regular graph and complete bipartite graph.

4. In [69, 70, 71], the Locally Repairable FR codes are constructed using graphs, where an FR code is called *Locally Repairable* FR code if $k > d$. The Locally Repairable FR code constructions are proposed based on the symmetric design in [130]. Locally recoverable FR code is constructed using Kronecker product technique in [75]. For a Locally Repairable FR code, note that each failed storage node can have multiple local repair alternatives in the system. In [76], Locally recoverable FR code is constructed using regular graph. In [51], Locally Repairable FR codes are constructed using t-design. Locally Repairable FR codes defined on (n, k, d, α) DSS with maximum file size $M(k)$

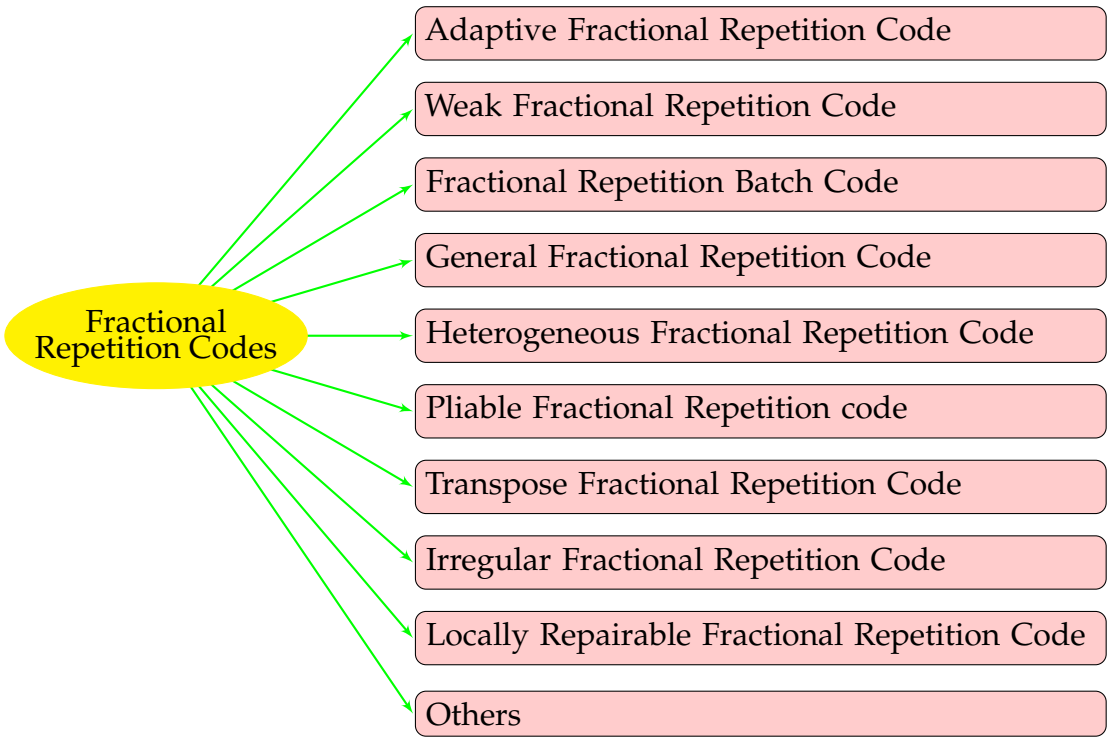

Figure 3.3: Various FR codes.

and minimum distance d_{min} satisfy the bound $d_{min} \leq n - \left\lceil \frac{M(k)}{\alpha} \right\rceil - \left\lceil \frac{M(k)}{d\alpha} \right\rceil + 2$,

5. In [107], *Pliable* FR codes are studied, where storage per node and replication factor for each packet can be adjusted simultaneously. The Pliable FR codes are constructed using Euclidean geometry, circulant permutation matrices, extended RS codes, affine permutation matrices, geometry decomposition, zigzag codes and Euler squares in [107].

6. In [84], *Load-Balanced* FR codes are studied, where multiple node failures in the FR code can be repaired in a sequence by downloading at most one block from any other active node. Note that the property reduces the number of disk reads which need to repair multiple nodes in the system.

3.2.3 Construction of FR codes with general parameters

FR codes (see Figure 3.3) are generalized into the following general directions.

1. FR codes are generalized to *Weak* [38, 85], *General* [132, 137] or *Irregular* [123] FR codes, where different number of packets are stored in each node. Those FR codes are constructed using partial regular graphs [38] and group divisible designs [137]. In [85], FR codes are constructed using ring construction, where all the replicated packets are distributed to nodes in specific order. In [132], the FR code is constructed using packings, coverings, and pairwise balanced designs. In [123], FR codes are studied as uniform hypergraphs. For the FR code on (n, k, d) DSS, the maximum file size

$$M(k) \geq \sum_{i=1}^{k} d_i - \binom{k}{2}, \tag{3.8}$$

where $d_1 \leq d_2 \leq \ldots \leq d_n$ [137]. Note that most of the constructions are from graph theory or design theory. In the Chapter 5, we have given a beautiful connection between sequences and such FR codes. FR codes are constructed using sequences, and parameters and properties are studied on the light of sequences in the Chapter 5.

2. FR codes are generalized to *variable* [131] or *heterogeneous* [133] FR codes, where each packet has different replication factor. The specific FR codes are constructed using group divisible designs.

3. In [134], the *Flexible* FR codes are generalized to FR codes, where node storage capacity of different nodes are different, the replication factor of different packets are different, and any two distinct nodes contain at most one common packet. For the FR code, the maximum file size $M(k)$ is bounded as

$$\sum_{i=n-k+1}^{n} \alpha_i - \binom{k}{2} \leq M(k) \leq \sum_{i=1}^{k} \alpha_i, \tag{3.9}$$

where $\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n$. The FR code is constructed using pairwise balance designs and group divisible designs. A heuristic construction for the FR code is also given in [134].

An excellent connection between Hypergraphs and those FR codes is studied in the Chapter 6. In the same chapter, a bijective function between Hy-

pergraphs and FR codes is defined to explore the FR codes in the light of Hypergraphs. Some basic problems like the existence of FR code for given parameters and optimality of FR codes are answered using properties of Hypergraphs.

CHAPTER 4

On Heterogeneous Distributed Storage System

The more storage you have, the more stuff you accumulate. -*Alexis Stewart* [1]

In Chapter 2, heterogeneous DSS is introduced. In this chapter, for the heterogeneous DSS, a linear bi-objective optimization problem with subject to the fundamental bound is established, and the trade-off curve between storage cost and repair costs per node is plotted. The fundamental bound is analyzed for a specific heterogeneous DSS with constant repair traffic and reconstruction degree. Conditions on the parameters of the codes on heterogeneous DSS are established. The techniques are similar to [119, 122].

4.1 Trade-Off for Heterogeneous Distributed Storage Systems

In this section, we have discussed the information flow graph, bi-objective optimization problem and trade-off for the heterogeneous DSS.

4.1.1 Information flow graph

For a network, the information flow graph is an acyclic weighted directed graph. At time t, an information flow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with a heterogeneous DSS, is shown in Figure 4.1, where $\mathcal{V}$ is a set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set

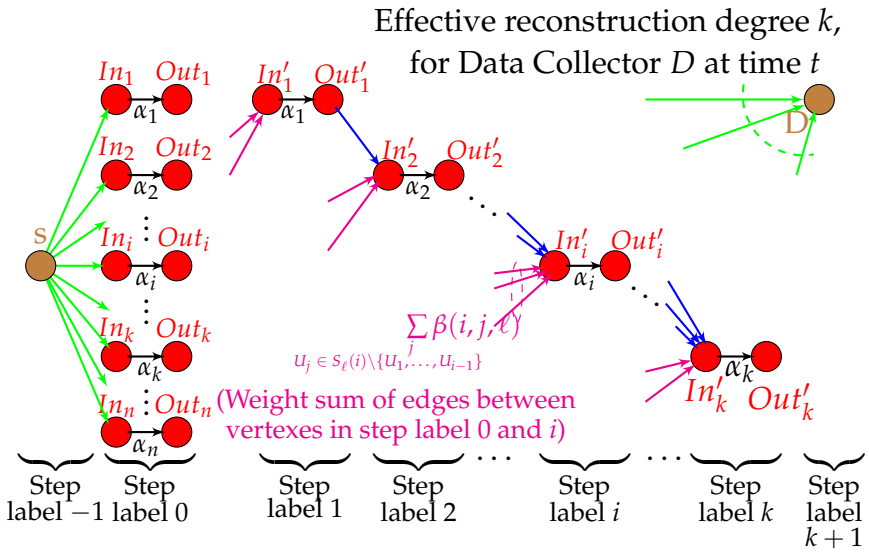

Figure 4.1: Information flow graph $\mathcal{G}$ for a heterogeneous DSS.

of edges. The information flow graph $\mathcal{G}$ is divided into $k+3$ (at time t, k being the reconstruction degree for a data collector) steps, starting from step label -1 to label $k+1$. Step label -1 contains source node say "s" and step label $k+1$ contains data collector node say "D". A typical node U_i $(i \in \{1,2,\ldots,n\})$ in heterogeneous DSS, is mapped to a pair of vertices "In_i" and "Out_i" in $\mathcal{V}$ $s.t.$ $(In_i, Out_i) \in \mathcal{E}$. Storage capacity α_i of node U_i is mapped to $w(In_i, Out_i)$, where $w(In_i, Out_i)$ is the weight associated with the edge $(In_i, Out_i) \in \mathcal{E}$. As given in Figure 4.1, the graph $\mathcal{G}$ consists n pair of vertices named In_i and Out_i in step label 0 which associated with n nodes U_i $(i \in \{1,2,\ldots,n\})$ of the heterogeneous DSS.

For a heterogeneous DSS, a failed node U_i $(i \in \{1,2,\ldots,n\})$ is replaced by new node U'_i having α_i packets during repair presses. Suppose packets from nodes of a surviving set $S_\ell(i)$ $(\ell \in \{1,2,\ldots,\tau_i\})$ are downloaded for the new node U'_i. Let $S_\ell(i) = \{U_j : j \in J \text{ for some } J \subset \{1,2,\ldots,n\}\backslash\{i\}\}$. For a node $U_j \in S_\ell(i)$, suppose $\beta(i,j,\ell)$ packets are downloaded from the node U_j for the node failure U_i. In the information graph $\mathcal{G}$, the node U'_i is mapped to a new pair of nodes In'_i and Out'_i $s.t.$ $(In'_i, Out'_i) \in \mathcal{E}$ with $w(In'_i, Out'_i) = \alpha_i$. The weight associated with the edge $(Out_j, In'_i) \in \mathcal{E}$ is $w(Out_j, In'_i) = \beta(i,j,\ell)$.

For a DSS with n nodes, at time t, consider a reconstruction set $\mathcal{A} = \{U_{g_i} : i = 1,2,\ldots,k\} \subset \{U_1, U_2, \ldots, U_n\}$ with reconstruction degree k. Recall the $k = |\mathcal{A}|$. The nodes can fail in any order, so, one can find a node sequence $\vec{u}$ according to

the order of the failed nodes. Let $\vec{u} = (U_{\lambda_1}\ U_{\lambda_2} \dots U_{\lambda_{|\mathcal{A}|}})$, where λ is a permutation on $\{g_1, g_2, \dots g_{|\mathcal{A}|}\}$. Here, we have considered only single node failure at a time t, because, the multiple node failures can be considered as a sequence of single node failure. So, the sequence of failed nodes can be repaired using some sequence of a surviving set called surviving sequence. Consider a surviving sequence $\vec{s} = (S_{\ell_1}(\lambda_1)\ S_{\ell_2}(\lambda_2) \dots S_{\ell_{|\mathcal{A}|}}(\lambda_{|\mathcal{A}|}))$ for the node sequence $\vec{u}$. At time t, we have considered the reconstruction degree is k, so $|\mathcal{A}| = k$. For simplicity, consider the following.

- Consider the reconstruction set $\mathcal{A} = \{U_1, U_2, \dots, U_k\}$ *i.e.*, the permutations g_i is identical permutation for $i = 1, 2, \dots, k$.
- For the reconstruction set $\mathcal{A}$, the nodes are failing in the sequence $\vec{u} = (U_1\ U_2 \dots U_k)$ *i.e.*, the permutations λ_i is identical permutation for $i = 1, 2, \dots, k$.
- For the node sequence $\vec{u} = (U_1\ U_2 \dots U_k)$, the surviving sequence is $\vec{s} = (S_\ell(1)\ S_\ell(2) \dots S_\ell(k))$, where index $\ell \in \{1, 2, \dots, \tau_j\}$ of each surviving set may not be same.

For step label 1 to step label k, each step is associated to a node failure according to the node sequence and the failed node will repair using the respective surviving set in the surviving sequence. In this case, the step label i is associated with the failed node U_i and the failed node will be repaired using the surviving set $S_\ell(i)$. Note that, at step label i, the total

$$\sum_{j \in J \setminus \{1,2,\dots,i-1\}} \beta(i, j, \ell) \tag{4.1}$$

packets are downloaded from those nodes $U_j \in S_\ell(i)$ which are not failed in any earlier steps. Hence,

$$\sum_{j \in J \setminus \{1,2,\dots,i-1\}} w(Out_j, In_i') = \sum_{j \in J \setminus \{1,2,\dots,i-1\}} \beta(i, j, \ell) \tag{4.2}$$

A file with size M is stored in n distinct nodes on heterogeneous DSS. In the

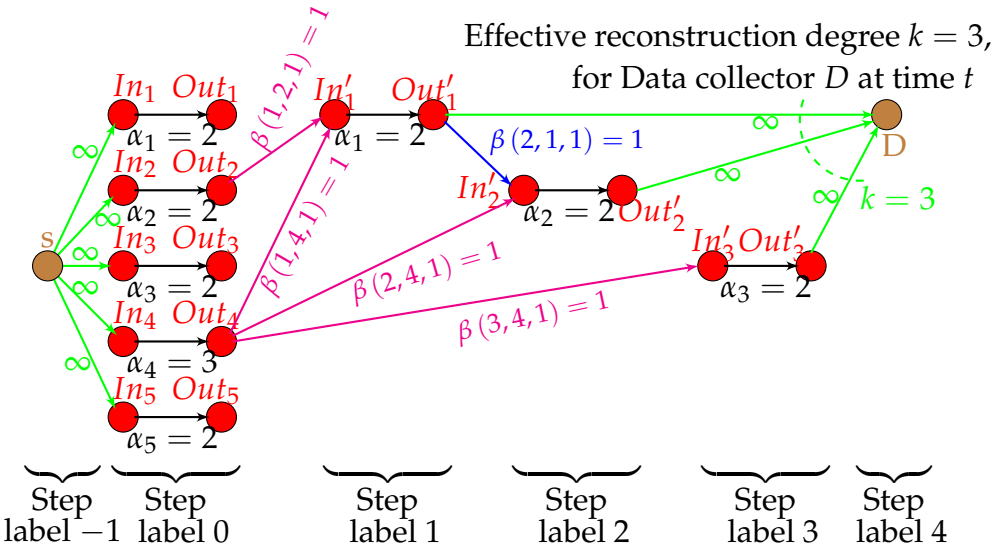

Figure 4.2: An information flow graph for a heterogeneous DSS (Figure 2.3).

associated information flow graph, source node s is connected with each vertex In_i in step label 0. The connection is represented with edge (s, In_i) with $w(s, In_i) \to \infty$ (see Figure 4.1).

A data collector D connects all the k nodes of the set $\{U'_1, U'_2, \ldots, U'_k\}$ and get the complete file information M. In the graph as in Figure 4.1, data collector D connects nodes Out'_i $(i = 1, 2, \ldots, k)$ from step label 1 to step label k and downloads the complete data file. Therefore, consider $(Out'_j, D) \in \mathcal{E}$ with $w(Out'_j, D) \to \infty$.

An information flow graph for a heterogeneous DSS (as discussed in Figure 2.3), is described in Figure 4.2. In particular, a data collector is connected with the nodes of $\mathcal{A}_1 = \{U_1, U_2, U_3\}$. In the information flow graph, if the nodes are failed then it will be repaired by nodes of $S_1(1), S_1(2)$ and $S_1(3)$ respectively. For a heterogeneous DSS (as considered in Figure 2.3), an information flow graph is shown in Figure 4.2 for a specifics data collector D connected with the nodes of $\mathcal{A}$=$\{U_1, U_2, U_3\}$. The particular information flow graph is plotted for the surviving sequence $(S_1(1)\ S_1(2)\ S_1(3)) \in \mathscr{S}\ ((U_1\ U_2\ U_3))$. Recall a surviving sequence is a sequence of surviving sets of distinct nodes and $\mathscr{S}$ is a set of all such surviving sequences. Formally, surviving sequence and surviving sequence set are defined in the Definitions 2.4 and 2.5.

For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, if $\mathcal{X} \cup \overline{\mathcal{X}} = \mathcal{V}$ then the $cut(\mathcal{X}, \overline{\mathcal{X}})$ represents the set

of all edges having one end vertex in set $\mathcal{X}$ and other vertex in set $\overline{\mathcal{X}}$ such that removing those all edges will improve the number of components in the graph $\mathcal{G}$. For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$ is the sum of capacity of all edges in $cut(\mathcal{X}, \overline{\mathcal{X}})$.

Consider the simple case (see Figure 4.1), where, at time t, the reconstruction set $\mathcal{A} = \{U_1, U_2, \ldots, U_k\}$, the node sequence $\vec{u} = (U_1\ U_2 \ldots U_k)$, the surviving sequence $\vec{s} = (S_\ell(1)\ S_\ell(3) \ldots S_\ell(k))$ and, for given $i = 1, 2, \ldots, k$, $S_\ell(i) = \{U_j : j \in J$ for some $J \subset \{1, 2, \ldots, n\} \backslash \{i\}\}$. For the case, at the step label i, from Equation 4.2, the contribution in min *cut-capacity*(s, D) is

$$\begin{aligned} &\min \left\{ \left(\sum_{j \in J \backslash \{1,2,\ldots,i-1\}} w(Out_j, In_i') \right), w(In_i, Out_i') \right\} \\ &= \min \left\{ \left(\sum_{j \in J \backslash \{1,2,\ldots,i-1\}} \beta(i, j, \ell) \right), \alpha_i \right\}. \end{aligned} \tag{4.3}$$

Therefore, for the given $\mathcal{G}$ (Figure 4.1),

$$\min \textit{cut-capacity}(s, D) = \sum_{i=1}^{k} \min \left\{ \left(\sum_{j \in J \backslash \{1,2,\ldots,i-1\}} \beta(i, j, \ell) \right), \alpha_i \right\}. \tag{4.4}$$

In [122, 119, 14, 29], *min-cut* bound is calculated by analyzing flow passing from source node s to data collector node D across the information flow graph for a DSS. Hence one can define flow across the information flow graph as follows.

Definition 4.1. *(Information Flow): A function $f : \mathcal{E} \to [0, \infty) \subset \mathbb{R}$ is called information flow or simply flow on an information flow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if*

1. *(capacity constraint): $\forall (x, y) \in \mathcal{E}$, $f((x, y)) \leq c((x, y))$, where $c((x, y)) = w(x, y)$ and $c((x, y))$ is capacity of edge (x, y) and*

2. *(flow conservation constraint:) $\forall y \in \mathcal{V} \backslash \{s, t\}$,*

$$\sum_{\substack{x \\ (x,y) \in \mathcal{E}}} f((x, y)) = \sum_{\substack{z \\ (y,z) \in \mathcal{E}}} f((y, z)). \tag{4.5}$$

For more details and examples on flow function, see [2, 27].

For a given information flow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, value of flow delivered to a data collector node D is defined as total amount of flow passes through the edges $(x, D) \in \mathcal{E}$ for all possible $x \in \mathcal{V}$.

In general, at time t, for a specific data collector D which connects nodes U_{λ_i} of set $\mathcal{A}_t \in \mathscr{A}$, has $|\mathcal{A}_t|! \prod_{i=1}^{|\mathcal{A}_t|} \tau_{\lambda_i}$ number of distinct information flow graphs are exist. Therefore,

$$\min_{\mathcal{G}} \max\text{-}flow(s, D) = \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \min_{\vec{s} \in \mathscr{S}(\vec{u})} \max\text{-}flow(s, D). \tag{4.6}$$

Hence, for the heterogeneous DSS at any time t,

$$\min_{t} \min_{\mathcal{G}} \max\text{-}flow(s, D) = \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \min_{\vec{s} \in \mathscr{S}(\vec{u})} \max\text{-}flow(s, D). \tag{4.7}$$

In this equation (Equation (4.7)), the minimum value for max-$flow(s, D)$ is calculated for information flow graphs which are obtained for all combinations of all reconstruction sets, node sequences, and respective surviving sets. At any time t, for every information flow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, D can recover the whole file M so

$$M \leq \min_{t} \min_{\mathcal{G}} \max\text{-}flow(s, D) = \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \min_{\vec{s} \in \mathscr{S}(\vec{u})} \max\text{-}flow(s, D). \tag{4.8}$$

In networks, maximum possible value of flow delivered to D, is governed by *min-cut max-flow* theorem [2, 27, 30]. *Min-cut max-flow* theorem says that across the network, maximum possible value of flow passes from source s to specific data collector D (denoted by $max\text{-}flow(s, D)$), is equal to minimum $cut\text{-}capacity(s, D)$, where

$$\min cut\text{-}capacity(s, D) = \min_{\substack{cut(\mathcal{X}, \overline{\mathcal{X}});\\ s \in \mathcal{X}, D \in \overline{\mathcal{X}};\\ \mathcal{X} \cup \overline{\mathcal{X}} = \mathcal{V}.}} \left\{ cut\text{-}capacity(\mathcal{X}, \overline{\mathcal{X}}) \right\}. \tag{4.9}$$

Therefore, from the Inequality (4.8),

$$M \leq \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \min_{\vec{s} \in \mathscr{S}(\vec{u})} cut\text{-}capacity(s, D). \tag{4.10}$$

4.1.2 Optimization problem for heterogeneous DSSs

In [119], for an information flow graph, flow analysis is done by taking a topological order of failed node connected with the data collector. For the heterogeneous DSS, it is shown that minimum possible value of flexible reconstruction degree is the lower bound of the cardinality of any *cut set* which separates source node and data collector node D. For the heterogeneous DSS, *min cut* bound is calculated in Theorem 4.1. Using that *min cut* bound, it is shown that the file size should be lower bound of *min cut* bound for the heterogeneous DSS. A bi-objective optimization linear programming problem with the constraint as *min cut* bound, is formulated to minimize system storage cost and system repair cost for the considered heterogeneous DSS. A family of solutions is calculated for the optimization problem by substituting some numerical values of system parameters. The numerical parameter is plotted the trade-off curve between system storage cost and system repair cost. The curve is compared with the trade-off curve for homogeneous DSS [22] and trade-off curve for heterogeneous DSS [122].

In a heterogeneous DSS, information delivered to data collector D depends on the min *cut-capacity*(s, D). For such heterogeneous DSS, the following theorem gives the bound on min *cut-capacity*(s, D).

Theorem 4.1. *(min-cut bound) Consider a heterogeneous DSS with n nodes. At time t, suppose a data collector D connects the nodes of a reconstruction set $\mathcal{A}_t \in \mathscr{A}$. If nodes of the reconstruction set $\mathcal{A}$ fail in order of the node sequence $\vec{u} = (U_{\lambda_1}\, U_{\lambda_2} \dots U_{\lambda_{|\mathcal{A}|}}) \in \mathscr{A}(\mathcal{A})$ and those failed nodes are repaired in order of the surviving sequence $\vec{s} = (S_{\ell_1}(\lambda_1)\, S_{\ell_2}(\lambda_2) \dots$ $\dots S_{\ell_{|\mathcal{A}|}}(\lambda_{|\mathcal{A}|})) \in \mathscr{S}$, then, for any reconstruction set, node sequence sequence and surviving sequence, the* min *cut-capacity*(s, D)

$$\geq \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \sum_{i=1}^{|\mathcal{A}_t|} \min \left\{ \left(\min_{\vec{s} \in \mathscr{S}(\vec{u})} \sum_{\substack{j \\ U_j \in S_{\ell_i}(\lambda_i) \setminus \{U_{\lambda_1}, \dots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\}, \tag{4.11}$$

where node storage capacity of node U_{λ_i} is α_{λ_i}, and $\beta\left(\lambda_i, j, \ell_i\right)$ packets are downloaded from the node $U_j \in S_{\ell_i}(\lambda_i)$ for the repair the failed node U_i.

Proof. Consider a heterogeneous DSS (n, k, d) associated with some information flow graphs. Every information flow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ has a source node s, a data collector node D associated with effective reconstruction degree k_t. In the heterogeneous DSS, a failed node U_i can be repaired by nodes of some surviving set $S_\ell(i)$, where $\ell \in \{1, 2, \ldots, \tau_i\}$.

Let $\mathcal{X} \subset \mathcal{V}$, $\mathcal{X} \cup \overline{\mathcal{X}} = \mathcal{V}$, $s \in \mathcal{X}$ and $D \in \overline{\mathcal{X}}$ such that some nonempty subset $cut(\mathcal{X}, \overline{\mathcal{X}}) \subset \mathcal{E}$ exist. Now if $\mathcal{X} = \mathcal{V} \backslash \{D\}$ then $cut\text{-}capacity(\mathcal{X}, \overline{\mathcal{X}}) \to \infty$. Similarly if $\mathcal{X} = \{s\}$ then again $cut\text{-}capacity(\mathcal{X}, \overline{\mathcal{X}}) \to \infty$. Hence, $\min cut\text{-}capacity(\mathcal{X}, \overline{\mathcal{X}})$ would be obtained by all those $Out'_j \in \overline{\mathcal{X}}$ and $In_i \in \mathcal{X}$, since it will give a finite $cut\text{-}capacity(\mathcal{X}, \overline{\mathcal{X}})$, where $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, k_t\}$.

Information flow graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is directed acyclic graph so that it can be represented in topological order of its vertices. For the topological order, sequences of node failure and a corresponding sequence of surviving sets are arranged by using definitions as given in the previous section. For that assume at time t, data collector D connects with all nodes of a set $\mathcal{A}_t \in \mathscr{A}$ and reconstruct the file M. $\mathscr{A}(\mathcal{A}_t)$ is the set of all possible sequences of nodes of $\mathcal{A}_t \in \mathscr{A}$. A sequence $\left(U_{\lambda_1} U_{\lambda_2} \ldots U_{\lambda_{k_t}}\right)$ represents the order of nodes failure from specific set $\mathcal{A}_t$. Recall the set of all possible surviving sequences $\left(S_{\ell_1}(\lambda_1)\, S_{\ell_2}(\lambda_2) \ldots S_{\ell_k}(\lambda_{k_t})\right)$ associated with a node sequence $\left(U_{\lambda_1} U_{\lambda_2} \ldots U_{\lambda_{k_t}}\right)$, is $\mathscr{S}\left(\left(U_{\lambda_1} U_{\lambda_2} \ldots U_{\lambda_{k_t}}\right)\right)$, where $|\mathcal{A}_t| = k_t$.

For a specific surviving sequence $\vec{s} = \left(S_{\ell_1}(\lambda_1)\, S_{\ell_2}(\lambda_2) \ldots S_{\ell_k}(\lambda_{k_t})\right)$ associated with the node sequence $\vec{u} = \left(U_{\lambda_1} U_{\lambda_2} \ldots U_{\lambda_{k_t}}\right)$, the following observations are possible.

For $Out'_{\lambda_1} \in \overline{\mathcal{X}}$ associated with the first node in node sequence $\vec{u}$, the following two cases are possible.

- If $In'_{\lambda_1} \in \mathcal{X}$ then edge $(In'_{\lambda_1}, Out'_{\lambda_1}) \in cut(\mathcal{X}, \overline{\mathcal{X}})$. Hence, α_{λ_1} will contribute in $cut\text{-}capacity(\mathcal{X}, \overline{\mathcal{X}})$.

- If $In'_{\lambda_1} \in \overline{\mathcal{X}}$ then edges $(Out_j, In'_{\lambda_1}) \in cut(\mathcal{X}, \overline{\mathcal{X}})$, where $U_j \in S_{\ell_1}(\lambda_1)$ and $S_{\ell_1}(\lambda_1) \in \vec{s}$ for any $\ell_1 \in \{1, 2, \ldots, \tau_{\lambda_1}\}$. Hence, this case contributes in

cut-capacity$(\mathcal{X}, \overline{\mathcal{X}})$ by

$$\sum_{\substack{j \\ U_j \in S_{\ell_1}(\lambda_1)}} \beta\left(\lambda_1, j, \ell_1\right). \tag{4.12}$$

So, contribution in min *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$ supported by node U_{λ_1}, is

$$\min\left\{\left(\sum_{\substack{j \\ U_j \in S_{\ell_1}(\lambda_1)}} \beta\left(\lambda_1, j, \ell_1\right)\right), \alpha_{\lambda_1}\right\}. \tag{4.13}$$

In general, contribution in min *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$ is supported by node $U_{\lambda_i} \in \vec{u}$. If $Out'_{\lambda_i} \in \overline{\mathcal{X}}$ then following two cases are possible.

- If $In'_{\lambda_i} \in \mathcal{X}$, then edge $(In'_{\lambda_i}, Out'_{\lambda_i}) \in cut(\mathcal{X}, \overline{\mathcal{X}})$. Hence α_{λ_i} will contribute in *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$.

- If $In'_{\lambda_i} \in \overline{\mathcal{X}}$ then all those edges (Out_j, In'_{λ_i}) will contribute in $cut(\mathcal{X}, \overline{\mathcal{X}})$ which are associated with $U_j \in S_{\ell_i}(\lambda_i) \backslash \{U_{\lambda_1}, U_{\lambda_2}, \ldots, U_{\lambda_{i-1}}\}$ for surviving set $S_{\ell_i}(\lambda_i) \in \vec{s}$, where $\ell_i \in \{1, 2, \ldots, \tau_{\lambda_i}\}$. Edges $(Out_{\lambda_j}, In'_{\lambda_i})$ associated with node $U_j \in S_{\ell_i}(\lambda_i) \backslash \{U_{\lambda_1}, U_{\lambda_2}, \ldots, U_{\lambda_{i-1}}\}$ are newly investigated from step label 0 for $cut(\mathcal{X}, \overline{\mathcal{X}})$. Edges $(Out'_{\lambda_m}, In'_{\lambda_i})$ must be excluded because they have investigated earlier at step label m, where $m \in \{1, 2, \ldots, i-1\}$ *s.t.* $U_{\lambda_m} \in S_\ell(\lambda_i)$. Hence this case contributes in *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$ by

$$\sum_{\substack{j \\ U_j \in S_{\ell_i}(\lambda_i) \backslash \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right). \tag{4.14}$$

So, contribution in min *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$ by node U_{λ_i} is

$$\min\left\{\left(\sum_{\substack{j \\ U_j \in S_{\ell_i}(\lambda_i) \backslash \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right)\right), \alpha_{\lambda_i}\right\}. \tag{4.15}$$

At the time instant t, if data collector D connects with each nodes $U_{\lambda_i} \in \mathcal{A}_t$ ($i \in \{1, 2, \ldots, k_t\}$) then the contribution in min *cut-capacity*$(\mathcal{X}, \overline{\mathcal{X}})$ for a specific

node sequence $\vec{u}$ associated with a specific surviving sequence $\vec{s}$, is

$$\sum_{i=1}^{k_t} \min \left\{ \left(\sum_{\substack{j \\ U_j \in S_{\ell_i}(\lambda_i) \setminus \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\}. \tag{4.16}$$

Therefore,

$$\begin{aligned}
&\min \textit{cut-capacity}(s, D) \\
&\geq \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \min_{\vec{s} \in \mathscr{S}(\vec{u})} \sum_{i=1}^{k_t} \min \left\{ \left(\sum_{\substack{j \\ U_j \in S_{\ell_i}(\lambda_i) \setminus \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\}.
\end{aligned}$$

But, for a node sequence $\vec{u}$,

$$\begin{aligned}
&\min_{\vec{s} \in \mathscr{S}(\vec{u})} \sum_{i=1}^{k_t} \min \left\{ \left(\sum_{\substack{j \\ U_j \in S_{\ell}(\lambda_i) \setminus \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\} \\
&= \sum_{i=1}^{k_t} \min \left\{ \left(\min_{\vec{s} \in \mathscr{S}(\vec{u})} \sum_{\substack{j \\ U_j \in S_{\ell}(\lambda_i) \setminus \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\}.
\end{aligned}$$

Hence,

$$\begin{aligned}
&\min \textit{cut-capacity}(s, D) \\
&\geq \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \sum_{i=1}^{k_t} \min \left\{ \left(\min_{\vec{s} \in \mathscr{S}(\vec{u})} \sum_{\substack{j \\ U_j \in S_{\ell}(\lambda_i) \setminus \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\}.
\end{aligned}$$

The *min-cut* bound is calculated for all possible node sequences $\vec{u}$ associated with all possible surviving sequences $\vec{u}$. Hence there exist at least one surviving sequences, say, $\vec{s}^*$ associated with node sequence, say, $\vec{u}^*$ for which the inequality holds with equality *i.e.*, the *min-cut* bound Inequality (4.11) is tight. □

Remark 4.1. *For a given heterogeneous DSS, at time t, if an arbitrary data collector*

connects each node U_{λ_j} in subset $\mathcal{A}_t \in \mathscr{A}$ then total number of possible information flow graphs are given by

$$\sum_{\substack{\mathcal{A}_t \\ \mathcal{A}_t \in \mathscr{A}}} \left(|\mathcal{A}_t| \,! \prod_{j=1}^{|\mathcal{A}_t|} \tau_{\lambda_j} \right).$$

In particular, for a specific information flow graph, the total number of computational comparisons are $2|\mathcal{A}_t|$. Hence One can say that the time complexity to calculate min-cut bound is

$$\mathcal{O}\left(\sum_{\substack{\mathcal{A}_t \\ \mathcal{A}_t \in \mathscr{A}}} \left(2|\mathcal{A}_t|(|\mathcal{A}_t| \,!) \prod_{j=1}^{|\mathcal{A}_t|} \tau_{\lambda_j} \right) \right).$$

By Theorem 4.1 one can calculate the minimum requirement of storage node capacity and repair bandwidth to store a file with size M. In other words, the upper bound of the stored file with size M is given by the following lemma.

Theorem 4.2 (Fundamental Bound)**.** *Let a file with size M be stored in a heterogeneous DSS with n nodes. At time t, suppose a data collector D connects the nodes of a reconstruction set $\mathcal{A}_t \in \mathscr{A}$. If nodes of the reconstruction set $\mathcal{A}$ fail in order of the node sequence $\vec{u} = (U_{\lambda_1} U_{\lambda_2} \dots U_{\lambda_{|\mathcal{A}|}}) \in \mathscr{A}(\mathcal{A})$ and those failed nodes are repaired in order of the surviving sequence $\vec{s} = (S_{\ell_1}(\lambda_1)\, S_{\ell_2}(\lambda_2) \dots S_{\ell_{|\mathcal{A}|}}(\lambda_{|\mathcal{A}|})) \in \mathscr{S}$, then, for any reconstruction set, node sequence sequence and surviving sequence, the file size*

$$M \leq \min_{\mathcal{A}_t \in \mathscr{A}} \min_{\vec{u} \in \mathscr{A}(\mathcal{A}_t)} \sum_{i=1}^{|\mathcal{A}_t|} \min \left\{ \left(\min_{\vec{s} \in \mathscr{S}(\vec{u})} \sum_{\substack{j \\ U_j \in S_{\ell_i}(\lambda_i) \setminus \{U_{\lambda_1}, \dots, U_{\lambda_{i-1}}\}}} \beta\left(\lambda_i, j, \ell_i\right) \right), \alpha_{\lambda_i} \right\}, \tag{4.17}$$

where node storage capacity of node U_{λ_i} ($i = 1, 2, \dots |\mathcal{A}|$) is α_{λ_i}, and $\beta\left(\lambda_i, j, \ell_i\right)$ packets are downloaded from the node $U_j \in S_{\ell_i}(\lambda_i)$ for the repair the failed node U_i.

Proof. Any arbitrary data collector D must be able to reconstruct the whole file with size M. Hence the maximum information flow value delivered to any data collector should be at least M. Now using *min-cut max-flow* theorem and Theorem 4.1 one can prove the lemma. □

The min *cut-capacity*(s, D) for the information flow graph as shown in Figure

4.2, will be $\min\{\alpha_1, \beta(1,2,1) + \beta(1,4,1)\} + \min\{\alpha_2, \beta(2,4,1)\} + \min\{\alpha_3, \beta(3,4,1)\}$ $= 2 + 1 + 2 = 5$ units. Note, the $\min cut\text{-}capacity(s, D) = 5 > M = 4$.

Remark 4.2. *The Fundamental Bound given in Theorem 4.2 is tight. The bound is calculated by taking minimum cut-capacity for various information flow graphs on a given DSS. Therefore, there exists at-least one combination of reconstruction set, node sequence, and surviving sequence such that it satisfies the bound with equality.*

Now, one can frame a optimization problem to find minimum system storage cost and system repair cost under the constraint that the maximum possible information deliver to data collector node D is at lest M. The fundamental bound on file size M (Inequality (4.17)) will be the constraint for the optimization problem.

Problem 4.1.

$$\begin{aligned} &\textit{Minimize: } [C_s(\vec{\alpha}), C_r(\vec{\beta})] \\ &\textit{subject to} \\ &\qquad \textit{Inequality } (4.17); \\ &\qquad\qquad \alpha_i \geq 0; \\ &\qquad\qquad \beta(i,j,\ell) \geq 0; \end{aligned}$$

where $i \in \{1, 2, \ldots, n\}$, $\ell \in \{1, 2, \ldots, \tau_i\}$ *and* $U_j \in S_\ell(i)$ *for* $j \in \{1, 2, \ldots, n\} \backslash \{i\}$.

Optimum values for the both objective functions of bi-objective optimization Problem 4.1, are plotted as trade-off curve between $C_s(\vec{\alpha})$ and $C_r(\vec{\beta})$. In this chapter, the optimization Problem 4.1 is solved by a weighted sum method for some numeric example.

Some specific cases for optimization Problem 4.1 are analyzed in the following subsection.

4.1.3 Optimization problem for specific heterogeneous DSSs

Given heterogeneous DSS can be reduced to following cases under some specific restrictions. The cases are as follows:

1) (Uniform Reconstruction): At time t, if an arbitrary data collector can retrieve the file by downloading data from exactly k nodes for any combination out of n nodes then the constraint Inequality (4.17) for the optimization Problem 4.1 has additional property $k_t = k, \forall t$.

2) (Uniform Repair Degree): For a heterogeneous DSS, if a failed node can be repaired by *any* d nodes out of remaining $n-1$ nodes. Under the particular assumption, for a reconstruction set $\mathcal{A}_t$, nodes are failed in order of the node sequence $\vec{u} = (U_{\lambda_1}\ U_{\lambda_2} \dots U_{\lambda_{|\mathcal{A}|}}) \in \mathscr{A}(\mathcal{A})$ and those failed nodes are repaired in order of the surviving sequence $\vec{s} = (S_{\ell_1}(\lambda_1)\ S_{\ell_2}(\lambda_2) \dots S_{\ell_{|\mathcal{A}|}}(\lambda_{|\mathcal{A}|})) \in \mathscr{S}$, where $S_{\ell_1}(\lambda_1) \subseteq S_{\ell_2}(\lambda_2) \subseteq \dots \subseteq S_{\ell_{|\mathcal{A}|}}(\lambda_{|\mathcal{A}|})$. In this case $|S_{\ell_i}(m)| = d$, $\tau_m = \binom{n-1}{d}, \forall m \in \{1, 2, \dots, n\}$. Here $\min cut\text{-}capacity(s, D)$ will be given by the node sequence $\left(U_{\lambda_1}\ U_{\lambda_2} \dots U_{\lambda_{|\mathcal{A}_t|}}\right) \in \mathscr{A}(\mathcal{A}_t)$ associated with surviving sequence $\left(S_\ell(\lambda_1)\ S_\ell(\lambda_2) \dots S_\ell(\lambda_{|\mathcal{A}_t|})\right)$ such that $\alpha_{\lambda_1} \leq \alpha_{\lambda_2} \leq \dots \leq \alpha_{\lambda_{|\mathcal{A}_t|}}$ and $\{U_{\lambda_1}, U_{\lambda_2}, \dots, U_{\lambda_{i-1}}\} \subset S^{(\ell)}_{\lambda_i}$. Therefore, for the case of uniform repair degree, the optimization problem Problem 4.1 reduced to

Problem 4.2.

$$
\begin{aligned}
&\textit{Minimize: } [C_s(\vec{\alpha}), C_r(\vec{\beta})] \\
&\textit{subject to} \\
&M \leq \min_{\mathcal{A}_t \in \mathscr{A}} \sum_{\substack{i=1; \\ U_{\lambda_i} \in \mathcal{A}_t}}^{|\mathcal{A}_t|} \min \left\{ \alpha_{\lambda_i}, \sum_j \beta\left(\lambda_i, j, \ell_i\right) \right\}; \qquad (4.18) \\
&0 \leq \alpha_1 \leq \alpha_2 \leq \dots \leq \alpha_n; \\
&1 \leq \lambda_1 < \lambda_2 < \dots < \lambda_{|\mathcal{A}_t|} \leq n;
\end{aligned}
$$

where index j is the index of node $U_j \in S_{\ell_i}(\lambda_i) \backslash S_{\ell_{i-1}}(\lambda_{i-1})$ and $S_{\ell_0}(\lambda_0) = \emptyset$.

3) (Uniform Repair Download Amount): In this case, we assume that the downloaded amount from any arbitrary helper node to repair the system is constant say β. Hence, the optimization Problem 4.1 under the restriction has additional properties as $\beta\left(i, j, \ell\right) = \beta$, $\beta \geq 0$ ($\forall i \in \{1, 2, \dots, n\}$, all possible $j \in \{1, 2, \dots, n\} \backslash \{i\}$ and

$\forall \ell \in \{1, 2, \ldots, \tau_i\}$). Hence,

$$\sum_j \beta\left(\lambda_i, j, \ell_i\right) = \beta \sum_j |S_{\ell_i}(\lambda_i) \backslash \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}|,$$

where j's are the indexes of those nodes $U_j \in S_{\ell_i}(\lambda_i) \backslash \{U_{\lambda_1}, \ldots, U_{\lambda_{i-1}}\}$.

4) (Homogenous DSS): A heterogeneous DSS become a homogeneous DSS if all the parameters are uniform. Hence assume effective reconstruction degree for any data collector is k and storage capacity of each node is α. In addition let a node failure can repair by *any* d nodes out of remaining $n-1$ nodes by downloading β packets from each helper node. Under these restrictions, the constraint Inequalities (4.17) for the optimization Problem 4.1 reduced to

Problem 4.3.

$$\begin{aligned} &\textit{Minimize: } [C_s(\vec{\alpha}), C_r(\vec{\beta})] \\ &\textit{subject to} \\ &\qquad M \leq \sum_{i=1}^{k} \min\left\{\alpha, (d-i-1)\,\beta\right\}; \\ &\qquad \alpha \geq 0; \\ &\qquad \beta \geq 0. \end{aligned}$$

5) (Other): In this chapter, the considered heterogeneous DSS model can be reduced into some more specific DSS by applying some appropriate restrictions on constraints. For example, heterogeneous DSS with uniform reconstruction and uniform repair degree (case 1 and 2 respectively) collectively reduce to heterogeneous DSS as investigated in [122].

One can easily find the solution of the bi-objective optimization Problem 4.1 for some numerical values and plot the solution as a trade-off curve for the same. One can compare the trade-off curve with the trade-off curve for the existing heterogeneous DSS investigated in [122]. Hence in the next section we are calculating some optimum solutions for the numerical parameter for our model and comparing it with homogeneous model [119] and heterogeneous model [122].

Remark 4.3. *The trade-off curve between system repair cost and storage cost can be plotted*

by optimization Problem 4.1 for the exact repair if all the surviving sets are the collection of those helper nodes which are associated with exact repair of the failed nodes.

4.1.4 Trade-off for heterogeneous DSSs

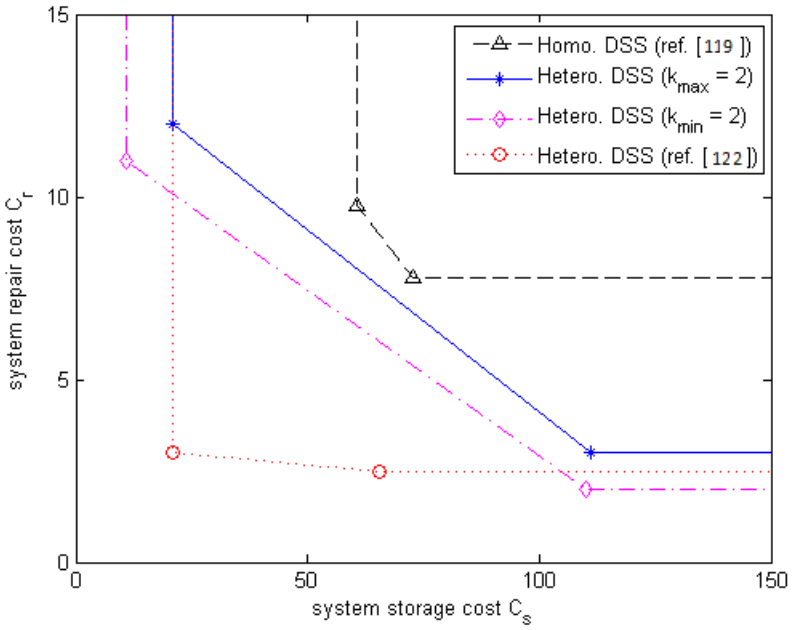

Figure 4.3: For various DSSs (considered in this work, [23, 122]), the optimal trade-off curves are plotted between system repair cost C_r and system storage cost C_s.

For the optimization Problem 4.1, linear programming problems with single objective function are solved. The single objective function is calculated by taking a linear combination of the two objective functions of the optimization problem 4.1. Then such linear programming problems are solved by taking a distinct linear combination factor between 10^{-3} and 10^{3}. Plotting trade-off and solving LP problems are done with the help of 'MATLAB,' and 'lp_solve' [18].

In Figure 4.3, four trade-off curves are plotted between system repair cost C_r and system storage cost C_s for the respective DSSs. In particular Figure 4.3, one curve is plotted for homogeneous DSS as investigated in [119], another one is drawn for a heterogeneous DSS as investigated in [122] and remaining two curves are plotted for two heterogeneous DSSs as studied in this chapter. In particular, one of the remaining two curves has minimum effective reconstruction degree $k_{\min}$ as 2 and other has maximum effective reconstruction degree $k_{\max}$ as 2. For all considered

DSSs the common parameters are as follow: $n = 4$, $M = 1$ unit, $\vec{s} = (1\ 10\ 10\ 100)$ and $\vec{r} = (10\ 1\ 1\ 1)$. For homogeneous DSS and heterogeneous DSS studied in [122] have reconstruction degree $k = 2$ and repair degree $d = 3$. Remaining both heterogeneous DSS have surviving sets $S_1(1) = \{U_2, U_3, U_4\}$, $S_1(2) = \{U_1, U_4\}$, $S_1(3) = \{U_1, U_2\}$, and $S_1(4) = \{U_2, U_3\}$.

In Figure 4.3, one can see that our heterogeneous DSS model has more optimum system storage and repair cost then the homogeneous DSS studied in [119]. Although the characteristics of our heterogeneous model and heterogeneous model investigated in [122] are different, but we obtained some more optimum points for our model as in Figure 4.3. It is shown in the last subsection that one can find heterogeneous DSS considered in [122] by taking some restrictions on our model.

Remark 4.4. *In the particular trade-off curves, non-integer solution of bi-objective optimization Problem 4.1 is also considered. Since the scaling of an arbitrary file size M to 1 leads to respective integer solution that is not necessarily scaled to some integer.*

4.2 Analysis of the Fundamental Bound

For heterogeneous DSS, the complexity of computing the fundamental bound (Inequality (4.17)) is high. To calculate codes achieving the fundamental bound, we consider a specific case of the heterogeneous DSS. In the next two sections, we described the heterogeneous DSS with uniform reconstruction degree k and repair traffic β, and, further, analyze the fundamental bound to obtain some relations for the optimal codes for the heterogeneous DSS.

4.2.1 Heterogeneous DSS Model

In heterogeneous DSS, a file is divided into encoded packets and the encoded packets are distributed among n distinct nodes U_i $(i = 1, 2, \ldots, n)$ such that each node has storage capacity α_i and repair degree d_i. A user can reconstruct the file by downloading data from any k $(< n)$ nodes but not less than k nodes. If a node U_i fails then data collector will download β packets from specific d_i nodes out of remaining $n - 1$ nodes. The particular d_i nodes are called helper nodes

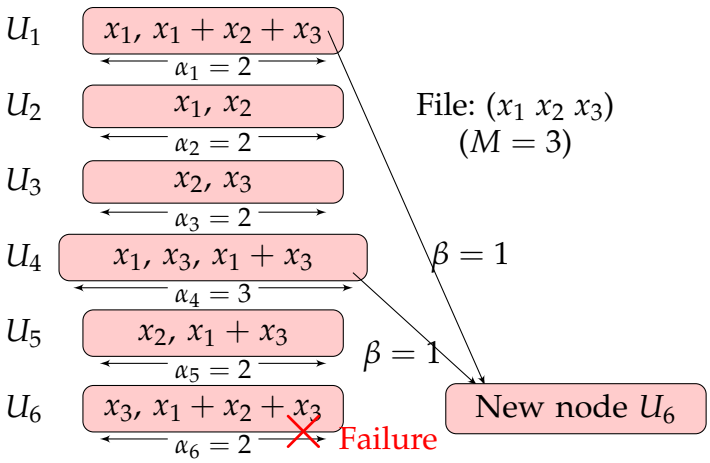

Figure 4.4: An example of a $(6,2)$ heterogeneous DSS, where repair bandwidth is proportional to repair degree.

for the failed node U_i. In this model, $\beta(i,j,\ell)=\beta$ for each $i,j\in\{1,2,\ldots,n\}$ and $\ell\in\{1,2,\ldots,\tau_i\}$. In this case, repair bandwidth for a node U_i is $\gamma_i=d_i\beta$.

An example of such a heterogeneous DSS is illustrated in Figure 4.4. In this example, A file is divided into 3 $(=M)$ distinct coded packets x_1, x_2 and x_3 on field $\mathbb{F}_q$. These three packets are encoded into thirteen distinct packets and distributed in $(6,2)$ heterogeneous DSS. For the heterogeneous DSS, the repair traffic β is 1. In the particular DSS, node storage capacity α_i is $2,2,2,3,2$ and 2 for $i=1,2,3,4,5,6$ (see Figure 4.4). Note that $\alpha_i=\gamma_i=d_i$ for each $i=1,2,3,4,5,6$. Similar to the parameters of the regenerating codes for homogeneous systems [23], we provide the parameters of Heterogeneous Regenerating codes in the next remark.

Remark 4.5. *For a (n,k) heterogeneous DSS, regenerating codes over a field $\mathbb{F}_q$ are described by the parameters $[n,\ k,\ \vec{d},\vec{\alpha},\beta,M]$, where M is the file size, β is the repair traffic, $\vec{d}=(d_1\ d_2\ldots d_n)$ and $\vec{\alpha}=(\alpha_1\ \alpha_2\ldots\alpha_n)$ are one dimensional arrays of repair degree d_i and node storage capacity α_i for node U_i indexed with $i=1,2,\ldots,n$.*

The fundamental bound for the specific heterogeneous DSS, is described in following theorem.

Theorem 4.3 (Fundamental Bound)**.** *For a (n,k) heterogeneous DSS, the file size M must satisfy the Inequality 4.19, where $\{U_{f_0}\}=\emptyset$, $\vec{s}=\left(S_{\ell_{f_1}}(f_1)\ S_{\ell_{f_2}}(f_2)\ldots S_{\ell_{f_n}}(f_n)\right)$, $\mathscr{T}$ is the set of all surviving sequences $\vec{s}$ with length n and $\ell_{f_i}\in\{1,2,...,\tau_{f_i}\}$ for*

$i = 1, 2, \ldots, n$.

$$M \leq \min_{\vec{s} \in \mathscr{T}} \left\{ \sum_{j=1}^{k} \min \left\{ \alpha_{f_j}, \left| S_{\ell_{f_j}}(f_j) \setminus \left(\bigcup_{\lambda=0}^{j-1} \{U_{f_\lambda}\} \right) \right| \beta \right\} \right\} \tag{4.19}$$

In [14], it is shown that there exists code which achieves the fundamental bound for such (n, k) heterogeneous DSS. Hence, one can get the optimal codes by reducing parameters which meet the fundamental bound. In the next section, parameters for the optimal codes are computed by minimizing node storage capacity and repair bandwidth.

4.2.2 Analysis for the Optimal Heterogeneous DSS Model

Consider a (n, k) heterogeneous DSS with τ_i number of surviving sets $S_{\ell_i}(i)$ and repair degree $\left|S_{\ell_i}(i)\right| = d_i$ ($\ell_i = 1, 2, \ldots, \tau_i$ and $i = 1, 2, \ldots, n$). If $\alpha_i > \left|S_{\ell_i}(i)\right| \beta$ then the failed node U_i can not be repaired so $\alpha_i \leq \left|S_{\ell_i}(i)\right| \beta$ for each i and ℓ_i. For optimality, $\alpha_i = d_i \beta$. Hence for constant repair traffic β, node storage capacity α_i and repair degree d_i are proportional to each other. Consider $c_i \in (0, 1) \subset \mathbb{R}$ such that $\sum_{i=1}^{n} c_i = 1$ and $\frac{c_i}{\alpha_i} = \frac{c_j}{\alpha_j}$ for $1 \leq i < j \leq n$. Hence, for each $i \in \{1, 2, \ldots, n\}$, $\alpha_i = c_i \sum_{j=1}^{n} \alpha_j = c_i \alpha^*$, where $\alpha^* = \sum_{j=1}^{n} \alpha_j$. Again, k is reconstruct degree so, $M \leq \sum_{i \in \mathcal{K}} \alpha_i = \sum_{i \in \mathcal{K}} c_i \alpha^*$ for any arbitrary set $\mathcal{K} \subset \{1, 2, ..., n\}$ such that $|\mathcal{K}| = k$. Hence, $M \leq \sum_{i=1}^{k} c_i \alpha^*$ for $c_1 \leq c_2 \leq ... \leq c_n$. For optimum case, one can reduce α^* up to α^*_{min} such that

$$M = \sum_{i=1}^{k} c_i \alpha^*_{min} \Rightarrow \alpha^*_{min} = M \left(\sum_{j=1}^{k} c_j \right)^{-1}, \tag{4.20}$$

For the $(6, 2)$ heterogeneous DSS as given Figure 4.4, $M = 3$, $c_4 = 3/13$ and $c_i = 2/13$ ($i = 1, 2, 3, 5, 6$). Therefore, $\alpha^* = 13$ and $M = 3 \leq \alpha^* \sum_{i=1}^{2} c_i = 4$.

Similarly for a fixed proportional factor α^*_{min}, one can minimize the repair traffic β such that Bound 4.3 holds with equality. For a specific surviving sequence $\vec{s} = \left(S_{\ell_{f_1}}(f_1)\, S_{\ell_{f_2}}(f_2) \ldots S_{\ell_{f_n}}(f_n) \right)$ with sufficient large repair traffic β, the inequality $\alpha_{f_m} \leq \left| S_{\ell_{f_m}}(f_m) \setminus \left(\bigcup_{\lambda=0}^{m-1} \{U_{f_\lambda}\} \right) \right| \beta$ holds for each $m = 1, 2, \ldots, k$. If we choose β

= β_{min} given in Equation 4.22 then β_{min} is the minimum value of repair traffic β which ensures

$$\left| S_{\ell_{f_m}}(f_m) \setminus \left(\bigcup_{\lambda=0}^{m-1} \{U_{f_\lambda}\} \right) \right| \beta_{min} \geq \alpha_{f_m} \tag{4.21}$$

for each f_m of an arbitrary surviving sequence.

$$\beta_{min} = \max_{\vec{s} \in \mathscr{T}} \left\{ \max_{1 \leq m \leq k} \left\{ \alpha_{f_m} \left| S_{\ell_{f_m}}(f_m) \setminus \left(\bigcup_{\lambda=0}^{m-1} \{U_{f_\lambda}\} \right) \right|^{-1} \right\} \right\} \tag{4.22}$$

For the $(6, 2)$ heterogeneous DSS as given Figure 4.4, $M = 3$, $c_4 = 3/13$, $c_i = 2/13$ $(i = 1, 2, 3, 5, 6)$ and $\beta_{min} = 1$.

CHAPTER 5

On Flower Codes

Just living is not enough... one must have sunshine, freedom, and a little flower. *-Hans Christian Andersen* [1]

In this chapter, a general approach for the construction of a GFR code using sequences is introduced. The properties of such GFR codes and universally good Flower codes are also discussed.

5.1 Flower Code: A General Approach

Motivated by the ring construction of GFR codes as given in [15, 38], one can construct a GFR code as follows. First, place all n nodes on a circle and then drop all replicated packets on those nodes one by one. One has to place the packets until all the replicated packets are consumed. It gives us a GFR code since all replicas of each packet are in the system. For example, consider $n = 4$ distinct nodes U_1, U_2, U_3 and U_4, and $\theta = 4$ distinct packets P_1, P_2, P_3 and P_4. The replication factor of those packets are $\rho_1 = 3, \rho_2 = 2, \rho_3 = 2$ and $\rho_4 = 2$. Let $\{P_2, P_1, P_1, P_3, P_4, P_2, P_4, P_1, P_3\}$ be a collection of all the replicas of all four packets. A Flower code can be constructed by selecting packets one by one from the collection and dropping it on nodes one by one, where all the 4 nodes are placed on a circle (see Figure 5.1).

The parts of this chapter are presented in Seventh International Workshop on Signal Design and Its Applications in Communications (IWSDA) 2015, and Sequences and Their Applications (SETA) 2018.

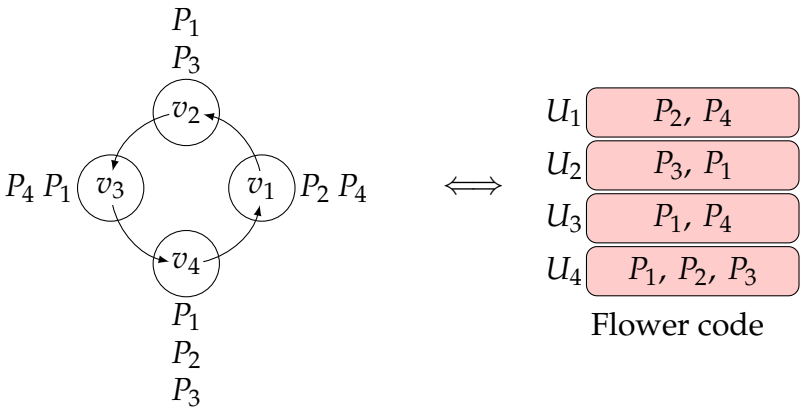

Figure 5.1: An example of a Flower code.

Definition 5.1. *(Flower Code): A system on (n, k)-DSS is called a Flower code in which packets selected from a collection are distributed among n nodes arranged on a circle, where the collection contains all the replicas of each packet.*

Note that, for a Flower code, the replica of each packet may be different.

To construct a Flower code, one can select the copies of coded packets randomly from the collection of replicated packets and he can place it one by one on nodes using some model such as the balls-and-bins model [82]. For the system, one can calculate the probability of receiving the complete file by user and the probability of data recovery when a node has failed. On the other hand, the selection of packets from the collection and dropping the selected packet on nodes can be made using finite binary characteristic sequences. More precisely, for finite binary characteristic sequences $\mathbf{y}$ (of length t) and $\mathbf{x}$ (of length ℓ), if $y_r = 1$ ($r \in \{1, 2, \ldots, t\}$) then the associated packet is selected to be dropped on a node, and if $x_m = 1$ ($m \in \{1, 2, \ldots, \ell\}$) then the selected packet is dropped on the associated node. For a Flower code, both sequences are called the *packet selection sequence* and the *packet dropping sequence* respectively. For the terms $y_j = 1$ and $x_m = 1$, one can select and drop a packet on a node in various methods. In this section, we consider the following.

- If the term $y_r = 1$ then the packet for which the packet-index is congruent to $r \ (mod\ \theta)$ is selected to be dropped on a node.

- If the term $x_m = 1$ then the selected packet is dropped on the node for which

the node-index is congruent to $m \ (mod \ n)$.

For example, Flower code as considered in the Figure 5.1 can be constructed using the two binary sequences $\mathbf{y} = 0100100010110101101$ and $\mathbf{x} = 101101111101$. In the rest of the chapter, for a flower code with n nodes and θ packets, the packet selection sequence of length t is denoted by $\mathbf{y}$ and the packet dropping sequence of length ℓ is denoted by $\mathbf{x}$.

The necessary conditions for the existence of a Flower code for any two binary sequences are given in the next Lemma.

Lemma 5.1. *For n nodes, θ packets, and binary sequences $\mathbf{y}$ and $\mathbf{x}$, a Flower code exists, if*

1. *the weight of both sequences are same i.e., $w_{\mathbf{x}} = w_{\mathbf{y}}$,*
2. *$w_{\mathbf{x}} \geq \theta$ and $w_{\mathbf{y}} \geq n$.*

Proof. The proof of part 1: For some $r \leq t$, if $y_r = 1$ only then a packet is selected to drop on a node so, the weight of the binary sequence $\mathbf{y}$ is the total number of replicated packets which are stored in the system. On the other hand, for some $m \leq \ell$, if $x_m = 1$ only then the selected packet is dropped on a node so, the weight of the binary sequence $\mathbf{x}$ is the total storage capacity of all nodes. However, the packet replica sum and the storage capacity sum are always the same for any system which follows the proof of part 1. Therefore, the Flower code exists, if the weight of both the sequences is the same. The proof of part 2 follows the fact that the total number of replicated packets which are stored in the system can not less then θ and the total storage capacity of all nodes cannot be less than n. □

From the construction of Flower code, one can observe the following Fact.

Fact 5.1. *For some positive integers $m \leq \ell$ and $r \leq t$, consider a Flower code with n nodes, θ packets and two characteristic binary sequences $\mathbf{y}$ and $\mathbf{x}$. For $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, \theta\}$, the packet P_j is dropped on the node U_i if and only if $x_m = 1$, $y_r = 1$ and $w_{\mathbf{x}}(m) = w_{\mathbf{y}}(r)$, where $i - m \equiv 0 \ (mod \ n)$ and $j - r \equiv 0 \ (mod \ \theta)$.*

The parameters of a Flower code are calculated in the following theorem.

Theorem 5.1. *The parameters of a Flower code with n nodes, θ packets and characteristic binary sequences* $\mathbf{y}$ *and* $\mathbf{x}$ *are* $\rho_j = \sum_{p=0}^{\left\lfloor \frac{t-j}{\theta} \right\rfloor} y_{p\theta+j}$ $(j = 1, 2, \ldots, \theta)$ *and* $\alpha_i = \sum_{s=0}^{\left\lfloor \frac{\ell-i}{n} \right\rfloor} x_{sn+i}$ $(i = 1, 2, \ldots, n)$.

Proof. From the Fact 5.1, the Theorem follows from the fact that a packet P_j is selected to drop on some node if and only if $y_{p\theta+j} = 1$ for some $p \in \{0, 1, \ldots, \left\lfloor \frac{t-j}{\theta} \right\rfloor\}$. Similarly, a selected packet is dropped on a node U_i if and only if $x_{sn+i} = 1$ for some $s \in \{0, 1, \ldots, \left\lfloor \frac{\ell-i}{n} \right\rfloor\}$. □

From the Theorem 5.1, one can observe the following propositions.

Proposition 5.1. *For each* $p = 0, 1, \ldots, \left\lfloor \frac{t-j}{\theta} \right\rfloor$, $y_{p\theta+j} = 0$ *if and only if the packet* P_j *does not have any copy in the system, where some* $j \in \{1, 2, \ldots, \theta\}$.

Proposition 5.2. *For each* $s = 0, 1, \ldots, \left\lfloor \frac{\ell-i}{n} \right\rfloor$, $x_{sn+n} = 0$ *if and only if the node* U_i *does not contain any packet, where some* $i \in \{1, 2, \ldots, n\}$.

The following remark gives the constraint on binary sequences for the distribution of multiple copies of a particular packet on a specific node.

Remark 5.1. *For n nodes and θ packets, consider a Flower code with binary sequences* $\mathbf{x}$ *and* $\mathbf{y}$. *For positive integers* $m_1 < m_2 \leq \ell$ *and* $r_1 < r_2 \leq t$, *let* $x_{m_1} = x_{m_2} = 1$ *and* $y_{r_1} = y_{r_2} = 1$ *such that* $w_{\mathbf{x}}(m_1) = w_{\mathbf{y}}(r_1)$ *and* $w_{\mathbf{x}}(m_2) = w_{\mathbf{y}}(r_2)$. *If* $m_1 - m_2 \equiv 0 \pmod{n}$ *and* $r_1 - r_2 \equiv 0 \pmod{\theta}$ *then the two replicas of a particular packet are dropped on a specific node.*

From the construction of GFR code, it is easy to observe that Flower code exists for each GFR code with any set of parameters.

Lemma 5.2. *For any GFR code, a Flower code exists.*

Proof. For a GFR code with n nodes and θ packets, one can always find the collection $S = \{(i, j) :$ Packet P_j is stored in node U_i, $i \in \{1, 2, \ldots, n\}$, $j \in \{1, 2, \ldots, \theta\}\}$ where $|S| = s$. Arrange all the pair of the collection S in a specific order and then extract the sequences of i's and j's. Denote the sequences by $\mathbf{a} = (a_1\ a_2 \ldots a_{|S|})$ and $\mathbf{b} = (b_1\ b_2 \ldots b_{|S|})$ respectively. Note that the sequence $\mathbf{a}$ is defined on

$\{1, 2, \ldots, n\}$ and the sequence $\mathbf{b}$ is defined on $\{1, 2, \ldots, \theta\}$. For any positive integer $z \leq |S|$, the packet P_{b_z} is stored on the node U_{a_z} in the GFR code. For the sequences $\mathbf{a}$ and $\mathbf{b}$, construct binary sequences $\mathbf{x} = 0^{a_1-1}10^{a_2-a_1-1}1 \ldots 10^{a_s-a_{s-1}-1}1$ and $\mathbf{y} = 0^{b_1-1}10^{b_2-b_1-1}1 \ldots 10^{b_s-b_{s-1}-1}1$, where subtractions for the sequences $\mathbf{x}$ and $\mathbf{y}$ are modulo n and modulo θ respectively. The binary sequences $\mathbf{x}$ and $\mathbf{y}$ are finite so assume the length of $\mathbf{x}$ and $\mathbf{y}$ are ℓ and t respectively. For some integers $m \leq \ell$ and $r \leq t$, let $w_x(m) = w_y(r)$, $x_m = 1$ and $y_r = 1$. For the integers m and r, one can fine a positive integer $z \leq s$ such that

$$m = ((a_1 - 1) + 1 + (a_2 - a_1 - 1) + 1 + \ldots + 1 + (a_z - a_{z-1} - 1) + 1) \ (mod\, n) \equiv a_z$$

and

$$r = ((b_1 - 1) + 1 + (b_2 - b_1 - 1) + 1 + \ldots + 1 + (b_z - b_{z-1} - 1) + 1) \ (mod\, \theta) \equiv b_z.$$

Hence, $m - a_z \equiv 0 \ (mod\ n)$ and $m - b_z \equiv 0 \ (mod\ \theta)$. From the Fact 5.1, the packet P_{b_z} is stored in the node U_{a_z} in the Flower code. □

Proposition 5.3. *An arbitrary GFR code is equivalent to a Flower code.*

For given $i \in \{1, 2, \ldots, n\}$, $j \in \{1, 2, \ldots, \theta\}$ and a Flower code with n nodes, θ packets and binary sequences $\mathbf{x}$ and $\mathbf{y}$, let $A_i(j)$ be the number of copies of the packet P_j which are stored in the node U_i in the Flower code. For example, in the Flower code (see Figure 5.1) one copy of the packet P_2 is stored in the node U_1 so, $A_1(2) = 1$. similarly, $A_1(3) = 0$, because packet P_3 is not stored in node U_1. For a Flower code, from the Fact 5.1 and the Theorem 5.1, one can observe the following proposition.

Proposition 5.4. *Consider a Flower code with n nodes, θ packets and binary sequences* $\mathbf{x}$ *and* $\mathbf{y}$*. For given* $i \in \{1, 2, \ldots, n\}$ *and* $j \in \{1, 2, \ldots, \theta\}$*, if* $I_{i,j} = \{r : w_{\mathbf{y}}(r) = w_{\mathbf{x}}(m), r - j \equiv 0 \ (mod\ \theta), m - i \equiv 0 \ (mod\ n)\}$ *then* $A_i(j) = \sum_{r \in I_{i,j}} y_r$*.*

The following Lemma gives the the necessary and sufficient condition for a flower code to be universally good, *i.e.*, to be an universally good GFR code.

Lemma 5.3. *Consider a Flower code with n nodes, θ packets and binary sequences* $\mathbf{x}$ *and* $\mathbf{y}$*. The Flower code is universally good if and only if, for all integers* $1 \leq i \neq p \leq \ell$*,*

$$\sum_{j=1}^{\theta} A_i(j)A_p(j) \leq 1, \tag{5.1}$$

where, $A_i(j)$ is given in the Proposition 5.4.

Proof. From the Fact 5.1, for positive integers $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, \theta\}$, $A_i(j) = 1$ if and only if the packet P_j is dropped on the node U_i. For any $p \in \{1, 2, \ldots, n\}$ such that $p \neq i$, the value $\sum_{j=1}^{\theta} A_i(j)A_p(j)$ is the total number of distinct packets common in both the nodes U_i and U_p. Hence, the Flower code is universally good, if $\sum_{j=1}^{\theta} A_i(j)A_p(j) \leq 1$ for $1 \leq i \neq p \leq n$. □

For the Flower code (see the Figure 5.1) with $n = \theta = 4$, and two binary sequences $\mathbf{y} = 0100100010110101101$ and $\mathbf{x} = 101101111101$, $A_1 = 0101$, $A_2 = 1010$, $A_3 = 1001$ and $A_4 = 1110$. Note that the Flower code is not universally good because $\sum_{j=1}^{4} A_2(j)A_4(j) = 2 \nleq 1$ *i.e.*, node U_2 and node U_4 share 2 distinct packets.

Remark 5.2. *For any integers $1 \leq i < p < s \leq n$, if a binary sequence* $\mathbf{x}$ *satisfies $\sum_{j=1}^{\theta} A_i(j)A_p(j) \leq 1$ and $\sum_{j=1}^{\theta} A_i(j)A_p(j)A_s(j) = 0$ then any two nodes share no more than one packet. Therefore, the Flower code is universally good.*

Remark 5.3. *From the Remark 5.2, it is easy to observe that if a binary sequence* $\mathbf{x}$ *with the entry sum 2θ satisfies $\sum_{j=1}^{\theta} A_i(j)A_p(j) \leq 1$ for any $i, p \in \{1, 2, \ldots, n\}$ and $i \neq p$ then the Flower code is universally good.*

For a Flower code, the maximum file size is calculated in the following Lemma.

Lemma 5.4. *For a universally good Flower code with n nodes, θ packets, and reconstruction degree k, the maximum file size*

$$M(k) = \max\left\{\sum_{i\in I}\sum_{j=1}^{\theta} A_i(j) - \sum_{i,p\in I; i<p}\sum_{j=1}^{\theta} A_i(j)A_p(j) : |I| = k, I \subset \{1, 2, \ldots, n\}\right\}. \tag{5.2}$$

Proof. For a subset $I \subset \{1, 2, \ldots, n\}$, the total number of packets stored in nodes (indexed from I) is $\sum_{i \in I} \sum_{j=1}^{\theta} A_i(j)$. The total number of common packets shared by those nodes is $\sum_{i,p \in I; i<p} \sum_{j=1}^{\theta} A_i(j) A_p(j)$. For universally good Flower code, three or more nodes do not share any packet. Hence, by inclusion exclusion principle, for the reconstruction degree $k = |I|$, the maximum file size is given by the Equation 5.2. □

The following subsection discusses a general construction of the two characteristic binary sequences for a Flower code.

5.1.1 Construction of universally good Flower code

For given positive integers n, θ and z, Algorithm 1 ensures two binary sequences $\mathbf{x}$ and $\mathbf{y}$ for which the Flower code is universally good. For a Flower code, the value z is the count of copies of packets which are placed in the system. In the Algorithm 1, for each $j = 1, 2, \ldots, \theta$ and $i = 1, 2, \ldots, n$, first initialize $A_i(j)$ with 0. By permuting node-indices and permuting packet-indices, for any Flower code, one can always find a Flower code in which the packet P_1 is dropped on the node U_1, where both the Flower codes have same properties. So, the binary sequences are initialized with $\mathbf{x} = 1$ and $\mathbf{y} = 1$ which ensures the packet P_1 is dropped on the node U_1. In each while loop, the sequences $\mathbf{x}$ and $\mathbf{y}$ which satisfy the Inequality (5.1) are concatenated with $0^{a-1}1$ and $0^{b-1}1$. For the loop, the positive integers a and b are selected such that those concatenated sequences $\mathbf{x}0^{a-1}1$ and $\mathbf{y}0^{b-1}1$ satisfy the Inequality (5.1). Note that $w_{\mathbf{x}0^{a-1}1} = w_{\mathbf{x}} + 1$ and $w_{\mathbf{y}0^{b-1}1} = w_{\mathbf{y}} + 1$ and hence, by induction with initial condition, $x_1 = y_1 = 1$, $w_{\mathbf{x}0^{a-1}1} = w_{\mathbf{y}0^{b-1}1}$. In the Algorithm 1, the positive integers a and b can be a distinct for different while loop as well as within a loop. One can introduce randomness by selecting a and b as random integers. The algorithm will terminate, if $w_{\mathbf{x}} = w_{\mathbf{y}} > z$. The condition on while loop ensures the weight of each binary sequence is z.

An Example of the construction of sequences is illustrated in Table 5.1 for $n = 3$ and $\theta = 3$. From the algorithm, $\mathbf{x} = 11001100101$ and $\mathbf{y} = 100101010011$. The Flower code is $U_1 = \{P_1, P_2\}$, $U_2 = \{P_1, P_3\}$, $U_3 = \{P_2, P_3\}$. Note that the Flower code is universally good.

Table 5.1: For $n = \theta = 4$, construction of sequences $\mathbf{x}$ and $\mathbf{y}$ using the Algorithm 1.

Sr. No.	a	$\mathbf{x}$	b	$\mathbf{y}$	$A_i(j)$
1.	-	1	-	1	$A_1(1) = 1$
2.	1	10^01	3	10^21	$A_2(1) = 1$
3.	3	10^010^21	2	10^210^11	$A_2(3) = 1$
4.	1	$10^010^210^01$	2	$10^210^110^11$	$A_3(2) = 1$
5.	3	$10^010^210^010^21$	3	$10^210^110^110^21$	$A_1(2) = 1$
6.	2	$10^010^210^010^210^11$	1	$10^210^110^110^210^01$	$A_3(3) = 1$

Algorithm 1 Construction of sequences $\mathbf{x}$ and $\mathbf{y}$ for a universally good Flower code.

Require: $n, \theta \geq 1$ and an integer $z \geq \max\{n, \theta\}$.
Ensure: The finite binary sequences $\mathbf{x}$ and $\mathbf{y}$.
  Initialize: $A_i(j) = 0$ for each $j = 1, 2, \ldots, \theta$ and $i = 1, 2, \ldots, n$.
  Initialize: $\mathbf{x} = x_1 = 1$, $\mathbf{y} = y_1 = 1$ and update $A_1(1) = 1$.
  while $w_{\mathbf{x}}(m) < z$ for $m > 1$ **do**
    For positive integers a $(\leq n)$ and b $(\leq \theta)$
    if $A_p(j) + \sum_{s=1, s\neq j}^{\theta} A_i(s)A_p(s) \leq 1$ for $p = 1, 2, \ldots, n$ and $i \neq p$, where $i - (m + a) \equiv 0 \ (mod\ n)$ and $j - (r + b) \equiv 0 \ (mod\ \theta)$ **then**
      Update $\mathbf{x}$ by $\mathbf{x}0^{a-1}1$ and $\mathbf{y}$ by $\mathbf{y}0^{b-1}1$
      Update $A_i(j)$ by $A_i(j) + 1$
      Update m by $m + a$, and r by $r + b$
    end if
  end while
  Update ℓ by m, and t by r

For the binary sequences generated by the Algorithm 1, the following Lemma ensures the universally good condition of the Flower code.

Lemma 5.5. *For positive integers n and θ, if the binary sequences $\mathbf{x}$ and $\mathbf{y}$ are generated by the Algorithm 1 then the Flower code is universally good.*

Proof. For positive integers a and b in a loop of the Algorithm, $x_{m+a} = y_{r+b} = 1$ and $w_x(m + a) = w_y(r + b)$. Therefore, the packet P_j will be placed on node U_i, where $i - (m + a) \equiv 0 \ (mod\ n)$, $j - (r + b) \equiv 0 \ (mod\ \theta)$, $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, \theta\}$. For each integer $p = 1, 2, \ldots, n$ such that $i \neq p$, $A_p(j) + \sum_{s=1, s\neq j}^{\theta} A_i(s)A_p(s) = \sum_{s=1}^{\theta} A_i(s)A_p(s) \leq 1$, where $A_i(j) = 1$ for $j \in \{1, 2, \ldots, \theta\}$ such that $j - (r + b) \equiv 0 \ (mod\ \theta)$. From the Lemma 5.3, the Lemma holds. □

The Flower code which corresponds to the dual GFR code is called a dual Flower code. The following theorem gives the connection between a Flower code

and its dual.

Theorem 5.2. *Consider a Flower code with n nodes, θ packets, and two binary sequences* $\mathbf{y}$ *and* $\mathbf{x}$ *(the packet selection sequence and the packet dropping sequence respectively). A Flower code with n^* nodes, θ^* packets and two binary sequences* $\mathbf{y}^*$ *and* $\mathbf{x}^*$ *(the packet selection sequence and the packet dropping sequence respectively) is the dual code if and only if $n^* = \theta$, $\theta^* = n$, $\mathbf{x}^* = \mathbf{y}$ and $\mathbf{y}^* = \mathbf{x}$.*

Proof. The proof follows the Definition 2.12 and the Fact 5.1. □

The following theorem gives the connection for universally good condition on Flower code and its dual code.

Theorem 5.3. *The dual of a universally good Flower code is again universally good.*

Proof. Indeed, if a Flower code is universally good then, in the dual, two packets P_i^* $(1 \leq i \leq \theta^*)$ and P_j^* $(1 \leq j \leq \theta^*)$ cannot stored in two distinct nodes U_r^* $(1 \leq r \leq n^*)$ and U_s^* $(1 \leq s \leq n^*)$, as then, packets P_r and P_s are stored in both nodes U_i and U_j which contradicts $|U_i \cap U_j| \leq 1$. □

From the Theorem 5.1 and 5.2, one can easily observe the following proposition.

Proposition 5.5. *For a Flower code with n nodes, θ packets and characteristic binary sequences* $\mathbf{y}$ *and* $\mathbf{x}$*, the parameters of dual Flower code are $\alpha_j^* = \sum_{p=0}^{\left\lfloor \frac{t-j}{\theta} \right\rfloor} y_{p\theta+j}$ $(j = 1, 2, \ldots, \theta)$ and $\rho_i^* = \sum_{s=0}^{\left\lfloor \frac{\ell-i}{n} \right\rfloor} x_{sn+i}$ $(i = 1, 2, \ldots, n)$.*

Now, we explore some specific cases of Flower code such as Flower code with packet selection sequence $\mathbf{y} = 1^t$ and Flower code with packet dropping sequence $\mathbf{x} = 1^\ell$ in the following two sections.

5.1.2 Flower codes with all one packet selection sequences

In the subsection, we discuss the properties (such as parameters, universally good condition and dual property) of Flower codes with all one packet selection sequence. The Flower code with uniform replication factor can also be constructed by taking packet selection sequence $\mathbf{y} = 1^{\rho\theta}$. For n nodes, θ packets, and binary sequences $\mathbf{x}$ and $\mathbf{y} = 1^t$, from the Lemma 5.1, the Flower code exists if $t = w_{\mathbf{x}}$.

From the Fact 5.1, in the Flower code, packet P_j is stored in node U_i if and only if $x_m = 1$, $w_{\mathbf{x}}(m) - j \equiv 0 \ (mod\ \theta)$ and $m - i \equiv 0 \ (mod\ n)$. The parameters of such Flower codes are calculated in the following Theorem.

Theorem 5.4. *For a Flower code with n nodes and θ packets and a binary sequence $\mathbf{x}$ of length ℓ, the packet replication factor of the packet P_j ($j = 1, 2, \ldots, \theta$) is*

$$\rho_j = \begin{cases} \frac{w_{\mathbf{x}}}{\theta} & : \text{if } \eta = 0, \\ 1 + \left\lfloor \frac{w_{\mathbf{x}}}{\theta} \right\rfloor & : \text{if } j = 1, 2, \ldots, \eta \text{ and } \eta \neq 0, \\ \left\lfloor \frac{w_{\mathbf{x}}}{\theta} \right\rfloor & : \text{if } j = \eta + 1, \ldots, \theta \text{ and } \eta \neq 0; \end{cases}$$

where $\eta = w_{\mathbf{x}} - \theta \left\lfloor \frac{w_{\mathbf{x}}}{\theta} \right\rfloor$.

Proof. From the division algorithm, $w_{\mathbf{x}} = \theta \left\lfloor \frac{w_{\mathbf{x}}}{\theta} \right\rfloor + \eta$ for some $0 \leq \eta < \theta$. So, $\frac{w_{\mathbf{x}} - \eta}{\theta}$ is a positive integer $\left\lfloor \frac{w_{\mathbf{x}}}{\theta} \right\rfloor$. Now, from the Theorem 5.1, for $\mathbf{y} = 1^{w_{\mathbf{x}}}$,

$$\begin{aligned} \rho_j &= \sum_{p=0}^{\left\lfloor \frac{w_{\mathbf{x}} - j}{\theta} \right\rfloor} y_{p\theta + j} \\ &= \left\lfloor \frac{w_{\mathbf{x}} - j}{\theta} \right\rfloor + 1 \\ &= \left\lfloor \frac{w_{\mathbf{x}} - \eta}{\theta} + \frac{\eta - j}{\theta} \right\rfloor + 1 \\ &= \left\lfloor \frac{w_{\mathbf{x}} - \eta}{\theta} \right\rfloor + \left\lfloor \frac{\eta - j}{\theta} \right\rfloor + 1 \\ &= \left\lfloor \frac{w_{\mathbf{x}}}{\theta} \right\rfloor + \left\lfloor \frac{\eta - j}{\theta} \right\rfloor + 1 \end{aligned} \tag{5.3}$$

Now, there are two cases.

- Case 1: if $0 < j \leq \eta < \theta$ then $0 < \frac{\eta - j}{\theta} < 1$. Therefore, $\left\lfloor \frac{\eta - j}{\theta} \right\rfloor = 0$.
- Case 2: if $0 \leq \eta \leq j < \theta$ then $-1 < \frac{\eta - j}{\theta} < 0$. So, $\left\lfloor \frac{\eta - j}{\theta} \right\rfloor = -1$.

From Equation 5.3, and Case 1 and 2, the result holds. □

Using Theorem 5.4, one can observe the following remark.

Remark 5.4. *For n nodes and θ packets, consider a Flower code with binary sequences $\mathbf{x}$ (length ℓ) and $\mathbf{y} = 1^{w_{\mathbf{x}}}$. For two distinct integers m_1 ($\leq \ell$) and m_2 ($\leq \ell$), let*

$x_{m_1} = x_{m_2} = 1$. *If* $m_1 - m_2 \equiv 0 \ (mod\ n)$ *and* $w_{\mathbf{x}}(m_1) - w_{\mathbf{x}}(m_2) \equiv 0 \ (mod\ \theta)$ *then the two replicas of the packet* P_j *are dropped on node* U_i, *where* $w_{\mathbf{x}}(m_1) - j \equiv 0 \ (mod\ \theta)$ *and* $i - m_1 \equiv 0 \ (mod\ n)$.

Using the Definition of a Flower code and the Theorem 5.4, one can easily prove the following proposition.

Proposition 5.6. *For a Flower code with n nodes,* θ *packets, and binary sequences* $\mathbf{x}$ *(length* ℓ*) and* $\mathbf{y} = 1^{w_{\mathbf{x}}}$, *the replication factor* $\rho_j \in \{\rho - 1, \rho\}$ $(j = 1, 2, \ldots, \theta)$.

Lemma 5.6. *A Flower code with n nodes,* θ *packets, and binary sequences* $\mathbf{x}$ *(length* ℓ*) and* $\mathbf{y} = 1^{w_{\mathbf{x}}}$, *is universally good, if and only if* $\sum_{j=1}^{\theta} A_i(j) A_p(j) \leq 1$ *for each* $i, p = 1, 2, \ldots, n$ *and* $i \neq p$.

Proof. The proof follows the Theorem 5.1 for the packet selection sequence $\mathbf{y} = 1^{w_{\mathbf{x}}}$. □

Using the Remark 5.3, one can easily prove the following two Lemmas.

Lemma 5.7. *For given integers* n, θ *and a periodic binary sequence* $\mathbf{x}$ *with the period* $\tau < n < \theta$, *if* $\ell = \left\lceil \frac{2\theta\tau}{w_{\mathbf{x}}(\tau)} \right\rceil$ *then the Flower code with two characteristic binary sequences* $\mathbf{y} = 1^{\rho\theta}$ *and* $\mathbf{x}$ *will be universally good with* $\rho = 2$, *where* $gcd(n, \theta) = 1$ *and* $gcd(n, \tau) = 1$.

Proof. For a periodic binary sequence $\mathbf{x}$ with the period $\tau \leq n$, if $\ell = \left\lceil \frac{2\theta\tau}{w_x(\tau)} \right\rceil$ then the entry sum will be 2θ. For the particular periodic sequence, if $gcd(n, \theta) = 1$ and $gcd(n, \tau) = 1$ then any two node will not contain more than one common packet. Hence, the result holds. □

For example, $\tau = 3$, $n = 4$, $\theta = 5$, the sequence $\mathbf{x} = 110110110110110$ will give the universally good Flower code, where $\alpha_1 = \alpha_2 = 3$ and $\alpha_3 = \alpha_4 = 2$ and $\rho_j = 2$ $(j = 1, 2, 3, 4, 5)$. Note that the sequence $\mathbf{x}$ is the Fibonacci sequence with modulo 2.

Proposition 5.7. *For* n, θ, *and the a binary sequence* $\mathbf{x}$, *the Flower code is a universally good, where the sequence is generated from the Algorithm 1 with* $b = 1$.

5.1.3 Constructions of Flower codes from various sequences

In order to construct a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$, one can drop θ packets P_j ($j = 1, 2, \ldots, \theta$). This motivates us to define *cycle m* and *jump* for $m \in \{1, 2, \ldots, \rho\}$.

Definition 5.2. *(Cycle): For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$, and for every m ($1 \leq m \leq \rho$), a cycle is defined as dropping of all θ packets (with no repetition) on n nodes.*

An example of such cycle is shown in Table 5.2.

Definition 5.3. *(Jump): The number of null packets between two consecutive packets from $\{1, 2, \ldots, \theta\}$ while dropping them on n nodes $U_1, U_2, \ldots, U_n$ is known as a Jump.*

An example of such jump is shown in Table 5.2. In the table, dash between two packets (such as packets indexed by 3 and 4) represents jump for the certain cycle.

Table 5.2: An example of jumps and cycles for the 4 packets on 3 nodes.

	Cycle 1					**Cycle 2**						
Packet Index	1	2	3	–	4	–	–	1	2	–	3	4
Node Index	1	2	3	1	2	3	1	2	3	1	2	3

One can associate a binary characteristic sequence (dropping sequence), node sequence (ordering the node sequence where the packet is dropped) and node-packet incidence matrix with any GFR code. We now formally collect these definitions.

Definition 5.4. *(Dropping Sequence): a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ can be characterized by a binary characteristic sequence (weight of the sequence is $\rho\theta$) which is one whenever a packet is dropped on a node and zero whenever no packet is dropped.*

For a given $n = 4$, $\theta = 6$ and dropping sequence **d** = 11111100011100111, one can find the GFR code $\mathscr{C} : (4, 6, 3, 2)$ with $U_1 = \{P_1, P_5, P_6\}$, $U_2 = \{P_1, P_2, P_6\}$, $U_3 = \{P_2, P_3, P_4\}$ and $U_4 = \{P_3, P_4, P_5\}$. From the dropping sequence, the GFR code is generated by dropping packet P_1 on node U_1, since $d_1 = 1$ and so on.

Dropping sequence **d** with length l of a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ has the following properties.

1. The m^{th} term of the dropping sequence $\mathbf{d}$ is

$$d_m = \begin{cases} 1 & : \text{ if packet is dropped on node } U_t; \\ 0 & : \text{ if packet is not dropped on node } U_t, \end{cases}$$

where

$$t = \begin{cases} m \ (mod \ n) & : \text{ if } n \nmid m; \\ n & : \text{ if } n \mid m. \end{cases}$$

2. WLOG one can set $d(l) = 1$ for dropping sequence $\mathbf{d}$ with length $l \in \mathbb{N}$ $s.t.$ weight $w_{\mathbf{d}} = \rho\theta$ and $d_m \in \{0, 1\}$.

3. It is clear that $l \in \{\rho\theta, \rho\theta + 1, ..., \rho\theta + (n-1)^{\rho\theta-1}, ...\}$ but one can reduce the value of l such that $\rho\theta \leq l \leq \rho\theta + (n-1)^{\rho\theta-1}$ by puncturing n consecutive 0's in $\mathbf{d}$. It can be observed easily that if $l > \rho\theta + (n-1)^{\rho\theta-1}$ then $\exists\, q(\geq n)$ consecutive 0's in $\mathbf{d}$.

4. If $d_r = 1$ $(1 \leq r \leq l)$ then the packet P_λ is distributed on the node U_t, where $\eta(r) = w_{\mathbf{d}}(r) \ (mod \ \theta)$ and

$$\lambda = \begin{cases} \eta(r) & : \text{ if } \eta(r) \neq 0; \\ \theta & : \text{ if } \eta(r) = 0. \end{cases}$$

Definition 5.5. *(Node Sequence): A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ can be characterised by a finite sequence $\mathbf{s}$ of length $\rho\theta$ defined from $\{1, 2, \ldots, \rho\theta\}$ $(\subset \mathbb{N})$ to $\{1, 2, ..., n\}$ $(\subset \mathbb{N})$ such as packet P_t $(t \in \{1, 2, \ldots \theta\})$ is dropped on node U_{s_i} $(s_i \in \{1, 2, \ldots n\})$, where*

$$t = \begin{cases} i \ (mod \ \theta) & : \ if \ (i \ (mod \ \theta)) \neq 0; \\ \theta & : \ if \ (i \ (mod \ \theta)) = 0. \end{cases} \tag{5.4}$$

For given n nodes and θ packets, a possible node sequence is $(1\,2\,3\,4\,1\,2\,2\,3\,4\,3\,4\,1)$ with alphabet $\{1, 2, 3, 4\}$ for the GFR code $\mathscr{C} : (4, 6, 3, 2)$ with $U_1 = \{P_1, P_5, P_6\}$, $U_2 = \{P_1, P_2, P_6\}$, $U_3 = \{P_2, P_3, P_4\}$ and $U_4 = \{P_3, P_4, P_5\}$.

Remark 5.5. *For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$, there may exist more then one dropping sequences $\mathbf{d}$ with length l as well as node sequences $\mathbf{s}$ with length $\rho\theta$.*

Next section describes the construction of the flower codes. A generalized ring construction of GFR codes for $\rho = 2$ was described in [38] which gives homogeneous (heterogeneous) GFR codes for θ a multiple of n (or θ not a multiple of n). In this section, we generalize $\rho = 2$ construction to $\rho > 2$. To construct the GFR code of replication factor ρ, we first place n nodes on a circle. Now we can place θ packets on each of them one by one until a cycle is complete. One has to place the packets until we complete all ρ cycles. It rises to a GFR code since all θ packets are replicated ρ times in the system. Now we can vary the packet dropping mechanism by introducing jumps within a cycle or after every cycle. To do so first, we define a different kind of jumps. This process yields several classes of interesting GFR codes.

Definition 5.6. *(Internal & External Jumps): An internal jump is a jump applied within a cycle $m, 1 \leq m \leq \rho$. Similarly a jump is called external jump if it is applied between two consecutive cycles m & $m+1, 1 \leq m \leq \rho$. In particular, internal (external) jump function is denoted by $f_{in}(f_{ex})$ with domain $\{1, 2, ..., \rho\theta\} \setminus \{\theta, 2\theta, \ldots, \rho\theta\}$ ($\{1, 2, ..., \rho\}$). Both jump functions have common co-domain $\mathbb{N} \cup \{0\}$.*

For given nodes $n = 3$, packets $\theta = 4$, internal jump function $f_{in} = 1$ and external jump function f_{ex}, one can construct node sequence $(1\ 3\ 2\ 1\ 2\ 1\ 3\ 2)$ for some GFR code $\mathscr{C} : (3, 4, 3, 2)$.

Remark 5.6. *A special kind of jump, (for example, see Definition 5.7), can be described by a characteristic function $\xi(i) = 1$ (drop) or 0 (do not drop) which tells us when to drop a packet at a position $i, 1 \leq i \leq n$.*

Definition 5.7. *(Subset Type Jumps): Let $\{1, 2, \ldots, n\}$ be an index set of n nodes and let $A \subseteq \{1, 2, \ldots n\}$. A jump is called subset type jump if it's characteristic function $\xi(i)$ is given by*

$$\xi(i) = \begin{cases} 1 & : \text{if } i \in A; \\ 0 & : \text{if } i \notin A. \end{cases}$$

For example, an Flower code $\mathscr{F}_{\mathscr{C}} : (8, 7, (3\ 4\ 2\ 2\ 3\ 3\ 3\ 1), 3)$ is constructed using subset type jumps $A_1 = \{1, 2, 4\}$, $A_2 = \{5, 6, 7, 8\}$ and $A_3 = \{2, 3, 5, 6, 7\}$ in Table 5.3. Now we are ready to define a Flower code with single ring having a subset type jump within it's ρ cycles.

Definition 5.8. *(Flower Code with Single Ring): A Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ with single ring and having a subset type jumps can be defined by first placing n nodes along a single ring and then dropping packets as per a subset jump $A_m (1 \leq m \leq \rho)$ (see Definition 5.7) within every cycle $m (1 \leq m \leq \rho)$ till we drop all $\rho\theta$ packets on the ring.*

Table 5.3: An example of a Flower code constructed by subset type jumps.

Node	**Packet distribution**							**Node Capacity**
	For A_1			**For** A_2		**For** A_3		α_i
U_1	P_1	P_4	P_7	-	-	-	-	3
U_2	P_2	P_5	-	-	-	P_1	P_6	3
U_3	-	-	-	-	-	P_2	P_7	2
U_4	P_3	P_6	-	-	-	-	-	2
U_5	-	-	-	P_1	P_5	P_3	-	3
U_6	-	-	-	P_2	P_6	P_4	-	3
U_7	-	-	-	P_3	P_7	P_5	-	3
U_8	-	-	-	P_4	-	-	-	1

Lemma 5.8. *Consider a Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ with single ring and having subset type jumps on node subsets $A_1, A_2, \ldots A_\rho$. If total number of packets distributed on node $U_{i_t} (i_t \in A_m; 1 \leq m \leq \rho$ and $t = 1, 2, \ldots, |A_m|)$ for subset type jump with single ring on node subset A_m is $P(U_{i_t}, A_m)$ then*

$$P(U_{i_t}, A_m) = \begin{cases} \left\lceil \frac{\theta}{|A_m|} \right\rceil & : \text{ if } t \leq \theta \ (mod\ |A_m|); \\ \\ \left\lfloor \frac{\theta}{|A_m|} \right\rfloor & : \text{ otherwise.} \end{cases}$$

Proof. Suppose $A_1, A_2, \ldots A_\rho$ are the subsets of $\{1, 2, \ldots, n\}$. For the given n nodes, θ packets and A_m $(m = 1, 2, \ldots \rho)$ one can construct flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ with single ring and subset type jump on A_m, for each m. Since subset type jump on particular subset A_m is assigning θ distinct packets on nodes $U_{i_t} \in A_m$ $(1 \leq t \leq |A_m|)$ in some specific order. If $|A_m|$ divides θ then number of packets

dropped on a particular node U_{i_t} is $\frac{\theta}{|Am|}$. If $|A_m|$ does not divide θ then after dropping $\left\lfloor \frac{\theta}{|Am|} \right\rfloor$ packets on each node of A_m, there will remain $\theta \ (mod \ |Am|)$ packets to assign nodes. Hence, there are $\theta \ (mod \ |Am|)$ number of nodes U_{i_t} ($i_t \in A_m$) with $\left\lfloor \frac{\theta}{|Am|} \right\rfloor + 1$ packets each. Remaining nodes in the set A_m have $\left\lfloor \frac{\theta}{|Am|} \right\rfloor$ packets each. Hence, the lemma is proved. □

Lemma 5.9. *For a Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ with single ring and having subset type jumps on node subsets $A_1, A_2, \ldots A_\rho$, the total number of packets stored on a particular node U_i is*

$$\alpha_i = \sum_{m=1}^{\rho} P(U_i, A_m),$$

where, $P(U_i, A_m) = 0$ for $U_i \notin A_m$.

Proof. For a Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ with single ring and having subset type jumps, all packets stored on a particular node are equal to the sum of total number of packets dropped on the node for each subset type jump of A_m ($1 \leq m \leq \rho$). Hence, the lemma is proved. □

Remark 5.7. *For a Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ with a subset type jump $A_m (1 \leq m \leq \rho)$ has the following properties*

- *If $|A_{\max}| = \max\{|A_1|, |A_2|, ..., |A_\rho|\}$ then*

$$\left\lfloor \frac{\theta}{|A_{\max}|} \right\rfloor \leq \alpha_i \leq \alpha \leq \sum_{m=1}^{\rho} \left\lceil \frac{\theta}{|A_m|} \right\rceil.$$

- *If $\exists \ U_i$ s.t. $U_i \in \bigcap_{m=1}^{\rho} A_m$ then the node U_i has the maximum number of distributed distinct packets i.e., $|U_i| = \alpha$ but the converse is not true.*

- *For $1 \leq i, j, p \leq n$, if $\exists \ U_p \ (p \in A_m)$ s.t.*

$$U_p \notin \bigcup_{\substack{i=1 \\ i \neq m}}^{\rho} A_i, \text{ then } \alpha_{\max} \geq \left\lceil \frac{\theta}{|A_m|} \right\rceil.$$

To construct GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$, one can concatenate ρ distinct cycles with some internal and external jumps, where each cycle defined on n nodes.

Definition 5.9. *(Flower Code with Multiple Rings): A system with ρ cycles of θ packets in which packets are distributed among n nodes arranged on a circle with internal jump function $f_{in} : \{1, 2, ..., \rho\theta\} \setminus \{\theta, 2\theta, \ldots, \rho\theta\} \to \mathbb{N} \cup \{0\}$ and external jump function $f_{ex} : \{1, 2, ..., \rho\} \to \mathbb{N} \cup \{0\}$ is called Flower code $\mathscr{F}_{\mathscr{C}}$ with parameters $n, \theta, \vec{\alpha}$ and ρ, where $\vec{\alpha} = (\alpha_1\ \alpha_2 \ldots \alpha_n)$.*

Note that the Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ has internal and external jump functions (f_{in} and f_{ex} respectively) and each Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ is a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ so terms of node sequence **s** (length $\rho\theta$) can be represented in terms of f_{in} and f_{ex} as described in Theorem 5.5 .

Theorem 5.5. *Consider a Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ having an internal jump function $f_{in} : \{1, 2, ..., \rho\theta\} \setminus \{\theta, 2\theta, \ldots, \rho\theta\} \to \mathbb{N} \cup \{0\}$ and an external jump function $f_{ex} : \{1, 2, ..., \rho\} \to \mathbb{N} \cup \{0\}$. If node sequence of the Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ is **s** (length $\rho\theta$) then*

$$s_m = \begin{cases} 1 & : \text{if } m = 1; \\ \vartheta(m) & : \text{if } \vartheta(m) \neq 0, 1 < m \leq \rho\theta; \\ n & : \text{if } \vartheta(m) = 0, 1 < m \leq \rho\theta; \\ 0 & : \text{if } m > \rho\theta, \end{cases} \tag{5.5}$$

where

$$\vartheta(m) = \left\lfloor m + \sum_{\substack{i=0 \\ \theta \nmid i}}^{m-1} f_{in}(i) + \sum_{\substack{i=0 \\ \theta \mid i}}^{m-1} f_{ex}\left(\frac{i}{\theta}\right) \right\rfloor (mod\ n).$$

Proof. Suppose node sequence of the Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ is **s** of length $\rho\theta$ with an internal jump function $f_{in}(m) : \{1, 2, ..., \rho\theta\} \setminus \{\theta, 2\theta, \ldots, \rho\theta\} \to \mathbb{N} \cup \{0\}$ and an external jump function $f_{ex}(m) : \{1, 2, ..., \rho\} \to \mathbb{N} \cup \{0\}$. By the Definition 5.4 of Node Sequence **s** of Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$, we have U_{s_m} ($s_m \in \{1, 2, \ldots n\}$) is the node on which packet is dropped and WLOG for $m = 1$ one can take $s_1 = 1$. If p is the index of term s_p in the sequence **s** then for $p \neq 1$ and $p \leq \rho\theta$, following cases are raised.

1. If $\theta \mid p$ then θ^{th} packet is placed on s_p for a cycle and first packet for the next cycle is placed on s_{p+1}. Hence there is external jump between packets dropped on node s_p and node s_{p+1}. Clearly $\frac{p}{\theta}$ gives the index number of jump completed at node s_p.

2. If $\theta \nmid p$ then both packets dropped on nodes indexed s_p and s_{p+1}, are from same cycle so there is internal jump between the both nodes.

Note that both cases can not fall on same node s_m in node sequence **s**. Following case 1 for $p = m - 1$,

$$s_m = s_{m-1} + f_{ex}\left(\frac{m-1}{\theta}\right) + 1, \tag{5.6}$$

where $\theta \mid (m-1)$ and $m \in \{1, 2, \ldots \rho\theta\}$. Again for case 2,

$$s_m = s_{m-1} + f_{in}\,(m-1) + 1, \tag{5.7}$$

where $\theta \nmid (m-1)$ and $m \in \{1, 2, \ldots \rho\theta\}$.

Hence one can have the following recursive equation by the above equations (5.6) and (5.7) with boundary conditions $s_1 = 1$ and $s_i = 0\ \forall i(\in \mathbb{N}) > \rho\theta$ on node sequence **s** of length $\rho\theta$.

$$s_m = \begin{cases} 1 & : \text{ if } m = 1; \\ s_t + f_{ex}(t_{ex}) + 1 & : \text{ if } \theta \mid t, 1 < m \leq \rho\theta; \\ s_t + f_{in}(t) + 1 & : \text{ if } \theta \nmid t, 1 < m \leq \rho\theta; \\ 0 & : \text{ if } m > \rho\theta, \end{cases} \tag{5.8}$$

where $t_{ex} = \frac{m-1}{\theta}$ and $t = (m-1)$. Solving the recursive equation (5.8), one can get relation (5.5). □

Since each GFR code can be represented by node sequence, dropping sequence and incidence matrix so each Flower code can also be represented by node sequence, dropping sequence, and incidence matrix. The following lemmas establish the relationships among those collectively. The lemmas are as follows.

Lemma 5.10. *Consider a dropping sequence* $\boldsymbol{d}$ *of length* l *for a Flower code* $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$. *If* $d_t = 1$ *for any* t $(1 \leq t \leq l)$ *then NPDI Matrix* $B_{n\times\theta} = [b_{i,j}]_{n\times\theta}$ *is given by* $b_{i,j} = 1$, *where*

$$j = \begin{cases} w_{\boldsymbol{d}}(t) \ (mod\ \theta) & : \ if\ \theta \nmid w_{\boldsymbol{d}}(t); \\ \theta & : \ if\ \theta \mid w_{\boldsymbol{d}}(t). \end{cases} \tag{5.9}$$

and

$$i = \begin{cases} t \ (mod\ n) & : \ if\ n \nmid t; \\ n & : \ if\ n \mid t; \end{cases} \tag{5.10}$$

Proof. Let $\mathbf{d}$ be a dropping sequence of length l for a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ with n nodes and θ packets. For a given $m \in \{1, 2, \ldots, l\}$, if $d_m = 1$ then the index of the packet associated with the d_m is mapped to weight $w_{\mathbf{d}}(m)$. Hence, it proves the lemma. □

Lemma 5.11. *Consider a Flower code* $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ *with node sequence* $\boldsymbol{s}$ *of length* $\rho\theta$. *Its dropping sequence is given by* $\langle d(m)\rangle_{m=1}^{l}$ *s.t.* $\forall\ i, 1 \leq i \leq \rho\theta$

$$d(m) = \begin{cases} 1 & : \ if\ m = \sum_{j=1}^{i} \left(s_j - s_{j-1}\right) (mod\ n); \\ 0 & : \ if\ m \neq \sum_{j=1}^{i} \left(s_j - s_{j-1}\right) (mod\ n), \end{cases}$$

where $s_0 = 0$.

Proof. For a Flower code $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$, consider a dropping sequence $\mathbf{d}$ of length l. By the definition node sequence, two packets with consecutive indexes are associated with s_i and s_{i+1}, for some $i \in \{1, 2, \ldots, n\}$. Using the definition of dropping sequence, one can find that $[s_{i+1} - s_i]\ (mod\ n)$ number of zeros exist between two consecutive $1's$. In particular, the $1's$ are associative with the two packets. It proves the lemma. □

Lemma 5.12. *Consider a Flower code* $\mathscr{F}_{\mathscr{C}} : (n, \theta, \vec{\alpha}, \rho)$ *with dropping sequence* $\boldsymbol{d}$ *with*

length l. Its node sequence is given by $\mathbf{s}$ *of length* $\rho\theta$ *s.t. for each t* $(1 \leq t \leq l)$ *with* $d_t = 1$,

$$i = w_{\boldsymbol{d}}(t)$$

and

$$s_i = \begin{cases} t \ (mod\ n) & : \text{ if } n \nmid t; \\ n & : \text{ if } n \mid t. \end{cases}$$

Proof. Using the defination of node sequence and dropping sequence, one can easily prove the lemma by observing that the weight $w_{\mathbf{d}}(t)$ is associated with the packet for $d_t = 1$. □

5.1.4 Flower codes with all one packet dropping sequences

In the subsection, we discuss the properties (such as parameters, universally good condition, and dual property) of Flower codes with all one packet dropping sequence. The Flower code with uniform node storage capacity can also be constructed by taking packet dropping sequence $\mathbf{x} = 1^{\rho\theta}$. For n nodes, θ packets, and binary sequences $\mathbf{x} = 1^{\ell}$ and $\mathbf{y}$, from the Lemma 5.1, the Flower code exists if $\ell = w_{\mathbf{y}}$. From the Fact 5.1, in the Flower code, packet P_j is stored in node U_i if and only if $y_r = 1$, $r - j \equiv 0 \ (mod\ \theta)$ and $w_{\mathbf{y}}(r) - i \equiv 0 \ (mod\ n)$. The parameters of such Flower codes are calculated in the following Theorem.

Theorem 5.6. *For a Flower code with n nodes,* θ *packets and a binary sequence* $\mathbf{y}$ *of length t the node storage capacity of the node* U_i $(i = 1, 2, \ldots, n)$ *is*

$$\alpha_i = \begin{cases} \frac{w_{\mathbf{y}}}{n} & : \text{if } \eta = 0, \\ 1 + \left\lfloor \frac{w_{\mathbf{y}}}{n} \right\rfloor & : \text{if } i = 1, 2, \ldots, \eta \text{ and } \eta \neq 0, \\ \left\lfloor \frac{w_{\mathbf{y}}}{n} \right\rfloor & : \text{if } i = \eta + 1, \ldots, \theta \text{ and } \eta \neq 0; \end{cases}$$

where $\eta = w_{\mathbf{y}} - \theta \left\lfloor \frac{s}{n} \right\rfloor$.

Using the Theorem 5.6, one can observe the following remark.

Remark 5.8. *For n nodes and* θ *packets, consider a Flower code with a binary sequence* $\mathbf{x} = 1^{w_{\mathbf{y}}}$ *and* $\mathbf{y}$ *(length t). For two distinct integers* $r_1 \ (\leq t)$ *and* $r_2 \ (\leq t)$, *let* $y_{r_1} = y_{r_2} = 1$. *If*

$r_1 - r_2 \equiv 0 \ (mod\ \theta)$ and $w_{\mathbf{y}}(r_1) - w_{\mathbf{y}}(r_2) \equiv 0 \ (mod\ n)$ then the two replicas of the packet P_j are dropped on a node U_i, where $r_1 - j \equiv 0 \ (mod\ \theta)$ and $w_{\mathbf{y}}(r_1) - i \equiv 0 \ (mod\ n)$.

Using the Definition of a Flower code and the Theorem 5.6, one can easily prove the following proposition.

Proposition 5.8. *For a Flower code with n nodes, θ packets, and binary sequences $\mathbf{x} = 1^{w_{\mathbf{y}}}$ and $\mathbf{y}$ (length t), the node storage capacity $\alpha_i \in \{\alpha - 1, \alpha\}$ ($i = 1, 2, \ldots, n$).*

Lemma 5.13. *A Flower code with n nodes, θ packets, and binary sequences $\mathbf{x} = 1^{w_{\mathbf{y}}}$ and $\mathbf{y}$ (length t), is universally good, if and only if $\sum_{j=1}^{\theta} A_i(j)A_p(j) \leq 1$ for each $i, p = 1, 2, \ldots, n$ and $i \neq p$.*

Proof. The proof follows the Lemma 5.3 for the packet dropping sequence $\mathbf{x} = 1^{w_y}$. □

Proposition 5.9. *For n, θ, and the a binary sequence $\mathbf{y}$, the Flower code is universally good, when the sequence is generated from the Algorithm 1 with $a = 1$.*

CHAPTER 6

Generalized Fractional Repetition Codes and Hypergraphs

Every dance is a kind of fever chart, a graph of the heart. *-Martha Graham* [1]

In this chapter, motivated by [98], a bijection between a hypergraph and a GFR code has been established and using the bijection, bounds on the parameters of GFR codes are studied for various GFR codes. Considering the same bijection for the case of linear hypergraph, the construction of universally good adaptive GFR codes are discussed.

6.1 Hypergraphs and Generalized Fractional Repetition Codes

In Chapter 2, hypergraphs are introduced. Various codes on hypergraphs are also studied in [13, 19]. This motives us to study the properties of GFR codes in terms of parameters of hypergraphs. As introduced in Chapter 2, a hypergraph is the collection of some subsets of the set of hypervertices. On the same line, a GFR code is the collection of some subsets of the set of packets. Therefore, it is easy to observe that, in a GFR code, the distribution of nodes and packets are similar to the incidence of hypervertices and hyperedges in a hypergraph. It is natural that a GFR code can be represented by a hypergraph $(V, \mathcal{E})$ with n hypervertices and θ

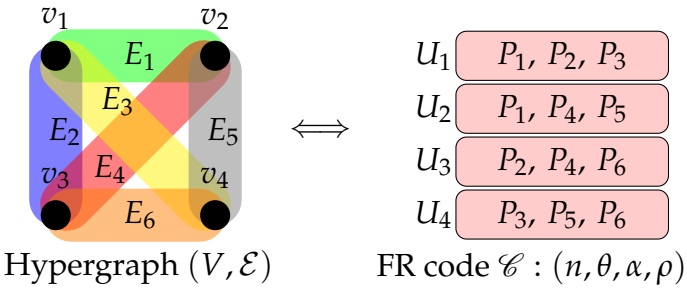

Figure 6.1: An example of the bijection between a Hypergraph and a GFR code.

hyperedges. The following fact is straightforward.

Fact 6.1. *A GFR code with n nodes, θ packets, node storage capacity α_i $(i = 1, 2, \ldots, n)$ and packet replication factor ρ_j $(j = 1, 2, \ldots, \theta)$ is equivalent to a hypergraph $(V, \mathcal{E})$ with $|V| = n$ and $|\mathcal{E}| = \theta$ such that $|\mathcal{E}(v_i)| = \alpha_i$ for $v_i \in V$, and $|E_j| = \rho_j$ for $E_j \in \mathcal{E}$, and hence, there exists a bijection $\varphi : \mathcal{E} \to \{P_j : j = 1, 2, \ldots, \theta\}$ such that $U_i = \varphi(\mathcal{E}(v_i)) = \{P_j = \varphi(E) : v_i \in E, E \in \mathcal{E}\}$, where $i = 1, 2, \ldots, n$.*

To illustrate the relationship between a hypergraph and a GFR code, let us consider an example.

Example 6.1. *As shown in Figure 6.1, consider a hypergraph $(V, \mathcal{E})$ with $V = \{v_i : i = 1, 2, 3, 4\}$ and $\mathcal{E} = \{E_j : j = 1, 2, 3, 4, 5, 6\}$, where $E_1 = \{v_1, v_2\}$, $E_2 = \{v_1, v_3\}$, $E_3 = \{v_1, v_4\}$, $E_4 = \{v_2, v_3\}$, $E_5 = \{v_2, v_4\}$ and $E_6 = \{v_3, v_4\}$. For the hypergraph, $\mathcal{E}(v_1) = \{E_1, E_2, E_3\}$, $\mathcal{E}(v_2) = \{E_1, E_4, E_5\}$, $\mathcal{E}(v_3) = \{E_2, E_4, E_6\}$ and $\mathcal{E}(v_4) = \{E_3, E_5, E_6\}$. As shown in Figure 6.1, consider a GFR code $\mathscr{C} : (n = 4, \theta = 6, \alpha = 3, \rho = 2)$ with 4 nodes and 6 packets such that $U_1 = \{P_1, P_2, P_3\}$, $U_2 = \{P_1, P_4, P_5\}$, $U_3 = \{P_2, P_4, P_6\}$ and $U_4 = \{P_3, P_5, P_6\}$, where $\vec{\alpha} = (3\ 3\ 3\ 3)$ and $\vec{\rho} = (2\ 2\ 2\ 2\ 2\ 2)$. Note that there exist the bijection $\varphi : \mathcal{E} \to \{P_j : j = 1, 2, \ldots, 6\}$ such that $\varphi(E_j) = P_j$ for each $j = 1, 2, \ldots, |\mathcal{E}|$. Hence, $\varphi(\mathcal{E}(v_i)) = \{P_j = \varphi(E_j) : E_j \in \mathcal{E}, v_i \in E_j\}$.*

The relation between the parameters of the hypergraph and the parameters of the GFR code are as follows.

- The hypervertices and hyperedges of the hypergraph are mapped to nodes and packets of the GFR code respectively.
- The number of nodes and the number of hypervertices are same *i.e.*, $n = |V|$.

- The number of distinct packets and the number of hyperedges are same *i.e* $\theta = |\mathcal{E}|$.
- The packet distribution in the node $\phi(\mathcal{E}(v)) = \{\phi(E) : E \in \mathcal{E}(v)\}$.
- The node storage capacity of any node is the same as the hypervertex degree of the respective hypervertex *i.e.*, $\alpha_i = |\mathcal{E}(v_i)|$, for $i = 1, 2, \ldots, n$.
- The replication factor of any packet is equal to the cardinality of the respective hyperedge *i.e.*, $\rho_j = |E_j|$, for $j = 1, 2, \ldots, \theta$.
- The maximum node storage capacity $\alpha = \max\{|\mathcal{E}(v)| : v \in V\}$ and the maximum replication factor $\rho = \max\{|E| : E \in \mathcal{E}\}$ for a GFR code.
- A reconstruction set of the corresponding GFR code is associated with a hypervertex cover of a sub-hypergraph $(V', \mathcal{E}')$ induced by $V' = \cup_{E \in \mathcal{E}(v)} E$, and $|\mathcal{E}'| \geq M(k)$.
- Consider a hypervertex cover $\mathscr{V}$ of a sub-hypergraph $(V', \mathcal{E}')$ of the hypergraph $(V, \mathcal{E})$, where $V' = \cup_{E \in \mathcal{E}(v)} E$ and $\mathcal{E}' = \mathcal{E}(v)$.
- In the corresponding GFR code $\mathscr{C}$, for a node failure U_i, the surviving set is associated with the set $\mathscr{V} \backslash \{v_i\}$, where $i = 1, 2, \ldots, |V|$.
- In the GFR code, the repair bandwidth for any node is equal to the hypervertex degree of respective hypervertex and the tolerance factor is equal to the corresponding hypergraph size.
- For the given positive integer $k < n$, let $(V', \mathcal{E}')$ be a sub-hypergraph induced by $V' \subset V$. The maximum file size $M(k)$ of the corresponding GFR code is $M(k) = \min\{|\mathcal{E}'| : V' \subset V, |V'| = k\}$.
- Let $(V', \mathcal{E}')$ be a sub-hypergraph induced by $V' \subset V$. The minimum distance of the corresponding GFR code is $d_{min} = \min\{|V \backslash V'| : V' \subset V \; s.t. \; |\mathcal{E}'| < \theta\}$.

For any GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, it is obvious that the parameters n and θ must be positive integer. Therefore, a Hypergraph $(V \neq \emptyset, \mathcal{E})$ with $|\mathcal{E}(v)| > 0$ (for each $v \in V$) and $|E| > 0$ (for each $E \in \mathcal{E}$) will corresponds to a GFR code with $|V|$ nodes

and $|\mathcal{E}|$ packets. For the rest of the chapter, hypergraph $(V \neq \emptyset, \mathcal{E})$ with $\mathcal{E}(v) \neq \emptyset$ (for each $v \in V$) and $E \neq \emptyset$ (for each $E \in \mathcal{E}$) is considered. In the following section, the correspondence between a hypergraph and GFR code is explored, and universally good GFR code is constructed. Since it is recursive construction, it justifies the adoptness of the system.

6.2 Construction of Universally Good Generalized Fractional Repetition Codes

As introduced in Chapter 2, for a GFR code, if any two node contain atmost one common packet then the GFR code is universally good, and, for any hypergraph, if any two hyperedges share atmost one hypervertex then the hypergraph is a linear hypergraph. Hence, the following fact is straightforward.

Fact 6.2. *A GFR code is universally good if and only if the GFR code is a linear hypergraph.*

From the fact 6.2, one can construct universally good GFR code by constructing a linear hypergraph. For a linear hypergraph, the parameters of a universally good GFR code is given in following Fact 6.3.

Fact 6.3. *For a non-empty hypervertex set V and a hyperedge set $\mathcal{E} \subset \{E : E \subset V\}$, a linear hypergraph $(V, \mathcal{E})$ is a universally good GFR code with $|V|$ nodes, $|\mathcal{E}|$ packets, node storage capacity $|\mathcal{E}(v_i)|$ ($v_i \in V$ for $i = 1, 2, \ldots, |V|$) and packet replication factor $|E_j|$ ($E_j \in \mathcal{E}$ for $j = 1, 2, \ldots, \mathcal{E}$).*

Recall that any sub-hypergraph (or induced sub-hypergraph) of a linear hypergraph is also a linear hypergraph. It motives us towards the recursive construction of a linear hypergraph by adding a hypervertex and/or adding a hyperedge into a linear hypergraph as per the following three facts 6.4, 6.5 and 6.6.

Fact 6.4. *Consider a linear hypergraph $(V, \mathcal{E})$. For a set $E \subset V$, the hypergraph $(V, \mathcal{E} \cup \{E\})$ is a linear hypergraph if and only if $|E \cap E'| \leq 1$ for each $E' \in \mathcal{E}$.*

Fact 6.5. *Consider a linear hypergraph $(V, \mathcal{E})$. For a hypervertex $v \notin V$ and a set $E \subset V$, the hypergraph $(V \cup \{v\}, \mathcal{E} \cup \{E \cup \{v\}\})$ is a linear hypergraph if and only if $|E \cap E'| \leq 1$ for each $E' \in \mathcal{E}$.*

Fact 6.6. *Consider a linear hypergraph $(V, \mathcal{E})$ and suppose $E' \subset E$, where $E \subset V$ and $E' \in \mathcal{E}$. For $E' \subset E$, the hypergraph $(V, (\mathcal{E} \backslash \{E'\}) \cup \{E\})$ is a linear hypergraph if and only if $|E \cap E''| \leq 1$ for each $E'' \in \mathcal{E}$, where $E'' \neq E'$.*

One can construct a linear hypergraph by adding one hyperedge in the hyperedge set (Fact 6.4), or injecting a hypervertex in vertex set and a hyperedge (Fact 6.5), or deleting a hyperedge from hyperedge set and adding a hyperedge which contains all the hypervertices of the deleted hyperedge (Fact 6.6) in a linear hypergraph. Considering iterations of Fact 6.4, Fact 6.5 and Fact 6.6 several time, one can get universally good GFR code with the parameters specified in Fact 6.3. For example, one can construct the GFR code of Figure 6.1 recursively as given in Table 6.1. In the example, a hypervertex v_4 and a hyperedge E_3 are added into the initial hypergraph by using the Fact 6.5. In the step 3 and 4, hyperedges E_5 and E_6 are added into the hypergraph.

Remark 6.1. *In Fact 6.4, a linear hypergraph is constructed by adding a hyperedge into a linear hypergraph. In the corresponding GFR codes, a new universally good GFR code with higher parameters can be constructed by adding a new packet into a universally good GFR code.*

Remark 6.2. *In Fact 6.5, a linear hypergraph is constructed by adding a hypervertex into a linear hypergraph. In the corresponding GFR codes, a new universally good GFR code with higher parameters can be constructed by adding a new node into a universally good GFR code.*

Remark 6.3. *In Fact 6.6, a linear hypergraph is constructed by replacing a hyperedge with a new hyperedge such that the new hyperedge contains all the hypervertices of the hyperedge. In the corresponding GFR codes, a new universally good GFR code with higher parameters is constructed by increasing the replication factor of a packet in a universally good GFR code.*

Table 6.1: Recursive construction of the GFR code (Figure 6.1).

Step No.	Hypergraph $(V, \mathcal{E})$	Adaptive Universally Good GFR code	Fact
1	$V = \{v_1, v_2, v_3\}$, $\mathcal{E} = \{E_1, E_2, E_4\}$, $E_1 = \{v_1, v_2\}$, $E_2 = \{v_1, v_3\}$, $E_4 = \{v_2, v_3\}$	$U_1 = \{P_1, P_2\}$, $U_2 = \{P_1, P_4\}$, $U_3 = \{P_2, P_3\}$	Initial Hypergraph
2	$V = \{v_1, v_2, v_3, v_4\}$ $\mathcal{E} = \{E_1, E_2, E_4, E_3\}$, $E_1 = \{v_1, v_2\}$, $E_2 = \{v_1, v_3\}$, $E_4 = \{v_2, v_3\}$, $E_3 = \{v_1, v_4\}$	$U_1 = \{P_1, P_2, P_3\}$, $U_2 = \{P_1, P_4\}$, $U_3 = \{P_2, P_3\}$, $U_4 = \{P_3\}$	Fact 6.5
3	$V = \{v_1, v_2, v_3, v_4\}$ $\mathcal{E} = \{E_i : i = 1, 2, 3, 4, 5\}$, $E_1 = \{v_1, v_2\}$, $E_2 = \{v_1, v_3\}$, $E_3 = \{v_1, v_4\}$, $E_4 = \{v_2, v_3\}$, $E_5 = \{v_2, v_4\}$	$U_1 = \{P_1, P_2, P_3\}$, $U_2 = \{P_1, P_4, P_5\}$, $U_3 = \{P_2, P_3\}$, $U_4 = \{P_3, P_5\}$	Fact 6.4
4	$V = \{v_1, v_2, v_3, v_4\}$ $\mathcal{E} = \{E_i : i = 1, 2, \ldots, 6\}$, $E_1 = \{v_1, v_2\}$, $E_2 = \{v_1, v_3\}$, $E_3 = \{v_1, v_4\}$, $E_4 = \{v_2, v_3\}$, $E_5 = \{v_2, v_4\}$, $E_6 = \{v_3, v_4\}$	$U_1 = \{P_1, P_2, P_3\}$, $U_2 = \{P_1, P_4, P_5\}$, $U_3 = \{P_2, P_3, P_6\}$, $U_4 = \{P_3, P_5, P_6\}$ (Universally good GFR code (Figure 6.1))	Fact 6.4

Lemma 6.1. *For each GFR code, dual of the GFR code exists.*

Proof. Any GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ is represented by a binary NPDI Matrix, where the entry sum of each row and each column are non-zero. The transpose of such binary matrix is again a binary matrix with nonzero entry sum of each row and entry sum of each column. Hence, there exist a GFR code $\mathscr{C}^* : (n^*, \theta^*, \vec{\alpha}^*, \vec{\rho}^*)$ with respect to the transposed binary matrix. Hence, the GFR code $\mathscr{C}^*$ is dual of the GFR code $\mathscr{C}$. □

For a GFR code, the dual GFR code is equivalent to the dual hypergraph of the corresponding hypergraph. For GFR codes with symmetric parameters, the file size of dual GFR code is studied in [135]. Considering the fact that the dual of a

linear hypergraph is again linear hypergraph, the following remark is obvious.

Remark 6.4. *The dual of a universally good GFR code is again universally good GFR code.*

6.3 Generalized Fractional Repetition Codes using Hypergraphs

In this Section, bounds on specific hypergraphs are mapped into the corresponding GFR code. Also for a GFR code, algebraic entropy, and existence condition are calculated.

6.3.1 Algebraic entropy of GFR code

The algebraic entropy of a hypergraph is discussed in [20]. Using a Hyper-FR mapping on a hypergraph, one can find the algebraic entropy for a GFR code in the following Theorem.

Theorem 6.1. *(Algebraic Entropy) Consider a GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ *with the node adjacency matrix* $A = [a_{i,j}]_{n\times n}$*. The algebraic entropy associated to the GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$*, is given by*

$$I = -\sum_{i=1}^{n} \lambda_i \log_2 \lambda_i,$$

where $\lambda_i \in \mathbb{R}$ *(for* $i = 1, 2, \ldots, n$*) are a discrete probability distribution. In particular,* λ_i *(*$i = 1, 2, \ldots, n$*) are the eigenvalues of matrix* $L(\mathscr{C}) = D - A$ *such that* $\sum_{i=1}^{n} \lambda_i = 1$ *and* $0 \leq \lambda_i \leq 1$ $(\forall i = 1, 2, \ldots, n)$*, where*

$$D = diag\left(\sum_{j=1}^{n} a_{1,j}, \sum_{j=1}^{n} a_{2,j}, \ldots, \sum_{j=1}^{n} a_{n,j}\right)$$

is the diagonal matrix.

Proof. $\forall\, v_i, v_j \in V$ the matrix element $a_{i,j} = |\mathcal{E}(v_i) \cap \mathcal{E}(v_j)|$ for $i \neq j$ and $a_{i,j} = 0$ otherwise. Define $L(\mathscr{H}) = D - A(\mathscr{H})$, where $D = diag(D(v_1), D(v_2), \ldots, D(v_n))$ and $D(v_i) = \sum_{j=1}^{n} a_{i,j}$. The square matrix $L(\mathscr{H})$ is positive semidefinite and symmetric

so eigenvalues of $L(\mathscr{H})$ are non negative real number say $\mu_i \geq 0$ $(i = 1, 2, \ldots, n)$. Moreover $\sum_{i=1}^{n} \mu_i = Tr(L(\mathscr{H})) = \sum_{i=1}^{n} D(v_i)$, where trace of matrix $L(\mathscr{H})$ is $Tr(L(\mathscr{H}))$. Now define $L'(\mathscr{H}) = \frac{L(\mathscr{H})}{\sum_{i=1}^{n} D(v_i)}$. The eigenvalues of matrix $L'(\mathscr{H})$ are $\lambda_i = \frac{\mu_i}{\sum_{i=1}^{n} D(v_i)}$. Note that $0 \leq \lambda_i \leq 1$ and $\sum_{i=1}^{n} \lambda_i = 1$, for each $i = 1, 2, \ldots, n$. Hence λ_i is discrete probability distribution for $i = 1, 2, \ldots, n$. Now one can define algebraic hypergraph entropy

$$I(\mathscr{H}) = -\sum_{i=1}^{n} \lambda_i \log_2 \lambda_i. \tag{6.1}$$

□

For more details on algebraic hypergraph entropy see [20, Chapter 1]. The significance of the algebraic entropy of a GFR code is not known yet. Note that bounds on parameters of various hypergraphs are well studied [17, 20]. Using the connection between hypergraph and GFR code, one can directly get the bounds on parameters of GFR code. In the following sections, the bounds on parameters for various GFR code are discussed.

6.3.2 Bounds for GFR Codes using Hypergraphs

For a given positive integers n, θ, α_i $(i = 1, 2, \ldots, n)$ and ρ_j $(j = 1, 2, \ldots, \theta)$, the necessary and sufficient condition for the existence of a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ is given in the following Theorem.

Theorem 6.2. *(Existence Condition) For two positive integers n and θ, consider positive integers ρ_j $(j = 1, 2, \ldots, \theta)$ and α_i $(i = 1, 2, \ldots, n)$ such that $\alpha = \alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n$. A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ exists with $|U_i| = \alpha_i$ and replication factor of packet P_j is ρ_j if*

1. $\sum_{j=1}^{\theta} \rho_j = \sum_{i=1}^{n} \alpha_i$ *and*
2. $\sum_{j=1}^{\theta} \min\{\rho_j, m\} \geq \sum_{i=1}^{m} \alpha_i$ *(for* $m < n, m \in \mathbb{N}$*).*

Proof. The proof follows from Fact 6.1 and Lemma 2.1. □

The alternative condition for the existence of a GFR code is given in the following Theorem.

Theorem 6.3. *(Alternate Existence Condition) For two positive integers n and θ, consider positive integers ρ_j ($j = 1, 2, \ldots, \theta$) and α_i ($i = 1, 2, \ldots, n$) such that $\rho = \rho_1 \geq \rho_2 \geq \ldots \geq \rho_n$. A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ exists such that $|U_i| = \alpha_i$ and replication factor of packet P_j is ρ_j if*

1. $\sum_{i=1}^{n} \alpha_i = \sum_{j=1}^{\theta} \rho_j$ *and*
2. $\sum_{i=1}^{n} \min\{\alpha_i, m\} \geq \sum_{j=1}^{m} \rho_j$ *(for* $m < \theta, m \in \mathbb{N}$*).*

Proof. The proof follows from the dual statement of the Theorem 6.2. □

For given n, θ, α_i ($i = 1, 2, \ldots, n$) and ρ_j ($j = 1, 2, \ldots, \theta$), one can observe that Theorems 6.2 and Theorem 6.3 hold simultaneously. Note that, for the given parameters n, θ, α_i ($i = 1, 2, \ldots, n$) and ρ_j ($j = 1, 2, \ldots, \theta$), if it does not satisfy one of the constraint in the Theorem 6.2 or Theorem 6.3, then a GFR code does not exist with these parameters.

Consider a GFR code with two packets having same replication factor such that the two packets are shared by same nodes. Let us call those packets as parallel packets. Similarly, let us call the two nodes U_i and U_j as parallel nodes if $U_i = U_j$ for $i \leq j$. For a GFR code without parallel packets and parallel nodes, the bound on node storage capacity and replication factor are given in the following two Lemmas.

Lemma 6.2. *A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ without parallel packets, satisfies $\alpha \leq 2^{n-1}$.*

Proof. The proof follows from the Lemma 2.2 and Fact 6.1. □

Lemma 6.3. *A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ without parallel nodes, satisfies $\rho \leq 2^{\theta-1}$.*

Proof. The proof follows from the dual statement (with respect to the hypergraph) of Lemma 6.2. □

Lemma 6.4. *For each $j = 1, 2, \ldots, \theta$, a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ with $\rho_j = \rho$ ($j = 1, 2, \ldots, \theta$) connected hypergraph, satisfies the following conditions.*

1. $\sum_{i=1}^{n} \alpha_i$ *is a multiple of* ρ*,*

2. $\sum_{i=1}^{n} \alpha_i \geq \frac{\rho(n-1)}{\rho-1}$,

3. $\alpha \leq \frac{1}{\rho} \sum_{i=1}^{n} \alpha_i$.

Proof. The proof follows from Lemma 2.10 and Fact 6.1. □

For a GFR code, let the set of nodes be a disjoint union of two subsets such that any node from one subset does not share packets with any node from the another subset. A bound on the GFR code is calculated in the following theorem.

Lemma 6.5. *Let $\mathscr{C} : (n, \theta = 2m, \vec{\alpha}, \vec{\rho})$ be a GFR code with message packets P_1, P_2,..., P_m, P'_1, P'_2,..., P_m such that $F_i \cap F'_j = \emptyset$ if and only if $i = j$, where $F_j = \{U_i : P_j \in U_i\}$ and $F'_j = \{U_i : P'_j \in U_i\}$. Then*

$$\sum_{j=1}^{m} \binom{|F_j| + |F'_j|}{|F_j|}^{-1} \leq 1.$$

Proof. The proof follows from Lemma 2.3 and Fact 6.1. □

6.3.3 Bounds for GFR Codes using Linear Hypergraphs

As introduced in Chapter 2, a hypergraph is a linear hypergraph if any two hyperedges share at most one hypervertex. Note that, in a hypergraph, any two hyperedges share at most one hypervertex if and only if any two hypervertices are shared by at most one hyperedge. Therefore, any two nodes in a GFR code, equivalent to a linear hypergraph, do not share more than one packet. A GFR code corresponding to a linear hypergraph will satisfy the following Lemmas.

Lemma 6.6. *A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ with $|U_i \cap U_j| \leq 1$ (for each $i \neq j$ and $i, j = 1, 2, \ldots, n$), satisfies $\sum_{j=1}^{\theta} \binom{\rho_j}{2} \leq \binom{n}{2}$.*

Proof. The proof follows from Fact 6.1 and Lemma 2.6. □

Lemma 6.7. *A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ with $|U_i \cap U_j| \leq 1$ (for each $i \neq j$ and $i, j = 1, 2, \ldots, n$), satisfies $\sum_{i=1}^{n} \binom{\alpha_i}{2} \leq \binom{\theta}{2}$.*

Proof. The proof follows from the dual statement of Lemma 6.6. □

The bound on the maximum file size stored in a GFR code is obtained in the following Theorems using the approach of hypergraph (see Theorem 4 in [137]).

Theorem 6.4. *A GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ *with* $|U_i \cap U_j| \leq 1$ *satisfies* $M(k) \geq \sum_{i=1}^{k} \alpha_i - \binom{k}{2}$, *where* $\alpha_i \leq \alpha_j$ $(1 \leq i < j \leq n)$.

Proof. The proof follows from the Lemma 2.4 and Lemma 2.5. □

Remark 6.5. *Recall that, a sub-hypergraph of a linear hypergraph is again a linear hypergraph and each linear hypergraph satisfies the Lemme 2.4. Hence, all the GFR codes which are equivalent to linear hypergraph satisfy the Theorem 6.4 for each* $k \leq n-1$. *So, each GFR code which is equivalent to a linear hypergraph is universally good.*

Remark 6.6. *A GFR code illustrated in [134], is equivalent to a linear hypergraph. The bound on maximum file size (Theorem 6.4) is also discussed in [134, Inequality 2].*

6.3.4 Bounds for GFR Codes using Simple Hypergraphs

As introduced in Chapter 2, a hypergraph $(V, \mathcal{E})$ is a simple hypergraph only if $E_i \subseteq E_j$ then $i = j$, for any $E_i, E_j \in \mathcal{E}$. In a GFR code which is equivalent to a simple hypergraph, all the nodes which share a packet, will not contain any other common packet. Any GFR code equivalent to a simple hypergraph will satisfy the following two lemmas.

Lemma 6.8. *A GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ *with* $\{U : P_i \in U\} \nsubseteq \{U : P_j \in U\}$ *for each* $i \neq j$ *and* $i, j = 1, 2, \ldots, n$, *satisfies* $\sum_{j=1}^{\theta} \binom{n}{\rho_j}^{-1} \leq 1$ *and* $\theta \leq \binom{n}{\lfloor n/2 \rfloor}$.

Proof. The Lemma follows from [17, Chapter 1, Theorem 2]. □

Lemma 6.9. *A GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ *with* $U_i \nsubseteq U_j$ *for each* $i \neq j$ *and* $i, j = 1, 2, \ldots, n$, *satisfies* $\sum_{i=1}^{n} \binom{\theta}{\alpha_j}^{-1} \leq 1$ *and* $n \leq \binom{\theta}{\lfloor \theta/2 \rfloor}$.

Proof. The proof follows from the duality (with respect to the hypergraph) on Lemma 6.8. □

Remark 6.7. *In the GFR code as considered in [10, 26, 103, 127, 129], if parallel nodes do not exist then the GFR code will satisfy the Lemma 6.8 and Lemma 6.9.*

In [26, 77], the GFR codes with symmetric parameters satisfy the bounds established in Lemma 6.6, 6.7, 6.8 and 6.9 with equality. In [133, 137, 134], the GFR codes with asymmetric parameters satisfy the bounds established in Lemma 6.6 and 6.7 with equality. It would be an interesting task to find more GFR codes with asymmetric parameters which satisfy Lemma 6.4, 6.5, 6.8 and 6.9 with equality.

6.3.5 Bounds for GFR Codes using Uniform Hypergraphs

As introduced in Chapter 2, for a positive integer r, an r-uniform hypergraph is a hypergraph $(V, \mathcal{E})$ with $|E| = r$, for each $E \in \mathcal{E}$. If a hypergraph is an r-uniform hypergraph then the hypergraph is called an r-uniform hypergraph. A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ associated with an r-uniform hypergraph $(V, \mathcal{E})$ will have the same replication factor for each packet *i.e.*, $\rho = \rho_j$ for each $j = 1, 2, \ldots, \theta$.

For a positive integer r, an r-uniform hypergraph $(V, \mathcal{E})$ is called an r-complete hypergraph denoted by $K^r_{|V|}$ if $\mathcal{E} = \{E : E \subseteq V, |E| = r\}$. The following Lemma gives a bound on the total number of packets of a GFR code.

Lemma 6.10. *For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$, the total number of distinct packets θ satisfies, $\theta \leq \binom{n}{\rho}$.*

Proof. In an r-complete hypergraph $K^r_{|V|}$ with no parallel hyperedges and isolated hypervertices, $|\mathcal{E}| = \binom{|V|}{r}$. An r-uniform hypergraph $(V, \mathcal{E})$ is a sub-hypergraph of the r-complete hypergraph $K^r_{|V|}$. Hence, $|\mathcal{E}| \leq \binom{|V|}{r}$. The number of distinct edges in the corresponding GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ equivalent to a ρ-uniform hypergraph $(V, \mathcal{E})$, θ will satisfy, $\theta \leq \binom{n}{\rho}$. □

Note that the condition given in Lemma 6.10, is derived in [28] for GFR codes with symmetric parameters. If an r-uniform hypergraph is a connected hypergraph then the hypergraph is called a connected r-uniform hypergraph.

Lemma 6.11. *A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ with $|U_i \cap U_j| \leq 1$ for each $i, j = 1, 2, \ldots, n$ satisfies $\theta \leq \frac{n(n-1)}{\rho(\rho-1)}$.*

Proof. The proof follows from Fact 6.1 and Lemma 2.9. □

Remark 6.8. *The GFR code associated to a ρ-uniform linear hypergraph is GFR code [137]. For a ρ-uniform linear hypergraph, the Theorem 6.4 follows from the bound on maximum file size stored in a GFR code [137, Theorem 4].*

Remark 6.9. *In [123, Fact 1], it has been shown that an IFR code is equivalent to a ρ-uniform hypergraph.*

6.3.6 Bounds for GFR Codes using Regular Hypergraphs

As introduced in Chapter 2, an s-regular hypergraph is a hypergraph $(V, \mathcal{E})$ such that $|\mathcal{E}(v_i)| = s$, for each $v_i \in V$. A hypergraph $(V, \mathcal{E})$ is called an s-regular hypergraph if $|\mathcal{E}(v_i)| = s$, for each $v_i \in V$. A GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ associated with an s-regular hypergraph $(V, \mathcal{E})$ will have the same node storage capacity for each node *i.e.*, $\alpha = \alpha_i$ for each $i = 1, 2, \ldots, n$. The following three Lemmas give bound on the GFR codes.

Lemma 6.12. *A GFR code $\mathscr{C} : (n, \theta, \alpha, \vec{\rho})$ associated with a connected α-regular hypergraph, satisfies the following conditions.*

1. $\sum_{j=1}^{\theta} \rho_j$ *is a multiple of* α,
2. $\sum_{j=1}^{\theta} \rho_j \geq \frac{\alpha(\theta-1)}{\alpha-1}$,
3. $\rho \leq \frac{1}{\alpha} \sum_{j=1}^{\theta} \rho_j$.

Proof. The proof follows from the dual statement (with respect to hypergraphs) of Lemma 6.4. □

Lemma 6.13. *A GFR code $\mathscr{C} : (n, \theta, \alpha, \vec{\rho})$ with $|U_i \cap U_j| \leq 1$ for each $i, j = 1, 2, \ldots, n$ satisfies $n \leq \frac{\theta(\theta-1)}{\alpha(\alpha-1)}$.*

Proof. The proof follows from the dual property of Lemma 6.11. □

A hypervertex of an s-regular hypergraph $(V, \mathcal{E})$, shares s distinct hyperedges, and hence, $|V| \leq \binom{|\mathcal{E}|}{s}$. The following Lemma ensures the existence of a GFR code with the same number of nodes and packets and same packet replication factor and node storage capacity.

Lemma 6.14. *For a positive integer r, there exists a GFR code $\mathscr{C} : (n, n, r, r)$ such that $2 \leq r \leq n-1$.*

Proof. The proof follows from the fact that there exists an r-uniform r-regular hypergraph with n hypervertices for $2 \leq r \leq n-1$. □

The node packet distribution of the GFR code $\mathscr{C} : (5, 5, r, r)$ with 5 nodes are the following for $r = 2, 3, 4$.

- There exists a GFR code $\mathscr{C} : (5, 5, 2, 2)$ with $U_1 = \{P_1, P_2\}$, $U_2 = \{P_2, P_3\}$, $U_3 = \{P_3, P_4\}$, $U_4 = \{P_4, P_5\}$ and $U_5 = \{P_5, P_1\}$ for $r = 2$.
- There exists a GFR code $\mathscr{C} : (5, 5, 3,)$ with $U_1 = \{P_1, P_2, P_3\}$, $U_2 = \{P_2, P_3, P_4\}$, $U_3 = \{P_3, P_4, P_5\}$, $U_4 = \{P_4, P_5, P_1\}$ and $U_5 = \{P_5, P_1, P_2\}$ for $r = 3$.
- There exists a GFR code $\mathscr{C} : (5, 5, 4, 4)$ with $U_1 = \{P_1, P_2, P_3, P_4\}$, $U_2 = \{P_2, P_3, P_4, P_5\}$, $U_3 = \{P_3, P_4, P_5, P_1\}$, $U_4 = \{P_4, P_5, P_1, P_2\}$ and $U_5 = \{P_5, P_1, P_2, P_3\}$ for $r = 4$.

Remark 6.10. *Each GFR code $\mathscr{C} : (n, \theta, \alpha, \rho)$ considered in [26] is associated with a ρ-uniform α-regular hypergraph $(V, \mathcal{E})$.*

Recall that the storage capacity of nodes in a GFR code is related to the replication factor of packets in the corresponding dual GFR code and vise versa. Following two Lemmas are based on the properties of duality.

Lemma 6.15. *If $\mathscr{C} : (n, \theta, \alpha, \vec{\rho})$ is a GFR code then the dual GFR code $\mathscr{C}^* : (n^*, \theta^*, \vec{\alpha}^*, \rho^*)$ is a GFR code with $\rho^* = \alpha$.*

Proof. The lemma follows from the fact that the dual of an r-uniform hypergraph is an r-regular hypergraph [17]. □

Lemma 6.16. *If $\mathscr{C} : (n, \theta, \vec{\alpha}, \rho)$ is a GFR code then the dual GFR code $\mathscr{C}^* : (n^*, \theta^*, \alpha^*, \vec{\rho}^*)$ is a GFR code with $\alpha^* = \rho$.*

Proof. The lemma follows from the fact that the dual of an s-regular hypergraph is an s-uniform hypergraph [17]. □

Remark 6.11. *In [57], the transpose code of a GFR code is a GFR code, corresponds to a dual hypergraph. All the GFR codes in [57], are corresponds to a ρ-uniform α-regular hypergraph. So, any transpose code corresponds to an α-uniform ρ-regular hypergraph.*

All the bounds in Lemma 6.6, 6.7, 6.13 and 6.11 are new bounds for the universally good GFR codes with asymmetric parameters as the best known to the authors. In [26, 77], GFR codes with symmetric parameters satisfy the bounds in Lemma 6.6, 6.7 and 6.9 with equality. It would be an interesting task to find more GFR codes with asymmetric parameters which satisfy Lemma 6.4, 6.5, 6.9, 6.6, 6.7, 6.13 and 6.11 with equality.

6.3.7 GFR Codes and Hypergraphs

Each GFR code studied in the literature is associated with a specific hypergraph. The various GFR codes and equivalent hypergraphs are listed in the Table 6.2. FR code as the Hyper-FR image of the listed hypergraph (Table 6.2), satisfies the constraints given in the definition of a GFR code in the respective article.

Table 6.2: GFR codes and respective hypergraphs.

GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$	Corresponding hypergraph $(V, \mathcal{E})$
GFR code [26, 103, 11]	ρ-uniform and α-regular hypergraph
HFR code [133]	Linear and α-regular hypergraph
IFR code [123]	ρ-uniform hypergraph
VFR code [131]	α-regular hypergraph with $\lvert E\rvert \in \{\rho_1, \rho_2\}$, for each $E \in \mathcal{E}$
GFR code [132, 137]	ρ-uniform and linear hypergraph
WFR code [38]	ρ-uniform hypergraph
GFR code [81, 56, 108] [128, 135, 120, 57, 77]	ρ-uniform, α-regular and linear hypergraph
Adaptive GFR code [129]	ρ-uniform α-regular hypergraph $s.t.$ induced sub-hypergraph is again ρ'-uniform α'-regular hypergraph
FFR code [134]	Linear hypergraph
GFR code [85]	ρ-uniform hypergraph
GFR code [15]	Hypergraph
GFR code [127, 10]	ρ-uniform and α-regular hypergraph
GFR code [104], [74, 75]	Intersecting, ρ-uniform and α-regular hypergraph
Locally repairable GFR code [69, 130, 71, 70]	ρ-uniform, α-regular and and non-linear hypergraph
FRB codes [102, 105, 103, 104]	ρ-uniform and α-regular hypergraph

CHAPTER 7

Further Properties of Generalized Fractional Repetition Codes

---

Math is a language that you use to describe statistics, but really it's about collecting information and putting it in an order that makes sense. -*Lauren Stamile* [1]

For any code, bounds help to estimate the range of optimal parameters. For any code, the code rate measures the redundancy added per symbol. In this chapter, bounds on maximum file size, GFR code rate, and DSS code rate are calculated for GFR codes.

7.1 Bounds on Generalized Fractional Repetition codes

In this section, for a GFR code, the bounds on file size stored in a GFR code is discussed in Lemma 7.1 and Lemma 7.2. Further, the existence of dual of a GFR code is discussed in the Lemma 6.1. A bound on distinct non-equivalent GFR codes are discussed in Lemma 7.1 and Lemma 7.2.

Lemma 7.1. *Consider a GFR code $\mathscr{C}$ with n nodes and reconstruction degree k. If the node storage capacity of node U_i is α_i and $\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n$, then the maximum file size $M(k)$ stored in the GFR code is bounded as*

$$M(k) \geq \sum_{i=1}^{k} \alpha_i - c\binom{k}{2}, \tag{7.1}$$

where any two distinct nodes have at-most c common packets and any 3 distinct nodes do not have any common packets.

Proof. For a GFR code, any pair of distinct nodes contains at-most c distinct packets. So, from the inclusion–exclusion principle from set theory, the total number of distinct packets in any k nodes is bounded by the difference of the total number of packets stored in those k nodes and the sum of common packets in any pair of nodes *i.e.*,

$$M(k) \geq \sum_{i \in I} \alpha_i - c\binom{k}{2},$$

for any $I \subset \{1, 2, \ldots, n\}$, where $|I| = k$. If $I = \{1, 2, \ldots, k\}$ and $\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_k$ then we obtained the desired lower bound. □

Lemma 7.2. *Let $\mathscr{C}$ be a GFR code with n nodes and reconstruction degree k. If ρ_j ($j \in \{1, 2, \ldots, \theta\}$) is the replication factor for the packet P_j then the maximum file size $M(k)$, for the GFR code, is bounded as*

$$M(k) \leq \left\lfloor \sum_{j=1}^{\theta} \left(1 - \frac{\binom{n-\rho_j}{k}}{\binom{n}{k}} \right) \right\rfloor. \tag{7.2}$$

Proof. The proof is motivated by the proof of the Lemma 14 in [26]. Let $\mathcal{S} = \{S_I = \cup_{i \in I} U_i : I \subset \{1, 2, \ldots, n\}, |I| = k\}$ be a collection of packet sets such that the packet set is contained in k-subsets of node set collectively. If the average cardinality of the sets in $\mathcal{S}$ is denoted by $\overline{S}$, then

$$\sum_{S_I \in \mathcal{S}} |S_I| = \binom{n}{k} \overline{S}. \tag{7.3}$$

But, an arbitrary packet P_j ($j \in \{1, 2, \ldots, \theta\}$) is the member of exactly $\binom{n}{k} - \binom{n-\rho_j}{k}$ distinct sets in $\mathcal{S}$. Hence,

$$\sum_{S_I \in \mathcal{S}} |S_I| = \sum_{j=1}^{\theta} \left(\binom{n}{k} - \binom{n-\rho_j}{k} \right). \tag{7.4}$$

Since, any k nodes have sufficient packets to reconstruct file so, $M(k) \leq |S_I|$ for

each $S_I \in \mathcal{S}$. Hence,

$$M(k) \leq \overline{S} = \left\lfloor \sum_{j=1}^{\theta} \left(1 - \frac{\binom{n-\rho_j}{k}}{\binom{n}{k}} \right) \right\rfloor. \tag{7.5}$$

□

Note that the Theorem 4 in [137] is a special case of the Lemma 7.1, and Lemma 14 in [26] is the special case of the Lemma 7.2.

Theorem 7.1. *If $A(n,\theta)$ is a set of all distinct GFR codes defined on n nodes and θ packets such that any two GFR codes in $A(n,\theta)$ are not equivalent then*

1. $|A(n+1,\theta)| \leq (2^{\theta}-1)|A(n,\theta)|$ *and*
2. $|A(n,\theta+1)| \leq (2^{n}-1)|A(n,\theta)|$.

Proof. One can construct a GFR code by adding a node or/and a packet into a GFR code $\mathscr{C} : (n,\theta,\vec{\alpha},\vec{\rho})$. A binary matrix represents each GFR code. One can construct a binary matrix (NPDI Matrix) by adding a binary row (a node) or/and a binary column (a packet) into a given NPDI Matrix (given GFR code) recursively. The addition follows the following steps recursively in a suitable order.

1. (Adding a row) For a given θ, there are $\sum_{j=1}^{\theta} \binom{\theta}{j} = 2^{\theta}-1$ distinct non-zero rows can be added into a given binary matrix (NPDI Matrix). Hence,

$$|A(n+1,\theta)| \leq (2^{\theta}-1)|A(n,\theta)|.$$

2. (Adding a column) For a given θ, there are $\sum_{i=1}^{n} \binom{n}{i} = 2^{n}-1$ distinct non-zero columns can be added into a given binary matrix (NPDI Matrix). Hence,

$$|A(n,\theta+1)| \leq (2^{n}-1)|A(n,\theta)|.$$

□

Theorem 7.2. *If $A(n,\theta)$ is a set of all distinct GFR codes defined on n nodes and θ packets such that*

1. *any two GFR codes in $A(n,\theta)$ are not equivalent, and*

2. *for any GFR code in $A(n,\theta)$, both the minimum node storage capacity and the minimum packet replication factor are at-least 2,*

then

1. $|A(n+1,\theta)| \leq (2^{\theta}-\theta-1)|A(n,\theta)|$ *and*

2. $|A(n,\theta+1)| \leq (2^{n}-n-1)|A(n,\theta)|$.

Proof. One can construct a GFR code $\mathscr{C} : (n',\theta',\vec{\alpha}',\vec{\rho}')$ by adding a node or/and a packet into a GFR code $\mathscr{C} : (n,\theta,\vec{\alpha},\vec{\rho})$. A binary matrix represents each GFR code. One can construct a binary matrix (NPDI Matrix) by adding a binary row (a node) or/and a binary column (a packet) into a given NPDI Matrix (given GFR code) recursively. The addition follows the following steps recursively in a suitable order.

1. (Adding a row) For a given θ, there are $\sum_{j=2}^{\theta}\binom{\theta}{j} = 2^{\theta}-\theta-1$ distinct rows that can be added into a given binary matrix (NPDI Matrix), where the added row contains at-least two number of ones. Hence,

$$|A(n+1,\theta)| \leq (2^{\theta}-\theta-1)|A(n,\theta)|.$$

2. (Adding a column) For a given θ, there are $\sum_{i=2}^{n}\binom{n}{i} = 2^{n}-n-1$ distinct non-zero columns that can be added into a given binary matrix (NPDI Matrix), where the added column contains at-least two number of ones. Hence,

$$|A(n,\theta+1)| \leq (2^{n}-n-1)|A(n,\theta)|.$$

□

7.2 Properties of Code Rate

Consider a GFR code $\mathscr{C} : (n,\theta,\vec{\alpha},\vec{\rho})$ with replication factor $\rho_j = 2$ (for each $j \in \{1,2,\ldots,\theta\}$) on a $(\theta, M(k))$ MDS code. The GFR code rate approaches to $1/2$, if

$M(k)/\theta$ approaches to 1. Note that the ratio $M(k)/\theta$ will approach 1 if the values of both the maximum file size $M(k)$ and the total number of distinct encoded packets θ are very close and are very high. Recall that the tolerance factor for such cases of the GFR code is $\rho - 1 = 1$. Hence, the code rate of a reliable GFR code is bounded by $1/2$. For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, the code rate

$$\mathcal{R}_{\mathscr{C}}(k) = \frac{M(k)}{\sum_{j=1}^{\theta} \rho_j} \geq \frac{\alpha_{min}}{\sum_{j=1}^{\theta} \rho_j} \geq \frac{\alpha_{min}}{\rho\theta}.$$

For any GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$, the growth of the GFR code rate is

$$\frac{\mathcal{R}_{\mathscr{C}}(k)}{\mathcal{R}_{\mathscr{C}}(k-1)} = \frac{M(k)}{M(k-1)} \leq \frac{\sum_{i=1}^{k} \alpha_i}{\alpha_{min}},$$

where $\alpha_{min} = min\{\alpha_i : i = 1, 2, \ldots, n\}$ and $k = 2, 3, \ldots n-1$.

For a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ defined on (n, k) DSS, the following Lemma gives a relation between the GFR code rate and the DSS code rate.

Lemma 7.3. *For a GFR code, the GFR code rate is bounded by the DSS code rate.*

Proof. For a GFR code, let a file, with size $M(k)$, be stored in a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$. Since $\frac{1}{k}M(k) \leq \frac{1}{n}\sum_{i=1}^{n} \alpha_i$,

$$\mathcal{R}_{\mathscr{C}}(k) = \frac{M(k)}{\sum_{i=1}^{n} \alpha_i} \leq \frac{k}{n}\left(\frac{\frac{1}{k}M(k)}{\frac{1}{n}\sum_{i=1}^{n} \alpha_i}\right) < \frac{k}{n} = \mathcal{R}_{DSS}(k).$$

□

Theorem 7.3. *For an (n, k) DSS, consider a GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ with $|U_i \cap U_j| \leq 1$ (for each $1 \leq i < j \leq n$). Then, the rate difference*

$$\mathcal{R}_{DSS}(k) - \mathcal{R}_{\mathscr{C}}(k) \leq \frac{1}{\sum_{i=1}^{n} \alpha_i}\binom{k}{2}.$$

Proof. For any $i, j \in [n]$ and $1 \leq i < j \leq n$, let a GFR code $\mathscr{C} : (n, \theta, \alpha, \rho)$ (with $|U_i \cap U_j| \leq 1$) be defined on (n, k) DSS. If α_{ave} is the average node storage capacity

on nodes, then the code rate difference

$$\begin{aligned}\mathcal{R}_{DSS}(k) - \mathcal{R}_{\mathscr{C}}(k) &= \frac{k\frac{1}{n}\sum_{i=1}^{n}\alpha_i - M(k)}{\sum_{i=1}^{n}\alpha_i} \\ &= \frac{k\alpha_{ave} - M(k)}{\sum_{i=1}^{n}\alpha_i}.\end{aligned}$$

Without loss of generality, let $\alpha_i \geq \alpha_j$ for $1 \leq i < j \leq n$. Hence,

$$k\alpha_{ave} \leq \sum_{i=1}^{k}\alpha_i.$$

Therefore,

$$\mathcal{R}_{DSS}(k) - \mathcal{R}_{\mathscr{C}}(k) \leq \frac{1}{\sum_{i=1}^{n}\alpha_i}\left(\sum_{i=1}^{k}\alpha_i - M(k)\right).$$

Note that $\sum_{i=1}^{n}\alpha_i - M(k)$ are the total number of common packets such that every common packet is shared by a pair of the nodes from the node set $\{U_1, U_2, \ldots, U_k\}$. Hence,

$$\mathcal{R}_{DSS}(k) - \mathcal{R}_{\mathscr{C}}(k) \leq \frac{1}{\sum_{i=1}^{n}\alpha_i}\binom{k}{2}.$$

□

Consider a GFR code $\mathscr{C} : (5, 10, 4, 2)$ with the node packet distribution as given in the Table 7.1. For $k = 1, 2, 3, 4$, the GFR code rate $\mathcal{R}_{\mathscr{C}}(k)$, DSS code rate k/n and difference between both the code rates are calculated in Table 7.2.

Table 7.1: The node packet distribution for the GFR code $\mathscr{C} : (5, 10, 4, 2)$.

U_1	P_1, P_2, P_3, P_4
U_2	P_1, P_5, P_6, P_7
U_3	P_2, P_5, P_8, P_9
U_4	P_3, P_6, P_8, P_{10}
U_5	P_4, P_7, P_9, P_{10}

For another example, consider a GFR code $\mathscr{C} : (6, 6, 2, 2)$ with the node packet distribution as given in the Table 7.3. For $k = 2, 3, 4, 5$, the code rate $\mathcal{R}_{\mathscr{C}}(k)$, the DSS code rate $\mathcal{R}_{DSS}(k)$ and the difference $\mathcal{R}_{DSS}(k) - \mathcal{R}_{\mathscr{C}}(k)$ are calculated in Table 7.4.

In this Chapter, we have given some constructions of universally good GFR code by concatenating two GFR codes with different packet symbols. The resultant

Table 7.2: For $k = 2, 3, 4$, the $(5, k)$ DSS code rate and the GFR code rate of the GFR code $\mathscr{C} : (5, 10, 4, 2)$.

Max. Reconstruction Degree k	DSS Code Rate $\mathcal{R}_{DSS}(k)$	Code Rate of GFR Code $\mathcal{R}_{\mathscr{C}}(k)$	The Rate Difference $k(k-1)/(2n\alpha)$
1	0.200	0.200	0.000
2	0.400	0.350	0.050
3	0.600	0.450	0.150
4	0.800	0.500	0.300

Table 7.3: The node packet distribution for the GFR code $\mathscr{C} : (6, 6, 2, 2)$.

U_1	P_1, P_2
U_2	P_2, P_3
U_3	P_3, P_4
U_4	P_4, P_5
U_5	P_5, P_6
U_6	P_1, P_6

GFR code is called concatenated GFR code. The code rate and its growth are discussed in the following theorem.

Theorem 7.4. *Consider a GFR code $\mathscr{C}^{(1)} : (n^{(1)}, \theta^{(1)}, \vec{\alpha}^{(1)}, \vec{\rho}^{(1)})$ with NPDI Matrix $B^{(1)}_{n\times\theta}$. For a positive integer m and $r < m$, let $\mathscr{C}^{(m)} : (n^{(m)} = mn, \theta^{(m)} = m\theta, \vec{\alpha}^{(1)}, \vec{\rho}^{(m)} = \vec{\rho})$ be a GFR code with NPDI Matrix $M^{(m)}$, where the matrix*

$$B^{(r+1)} = \begin{bmatrix} B^{(1)} & \boldsymbol{0} \\ \boldsymbol{0} & B^{(r)} \end{bmatrix}.$$

For a positive integer $K < n^{(m)}$ and $k \equiv K \pmod{n^{(1)}}$,

1. *the DSS code rate*
$$\mathcal{R}_{DSS}(K) = \frac{1}{m}\left(\left\lfloor \frac{K}{n^{(m)}} \right\rfloor + \frac{k}{n}\right),$$

2. *the GFR code rate*
$$\mathcal{R}_{\mathscr{C}^{(m)}}(K) = \frac{1}{m}\left(\left\lfloor \frac{K}{n^{(m)}} \right\rfloor \frac{1}{\rho_{ave}} + \mathcal{R}_{\mathscr{C}}(k)\right),$$

Table 7.4: For $k = 1, 2, 3, 4, 5$, the $(6, k)$ DSS code rate and the GFR code rate of a GFR code $\mathscr{C} : (6, 6, 2, 2)$.

Max. Reconstruction Degree k	DSS Code Rate $\mathcal{R}_{DSS}(k)$	GFR Code Rate $\mathcal{R}_{\mathscr{C}}(k)$	The Rate Difference $(k-1)/(2n)$
1	0.167	0.167	0
2	0.333	0.250	0.083
3	0.500	0.333	0.167
4	0.667	0.417	0.250
5	0.833	0.500	0.333

3. *and the rate difference*

$$\begin{aligned}&\mathcal{R}_{DSS}(K) - \mathcal{R}_{\mathscr{C}^{(n)}}(K)\\ &\qquad = \frac{1}{m}\left\{\mathcal{R}_{DSS}(k)\left(1 - \frac{1}{\rho_{ave}}\right) - \mathcal{R}_{\mathscr{C}}(k)\right\},\end{aligned}$$

where $\mathcal{R}_{DSS}(k)$ is $(n^{(1)}, k)$ DSS code rate and

$$\rho_{ave} = \frac{\sum_{i=1}^{n}\alpha_i}{\theta} = \frac{\sum_{j=1}^{\theta}\rho_j}{\theta}.$$

Proof. For $N = nm$ and $K = n\lfloor K/n \rfloor + k$, one can easily find the DSS code rate $\mathcal{R}_{DSS}(K)$, the GFR code rate $\mathcal{R}_{\mathscr{C}(K)}$ and the rate difference $\mathcal{R}_{DSS}(K) - \mathcal{R}_{\mathscr{C}'}(K)$. □

In a GFR code $\mathscr{C}^{(m)} : (n^{(m)} = mn, \theta^{(m)} = m\theta, \vec{\alpha}^{(1)}, \vec{\rho}^{(m)} = m\vec{\rho})$, if the average of the replication factor to the packets is

$$\rho_{ave}^{(m)} = \frac{\sum_{i=1}^{n^{(m)}}\alpha_i^{(m)}}{\theta^{(m)}},$$

then $\rho_{ave}^{(m)} = \rho_{ave}^{(1)}$, for any $m \in \mathbb{N}$.

For $r = 1, 2$, if two GFR codes $\mathscr{C}^{(r)} : (n^{(r)}, \theta^{(r)}, \vec{\alpha}^{(r)}, \vec{\rho}^{(r)})$ are universally good then the concatenated GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ is also universally good.

For any $m \in \mathbb{N}$, if the GFR code $\mathscr{C}^{(1)} : (n^{(1)}, \theta^{(1)}, \vec{\alpha}^{(1)}, \vec{\rho}^{(1)})$ is universally good then the constructed GFR code $\mathscr{C}^{(m)} : (n^{(m)}, \theta^{(m)}, \vec{\alpha}^{(m)}, \vec{\rho}^{(m)})$ (see Theorem 7.4) is also universally good, where $n^{(m)} = mn, \theta^{(m)} = m\theta, \vec{\alpha}^{(m)} = \vec{\alpha}^{(1)}$ and $\vec{\rho}^{(m)} = m\vec{\rho}$.

For a positive integer $\alpha > 1$, consider a matrix

$$W_{\alpha+1} = \begin{bmatrix} \mathbf{1}_\alpha & \mathbf{0}_{\binom{\alpha}{2}} \\ I_\alpha & W_\alpha \end{bmatrix},$$

where $\mathbf{0}_m$ is an zero-array with length m, $\mathbf{1}_m$ is an m-length array with each entry one, I_m is an identity matrix of dimension m and $W_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ for $m > 1$ and $m \in \mathbb{N}$. a GFR code $\mathscr{C} : (n = \alpha + 1, \theta = \binom{\alpha+1}{2}), \vec{\alpha}, \vec{\rho} = (2\,2 \ldots 2)$ with the NPDI Matrix $W_{\alpha+1}$ is universally good [26]. For the GFR code, the difference between the DSS code rate $\mathcal{R}_{DSS}(k)$ and the GFR code rate $\mathcal{R}_{\mathscr{C}}(k)$ is $k(k-1)/(2n\alpha)$. Note that the difference increases with $\mathcal{O}(k^2)$.

7.3 Reconstruction and Repair Degree of Generalized Fractional Repetition Codes

Given a Fractional Repetition (FR) code with symmetric parameters, calculating the reconstruction degree and repair degree in a DSS is a significant problem. In this section, we present algorithms for approximating the reconstruction and repair degree of GFR Codes.

Given a (n, k, d) DSS, one has to find a good GFR code $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ which matches with the parameters of DSS. Note that the parameter k in DSS is known as the reconstruction degree of DSS. If one wants to get the entire file, then one has to contact any k nodes in DSS. However, if we look at the definition of GFR code $\mathscr{C}$ one finds that it is independent of k (there is no direct formula for calculating the reconstruction degree). This motivates us to define the reconstruction degree of GFR code $\mathscr{C}$ as the the number k_{FR} so that if one wants the entire file (total $(\theta - 1)$ packets as one remaining packet one can get using MDS code) one has to contact the smallest set of any k_{FR} nodes in GFR code. Clearly, $k \leq k \leq k_{FR}$. To find the value k_{FR} of a GFR code we also define another reconstruction degree k of GFR code as the smallest subset of nodes of $\mathscr{C}$, which allows recovering the entire data (all $\theta - 1$ packets). Clearly, we also have $k \leq k_{FR}$. We present an Algorithm 2 to

compute k. It gives a lower bound on actual k_{FR}.

To find the reconstruction degree of a GFR code, one can always delete one packet from all the nodes (W.L.O.G., we usually delete the last packet θ) as we can recover it using the parity of MDS codes. Hence, for constructing entire data, it is sufficient to reconstruct only $(\theta - 1)$ packets. Thus WLOG we delete the last packet θ in the algorithm 2.

Algorithm 2 Algorithm to compute reconstruction degree k^*.

Require: Node packet distribution of GFR code after removing the last packet θ from all n nodes of $V^n = \{V_1, V_2, \dots V_n\}$.

Ensure: Reconstruction degree k^*_{upp}.

1: For $1 \leq i, j, m \leq n$, if $\exists\ V_i$ & V_j $s.t.$ $V_j \subseteq V_i$ then delete all such V_j for all possible nodes V_i and list remaining collection of nodes as $V^m = \{V_{i_1}, V_{i_2}, \dots, V_{i_m}\}$, $|V_{i_j}| = \alpha_{i_j}$ = number of packets in node V_{i_j}.

2: Let $V^l = \{V_{i_j} \in V^m : 1 \leq j \leq m$ & $|V_{i_j}| = max\{\alpha_{i_j}\}\}$.

3: Pick an arbitrary set $V_{i_j} \in V^l$, and call this set as P. Set the counter $k_\lambda = 1$, $1 \leq k_\lambda \leq m$ and $1 \leq \lambda \leq |V^l| = l$.

4: If $\exists\ V_{i_{j'}}\ (1 \leq j' \leq m) \in V^m\ s.t.\ V_{i_{j'}} \cap P = \emptyset$ then go to step 5 otherwise jump to step 6.

5: Pick $V_{i_{j''}}\ (1 \leq j'' \leq m) \in V^m$ which has max cardinality among all $V_{i_{j''}}$ in V^m with $V_{i_{j''}} \cap P = \emptyset$. Update $P = P \cup V_{i_{j''}}$, update counter $k_\lambda = (k_\lambda + 1)$ and go to step 4.

6: If $\exists V_{i_r}\ (1 \leq r \leq m) \in V^m\ s.t.\ V_{i_r} \not\subset P$ then go to step 7 otherwise go to step 8.

7: Pick $V_{i_{r'}}\ (1 \leq r' \leq m) \in V^m$ which has maximum $|V_{i_{r'}} \backslash P|$ among all $V_{i_{r'}} \in V^m$ having the condition $V_{i_{r'}} \not\subset P$ then update $P = P \cup V_{i_{r'}}$, update counter $k_\lambda = (k_\lambda + 1)$ and go to step 6.

8: If $1 \leq \lambda < l$, then store k_λ in $k_{\lambda'}$ and set $k_\lambda = k_{(\lambda+1)}$ and perform step 4 for $P = V_{i_{j'''}}\ (1 \leq j''' \leq m) \in V^l\ s.t.\ V_{i_{j'''}} \neq V_{i_j} \in V^l$, otherwise report $k^*_{upp} = min\{k'_\lambda\}_{\lambda=1}^{l}$.

Example 7.1. *Consider a GFR code* $\mathscr{C} : (n, \theta, \vec{\alpha}, \vec{\rho})$ *with* $U_1 = \{U_1, U_2, U_3, U_4\}$, $U_2 = \{U_1, U_6, U_9\}$, $U_3 = \{U_2, U_5, U_7, U_9\}$, $U_4 = \{U_3, U_5, U_6, U_8\}$ *and* $U_5 = \{U_4, U_7, U_8\}$.

- *Note that since* $n = 5$, *after removing any packet (say last packet* P_9*) we get* $V^5 = \{V_1, V_2, V_3, V_4, V_5\}$, *where* $V_1 = \{1, 2, 3, 4\}$, $V_2 = \{1, 6\}$, $V_3 = \{2, 5, 7\}$, $V_4 = \{3, 5, 6, 8\}$, $V_5 = \{4, 7, 8\}$ *each having cardinality as* $\{4, 3, 3, 4, 3\}$ *respectively.*

- *Further since there is no set* V^p $s.t.$ $V_p \subseteq V_q\ (1 \leq p, q \leq 5)$ *so step* 1 *yields* $V^m = V^5 = \{V_1, V_2, V_3, V_4, V_5\}$.

- *For step* 2, *note that there are only two sets* V_1, V_4 *of maximum cardinality* 4, *so* $V^l = \{V_1, V_4\}$ *now executing step* 3, *pick an arbitrary node* V_1 *as* $P = V_1$ *and initialise* $k_1 = 1$.

- *Now we skip step* 4, *since there does not exist any set* $V_i \in V^5$ *s.t.* $V_i \cap P = \emptyset$ *and we go to step* 6.

- *At step* 6, *we search* $V_i \in V^5$ *s.t.* $V_i \not\subset P$ *so we get* V_1, V_2, V_3, V_4, V_5.

- *For step* 7 *we have* $V_2 \backslash P = \{6\}$, $V_3 \backslash P = \{5, 7\}$, $V_4 \backslash P = \{5, 6, 8\}$ *and* $V_5 \backslash P = \{7, 8\}$ *among them* $|V_4 \backslash P|$ *is maximum. So* $P = P \cup V_4 = \{1, 2, 3, 4\} \cup \{3, 5, 6, 8\} = \{1, 2, 3, 4, 5, 6, 8\}$ *and* $k_1 = 1 + 1 = 2$.

- *According to step* 6, *again we search* $V_i \in V^5$ *s.t.* $V_i \not\subset P$ *and we get* V_3, V_5. *Now again* $V_3 \backslash P = \{7\}$ *and* $V_5 \backslash P = \{7\}$.

- *By step* 7, $P = P \cup V_5 = \{1, 2, 3, 4, 5, 6\} \cup \{4, 7, 8\} = \{1, 2, 3, 4, 5, 6, 7, 8\}$ *and* $k_1 = 2 + 1 = 3$ *since* $V_3 \backslash P$ *is maximum.*

- *According to step* 8, $k_1' = 3$ *and update* $k_1 = k_2 = 1$ *Compute* k_2' *for* $P = V_4 \in V^2$, $k_2' = 3$.

- *So* $k_{upp}' = min\{k_1, k_2\} = 3$.

Note that in general, Algorithm 2 computes an upper bound on k. However, Algorithm 3 gives an exact value of k *i.e.*, $k_{upp}^* = k = 3$. Algorithm 3 presents a case of GFR code $\mathscr{C} : (5, 8, 4, 2)$ for which $k = 2$ and $k_{upp} = 3$. Further, note that at the cost of complexity, one can modify the algorithm 2 at step 3, by taking P on all possible nodes in V_m to yield an exact reconstruction degree k^*. In particular, for GFR code (with symmetric parameters) this algorithm will always give an exact value of k^*.

Remark 7.1. *In Algorithm 2, 3, and 4, reconstruction degree and repair degrees are approximated for* $M(k) = \theta - 1$. *For some cases, these algorithms estimate exact values, i.e., reconstruction degree and repair degree.*

Algorithm 3 Algorithm to compute reconstruction degree k_{FR}.

Require: A set of packets $\Omega = \{1, 2, \ldots, \theta\}$ and node packet distribution of GFR code with n nodes $U^n = \{U_1, U_2, \ldots, U_n\}$.

Ensure: Exact reconstruction degree k_{FR}.

1: For $1 \leq m \leq n$ set $U^m = \{U_1, U_2, \ldots, U_m\}$. Take $m = n$.

2: Pick the set $U_m \in U^m$ and call this set as P. Set the counter $k_\lambda = 1, 1 \leq k_\lambda \leq m$ and $1 \leq \lambda \leq n$. If $\Omega \backslash P = \emptyset$ or singleton set then go to step 6 otherwise go to step 3.

3: If $\exists\, U_j\ (1 \leq j \leq m) \in U_m\ s.t.\ U_j \cap P = \emptyset$ then go to step 4 otherwise jump to step 5.

4: Pick an arbitrary $U_{j'}\ (1 \leq j' \leq m) \in U^m$ which has maximum cardinality among all $U_{j'}$ in U^m with $U_{j'} \cap P = \emptyset$. Update $P = P \cup U_{j'}$, update counter $k_\lambda = (k_\lambda + 1)$. Again if $\Omega \backslash P = \emptyset$ or singleton set then go to step 6 otherwise go to step 3.

5: Pick $U_r\ (1 \leq r \leq m) \in U^m\ s.t.\ U_r \subset P$ which has maximum $|U_r \backslash P|$ among all $U_r \in U^m$ having the condition $U_r \not\subset P$ then update $P = P \cup U_r$, update counter $k_\lambda = (k_\lambda + 1)$. Once again if $\Omega \backslash P = \emptyset$ or singleton set then go to step 6 otherwise go to step 5.

6: Store k_λ in k'_λ and set $k_\lambda = k_{(\lambda+1)}$.

7: If $1 \leq \lambda < n$ then calculate $U^{m-1} = U^m \backslash \{U_m\}$ and perform step 2 for $P = U_{j''}\ (1 \leq j'' \leq n) \in U^m$, otherwise report $k_{FR} = max\{k'_\lambda\}_{\lambda=1}^{n}$.

Algorithm 4 Algorithm to compute repair degree d_i.

Require: Incidence matrix $B_{n \times \theta}$ of GFR code.

Ensure: Repair degree d_i of node U_i.

1: For each node $i\ (1 \leq i \leq n)$, let $S_i^{\{i\}} = \{H_j \backslash \{i\} : i \in H_j, 1 \leq j \leq \theta\}$

2: Compute $T \subseteq \{1, 2, \ldots, \theta\}\ s.t.\ |T|$ is maximum among all possible subsets and for $t \in T$, $H_t \backslash \{i\} \in S_i^{\{i\}}$, and $\bigcap_t H_t \backslash \{i\} = \emptyset$. Set counter $l_q\ (1 \leq q \leq n) = |T| - 1$.

3: Update $S_i^{\{i\}} = S_i^{\{i\}} \backslash (H_t \backslash \{i\}), \forall t \in T$.

4: If $S_i^{\{i\}} = \emptyset$ then $d_i = \alpha_i - \sum_{\lambda=1}^{q} l_\lambda$, where $\alpha_i = |V_i|$, otherwise set $q = q + 1$ and go to step 2.

Example 7.2. *Consider the following node-packet distribution incidence matrix $B_{11\times 8}$ for GFR code $\mathscr{C}$: $(11, 8, 2, 3)$.*

$$B_{11\times 8} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

According to Algorithm 4 the calculation of repair degree d_i where $i \in \{1, 2, \ldots, 11\}$ for the node-packet distribution incidence matrix $M_{11\times 8}$ is as follows.

- $H_1 = \{1, 8, 5\}$, $H_2 = \{2, 5, 9\}$, $H_3 = \{5, 9, 3\}$, $H_4 = \{8, 1, 5\}$, $H_5 = \{2, 6, 8\}$, $H_6 = \{9, 4, 7\}$, $H_7 = \{10, 7, 1\}$, $H_8 = \{2, 6, 11\}$.
- *If we want to compute repair degree for 5^{th} node (i.e., d_5) then pick the all H_j s.t. $5 \in H_j$ i.e., H_1, H_2, H_3 and H_4.*
- *Now $S_5^{\{5\}} = \{H_1\backslash\{5\}, H_2\backslash\{5\}, H_3\backslash\{5\}, H_4\backslash\{5\}\}$, where $H_1\backslash\{5\} = \{1, 8\}$, $H_2\backslash\{5\} = \{2, 9\}$, $H_3\backslash\{5\} = \{9, 3\}$ and $H_4\backslash\{5\} = \{8, 1\}$.*
- *But $\bigcap_{r\in\{1,2,3,4\}} H_r\backslash\{5\} = \emptyset$ and there is no any common element among any three sets chosen from the $S_5^{\{5\}}$.*
- *But for $T = \{1, 4\}$ we have $H_1\backslash\{5\} \bigcap H_4\backslash\{5\} = \{1, 8\} = \emptyset$ so $l_1 = 2 - 1 = 1$.*
- *Now updated $S_5^{\{5\}}$ is $S_5^{\{5\}} = \{H_2\backslash\{5\}, H_3\backslash\{5\}\}$ then we have $H_2\backslash\{5\} \bigcap H_3\backslash\{5\} = \{9\} = \emptyset$ here $T = \{2, 3\}$ so $l_2 = 2 - 1 = 1$.*
- *Now Repair degree (d_5) = $\alpha_5 - l_1 - l_2 = 2$ where α_5 is weight of 5^{th} row in node-packet distribution incidence matrix $M_{11\times 8}$.*

CHAPTER 8

Conclusions and Future Work

The future belongs to those who believe in beauty of their dreams. -*Eleanor Roosevelt* [1]

In this thesis, we proposed a model of heterogeneous DSS with dynamic reconstruction degree, storage node capacity and repair bandwidth. In particular, at time t, a file can be reconstructed using a specific set of nodes and system is repaired for any failed node by contacting some set of helper nodes. For such heterogeneous DSS, the fundamental trade-off curve between system repair cost and system storage cost is investigated. To plot the trade-off curve, a bi-objective optimization problem is formulated with the constraints of *min-cut* bound and non-negative parameters of the heterogeneous DSS. The bi-objective optimization problem is solved by a weighted sum method for some numerical values of parameters of the heterogeneous model. Analyzing the trade-off curve, we observed some more optimum points than the existing heterogeneous model [122]. The considered model is close to the real world scenario. Our heterogeneous model is flexible enough to mold it into any existing heterogeneous or homogeneous DSS by considering appropriate restrictions. It would be an interesting task to construct codes achieving the optimum points on the trade-off curve.

Further, a novel class of GFR codes based on sequences is introduced in the thesis. Our work opens an excellent connection between GFR codes with a well-known area of sequences. In this thesis, we have calculated the bound for the universally good GFR code using sequences. Universally good GFR codes are

constructed using some families of binary sequences of finite length. It would be interesting to study some more bounds on GFR codes using sequences. Constructing GFR codes using a well-known class of sequences would be another exciting task in the near future. In the last chapter of the thesis, we have sketched the surface of the fascinating topic by establishing a correspondence between GFR code and hypergraph. It is shown that each GFR code studied in the literature is associated with some conditional hypergraph, where, for the hypergraph, size of hyperedges and degree of hypervertices are all non-zero positive integers. A construction of universally good adaptive GFR code with asymmetric parameters is discussed in this work. Bounds on universally good GFR codes are obtained from the properties of hypergraphs. Some new bounds are found on universally good GFR code defined on heterogeneous DSS. It would be an exciting work in the future to find the GFR codes which satisfy those bounds with equality. Analysis of GFR codes on generalized hypergraph could be another interesting problem. It will also be interesting future task to analyse the complexity of Algorithm 2, 3 and 4. In addition, obtaining algorithms for GFR codes to estimate the parameters is an another interesting task in near future.

Publications

Published/Accepted Publications

1. **Krishna Gopal Benerjee**, and Manish K. Gupta and Nikhil Agrawal, "Reconstruction and Repair Degree of Fractional Repetition Codes," IEEE International Symposium on Network Coding (NetCod), Calgary, Canada, June 2013, Extended abstract available: http://arxiv.org/abs/1305.4580 (**poster**).

2. Mit Sheth and **Krishna Gopal Benerjee**, and Manish K. Gupta, "Quorum Sensing for Regenerating Codes in Distributed Storage," Sixth International Conference on Communication Systems and Networks (COMSNETS), IISc Bengaluru, India, January 2014, Extended abstract available: http://arxiv.org/abs/1309.7429 (**poster**).

3. **Krishna Gopal Benerjee**, and Manish K. Gupta, "On Dress Codes with Flowers," In Proceedings of Seventh International Workshop on Signal Design and Its Applications in Communications (IWSDA), IISC Bengaluru, India, September 2015, pp. 108-112.

4. Dixita Limbachiya, **Krishna Gopal Benerjee**, Bansari Rao, and Manish K. Gupta, "Computational Algebraic Prospective of DNA Codes," International School on Computer Algebra (COCOA), IIT Gandhinagar, India, February 2016 (**poster**).

5. Dixita Limbachiya, **Krishna Gopal Benerjee**, Bansari Rao, and Manish K. Gupta, "On DNA Codes using the Ring $Z_4 + wZ_4$," In Proceedings of IEEE International Symposium on Information Theory (ISIT), Colorado, USA, June 2018, pp. 2401-2405.

6. **Krishna Gopal Benerjee**, and Manish K. Gupta, "On Universally Good Flower Codes," Sequences and Their Applications (SETA), Hong Kong University of Science and Technology, Hong Kong, October 2018, Extended abstract available: https://arxiv.org/abs/1805.08144.

Communicated

1. **Krishna Gopal Benerjee**, and Manish K. Gupta, "On Non-uniform Flower Codes," Cryptography and Communications: Discrete Structures, Boolean Functions and Sequences, January 2019 (submitted).

2. **Krishna Gopal Benerjee**, Sourav Deb, and Manish K. Gupta, "On Conflict Free DNA Codes," IEEE Transactions on Information Theory, March 2019, Extended abstract available: https://arxiv.org/abs/1902.04419 (submitted).

3. **Krishna Gopal Benerjee**, and Manish K. Gupta, "Bounds on Generalized FR Codes using Hypergraphs," Electronic Journal of Graph Theory and Applications, June 2019 Extended abstract available: https://arxiv.org/abs/1711.07631 (submitted).

Foram/Presentation

1. **Krishna Gopal Benerjee** "On Optimal Secure Weak Regenerating Codes and Its Applications," Sixth International Conference on Communication Systems and Networks (COMSNETS), IISc Bengaluru, India, Jan 2014, Ph.D. Workshop.

Technical Reports

1. **Krishna Gopal Benerjee**, and Manish K. Gupta, "On Heterogeneous Regenerating Codes and Capacity of Distributed Storage Systems," *ArXiv e-prints*, cs.IT, 2014. [Online] Available: http://arxiv.org/abs/1402.3801.

2. **Krishna Gopal Benerjee**, and Manish K. Gupta, "On Optimal Heterogeneous Regenerating Codes," *ArXiv e-prints*, cs.IT, 2016. [Online] Available: http://arxiv.org/abs/1607.06313.

3. **Krishna Gopal Benerjee**, and Manish K. Gupta, "On Code Rates of Fractional Repetition Codes," *ArXiv e-prints*, cs.IT, 2017. [Online] Available: http://arxiv.org/abs/1711.08869.

References

[1] Brainy Quote (14 august 2018) https://www.brainyquote.com/.

[2] R. Ahlswede, N. Cai, S. Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, July 2000.

[3] I. Ahmad and C. Wang. Locally repairable regenerating codes: Node unavailability and the insufficiency of stationary local repair. *IEEE Transactions on Information Theory*, 64(5):3493–3512, May 2018.

[4] I. Ahmad and C. C. Wang. When and by how much can helper node selection improve regenerating codes? In *Proc. of 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 459–466, September 2014.

[5] I. Ahmad and C. C. Wang. When can intelligent helper node selection improve the performance of distributed storage networks? *IEEE Transactions on Information Theory*, 64(3):2142–2171, March 2018.

[6] S. Akhlaghi, A. Kiani, and M. Ghanavati. A fundamental trade-off between the download cost and repair bandwidth in distributed storage systems. In *Proc. of International Symposium on Network Coding (NetCod)*, pages 1–6, June 2010.

[7] S. Akhlaghi, A. Kiani, and M. R. Ghanavati. Cost-bandwidth tradeoff in distributed storage systems. *Computer Communications*, 33(17):2105 – 2115, November 2010. Special Issue: Applied sciences in communication technologies.

[8] Amazon. Amazon elastic compute cloud (Amazon EC2). January 2013.

[9] L. S. amd Oggier F. *An Overview of Coding for Distributed Storage Systems. In: Greferath M., Pavčević M., Silberstein N., Vázquez-Castro M. (eds) Network Coding and Subspace Designs*. Signals and Communication Technology. Springer, Cham, January 2018.

[10] S. Anil, M. K. Gupta, and T. A. Gulliver. Enumerating Some Fractional Repetition Codes. *ArXiv e-prints*, March 2013, [Online] Available: http://arxiv.org/abs/1303.6801.

[11] H. Aydinian and H. Boche. Fractional repetition codes based on partially ordered sets. In *Proc. of IEEE Information Theory Workshop (ITW)*, pages 51–55, November 2017.

[12] S. B. Balaji, M. Nikhil Krishnan, M. Vajha, V. Ramkumar, B. Sasidharan, and P. Vijay Kumar. Erasure Coding for Distributed Storage: An Overview. *ArXiv e-prints*, June 2018, [Online] Available: https://arxiv.org/abs/1806.04437.

[13] A. Barg and G. Zemor. Codes on hypergraphs. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 156–160, July 2008.

[14] K. G. Benerjee and M. K. Gupta. On Heterogeneous Regenerating Codes and Capacity of Distributed Storage Systems. *ArXiv e-prints*, February 2014, [Online] Available: https://arxiv.org/abs/1402.3801.

[15] K. G. Benerjee and M. K. Gupta. On dress codes with flowers. In *Proc. of Seventh International Workshop on Signal Design and its Applications in Communications (IWSDA)*, pages 108–112, September 2015.

[16] K. G. Benerjee, M. K. Gupta, and N. Agrawal. Reconstruction and Repair Degree of Fractional Repetition Codes. *ArXiv e-prints*, May 2013, [Online] Available: https://arxiv.org/abs/1305.4580.

[17] C. Berge. *Hypergraphs*, volume 45 of *North-Holland Mathematical Library*. North-Holland, August 1989. Combinatorics of Finite Sets.

[18] M. Berkelaar, P. Eikland, and P. Notebaert. lp_solve-a mixed integer linear programming (milp) solver. *Version 5.5.2.0*, 2011.

[19] Y. Bilu and S. Hoory. On codes from hypergraphs. *European Journal of Combinatorics*, 25(3):339 – 354, April 2004.

[20] A. Bretto. *Hypergraph Theory An introduction*. Springer International Publishing, 2013.

[21] J. Chen and K. W. Shum. Repairing multiple failures in the suh-ramchandran regenerating codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1441–1445, July 2013.

[22] A. G. Dimakis, P. B. Godfrey, M. J. Wainwright, and K. Ramchandran. Network coding for distributed storage systems. In *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, pages 2000–2008, May 2007.

[23] 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*, 56(9):4539–4551, September 2010.

[24] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh. A survey on network codes for distributed storage. *Proceedings of the IEEE*, 99(3):476–489, March 2011.

[25] I. M. Duursma. Outer bounds for exact repair codes. *ArXiv e-prints*, June 2014, [Online] Available: https://arxiv.org/abs/1406.4852.

[26] S. El Rouayheb and K. Ramchandran. Fractional repetition codes for repair in distributed storage systems. In *Proc. of 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1510 –1517, October 2010.

[27] P. Elias, A. Feinstein, and C. Shannon. A note on the maximum flow through a network. *IRE Transactions on Information Theory*, 2(4):117–119, December 1956.

[28] T. Ernvall. The existence of fractional repetition codes. *ArXiv e-prints*, January 2012, [Online] Available: https://arxiv.org/abs/1201.3547.

[29] T. Ernvall, S. El Rouayheb, C. Hollanti, and H. Poor. Capacity and security of heterogeneous distributed storage systems. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1247–1251, July 2013.

[30] L. R. Ford and D. R. Fulkerson. Maximal Flow through a Network. *Canadian Journal of Mathematics*, 8:399–404, February 1956.

[31] B. Gastón, J. Pujol, and M. Villanueva. A realistic distributed storage system that minimizes data storage and repair bandwidth. In *Proc. of Data Compression Conference*, pages 491–491, March 2013.

[32] M. Gerami, M. Xiao, and M. Skoglund. Optimal-cost repair in multi-hop distributed storage systems. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1437–1441, July 2011.

[33] S. Ghemawat, H. Gobioff, and S.-T. Leung. The google file system. In *Proc. of Nineteenth ACM symposium on Operating systems principles*, SOSP '03, pages 29–43, October 2003.

[34] M. J. E. Golay. Notes on digital coding. *Proceedings of the IRE*, 37:675–675, June 1949.

[35] N. Golrezaei, A. G. Dimakis, and A. F. Molisch. Wireless device-to-device communications with distributed caching. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2781–2785, July 2012.

[36] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin. On the locality of codeword symbols. *IEEE Transactions on Information Theory*, 58(11):6925–6934, November 2012.

[37] S. Goparaju, S. E. Rouayheb, and R. Calderbank. New codes and inner bounds for exact repair in distributed storage systems. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1036–1040, June 2014.

[38] M. K. Gupta, A. Agrawal, and D. Yadav. On Weak Dress Codes for Cloud Storage. *ArXiv e-prints*, February 2013, [Online] Available: https://arxiv.org/abs/1302.3681.

[39] R. W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160, April 1950.

[40] R. Hill. *A first course in coding theory*. Oxford [Oxfordshire] : Clarendon Press, 1986.

[41] H. D. L. Hollmann. On the minimum storage overhead of distributed storage codes with a given repair locality. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1041–1045, June 2014.

[42] H. Hou and Y. S. Han. Basic codes for distributed storage systems. In *Proc. of 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9, July 2017.

[43] H. Hou, K. W. Shum, M. Chen, and H. Li. Basic regenerating code: Binary addition and shift for exact repair. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1621–1625, July 2013.

[44] H. Hou, K. W. Shum, M. Chen, and H. Li. Basic codes: Low-complexity regenerating codes for distributed storage systems. *IEEE Transactions on Information Theory*, 62(6):3053–3069, June 2016.

[45] H. Hou, K. W. Shum, and H. Li. Construction of exact-basic codes for distributed storage systems at the MSR point. In *Proc. of IEEE International Conference on Big Data (Big Data)*, pages 33–38, October 2013.

[46] Y. Hu, Y. Xu, X. Wang, C. Zhan, and P. Li. Cooperative recovery of distributed storage systems from multiple losses with network coding. *IEEE Journal on Selected Areas in Communications*, 28(2):268–276, February 2010.

[47] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin. Erasure coding in windows azure storage. In *Proc. of USENIX conference on Annual Technical Conference*, pages 2–2, June 2012.

[48] X. Huang, H. Li, T. Zhou, Y. Zhang, H. Guo, H. Hou, H. Zhang, and K. Lei. Minimum storage basic codes: A system perspective. In *Proc. of IEEE International Conference on Big Data (Big Data)*, pages 39–43, October 2013.

[49] M. Itani, S. Sharafeddine, and I. ElKabbani. Practical multiple node failure recovery in distributed storage systems. In *Proc. of IEEE Symposium on Computers and Communication (ISCC)*, pages 901–907, June 2016.

[50] S. Jaggi, Y. Cassuto, and M. Effros. Low complexity encoding for network codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 40–44, July 2006.

[51] G. M. Kamath, N. Silberstein, N. Prakash, A. S. Rawat, V. Lalitha, O. O. Koyluoglu, P. V. Kumar, and S. Vishwanath. Explicit mbr all-symbol locality codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 504–508, July 2013.

[52] A. M. Kermarrec, N. L. Scouarnec, and G. Straub. Repairing multiple failures with coordinated and adaptive regenerating codes. In *Proc. of International Symposium on Networking Coding*, pages 1–6, July 2011.

[53] A. Keshavarz-Haddad and M. A. Khojastepour. Rotate-and-add coding: A novel algebraic network coding scheme. In *Proc. of IEEE Information Theory Workshop (ITW)*, pages 1–5, Aug 2010.

[54] M. A. Khojastepour, A. Keshavarz-Haddad, and A. S. Golsefidy. On capacity achieving property of rotational coding for acyclic deterministic wireless networks. In *Proc. of 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, pages 313–317, May 2010.

[55] A. Kiani and S. Akhlaghi. Selective regenerating codes. *IEEE Communications Letters*, 15(8):854–856, August 2011.

[56] Y.-S. Kim, H. Park, and J.-S. No. Construction of new fractional repetition codes from relative difference sets with $\lambda = 1$. *Entropy*, 19(10):112–128, October 2017.

[57] J. C. Koo and J. T. Gill. Scalable constructions of fractional repetition codes in distributed storage systems. In *Proc. of 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1366–1373, September 2011.

[58] M. N. Krishnan and P. V. Kumar. On MBR codes with replication. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 71–75, July 2016.

[59] J. Kubiatowicz, D. Bindel, Y. Chen, S. E. Czerwinski, P. R. Eaton, D. Geels, R. Gummadi, S. C. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Y. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proc. of 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 190–201, November 2000.

[60] H. Lee and J. Lee. An outer bound on the storage-bandwidth tradeoff of exact-repair cooperative regenerating codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 66–70, July 2016.

[61] H. Lee and J. Lee. An outer bound on the storage-bandwidth tradeoff of exact-repair regenerating codes and its asymptotic optimality in high rates. In *Proc. of IEEE Information Theory Workshop (ITW)*, pages 255–259, September 2016.

[62] H. Lee and J. Lee. An outer bound on the storage-bandwidth tradeoff of exact-repair cooperative regenerating codes. *IEEE Transactions on Information Theory*, 63(11):7267–7282, November 2017.

[63] D. Leong, A. G. Dimakis, and T. Ho. Distributed storage allocations. *IEEE Transactions on Information Theory*, 58(7):4733–4752, July 2012.

[64] J. Li and B. Li. Cooperative repair with minimum-storage regenerating codes for distributed storage. In *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, pages 316–324, April 2014.

[65] J. Li, X. Tang, and U. Parampalli. A framework of constructions of minimal storage regenerating codes with the optimal access/update property. *IEEE Transactions on Information Theory*, 61(4):1920–1932, April 2015.

[66] Z. Li, T. Ho, D. Leong, and H. Yao. Distributed storage allocation for heterogeneous systems. In *Pro. of 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 320–326, October 2013.

[67] Q. Liu, D. Feng, Z. Shi, and M. Fu. General functional regenerating codes with uncoded repair for distributed storage system. In *Proc. of 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 372–381, May 2015.

[68] Microsoft. SkyDrive Live. January 2013.

[69] M. Y. Nam, J. H. Kim, and H. Y. Song. Locally repairable fractional repetition codes. In *Proc. of Seventh International Workshop on Signal Design and its Applications in Communications (IWSDA)*, pages 128–132, September 2015.

[70] M. Y. Nam, J. H. Kim, and H. Y. Song. Locally repairable fractional repetition codes. *The Journal of Korean Institute of Communications and Information Sciences*, 40(9):1741–1753, September 2015.

[71] M.-Y. NAM, J.-H. KIM, and H.-Y. SONG. Some constructions for fractional repetition codes with locality 2. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E100.A(4):936–943, April 2017.

[72] P. S. Neelakanta and P. S. Nelakanta. *A Textbook on ATM Telecommunications: Principles and Implementation*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1999.

[73] V. Ntranos, G. Caire, and A. G. Dimakis. Allocations for heterogenous distributed storage. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2761–2765, July 2012.

[74] O. Olmez and A. Ramamoorthy. Repairable replication-based storage systems using resolvable designs. In *Proc. of 50th Annual Allerton Conference on*

Communication, Control, and Computing (Allerton), pages 1174–1181, October 2012.

[75] O. Olmez and A. Ramamoorthy. Constructions of fractional repetition codes from combinatorial designs. In *Proc. of Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pages 647–651, November 2013.

[76] O. Olmez and A. Ramamoorthy. Replication based storage systems with local repair. In *Proc. of International Symposium on Network Coding (NetCod)*, pages 1–6, June 2013.

[77] O. Olmez and A. Ramamoorthy. Fractional repetition codes with flexible repair from combinatorial designs. *IEEE Transactions on Information Theory*, 62(4):1565–1591, April 2016.

[78] D. S. Papailiopoulos and A. G. Dimakis. Locally repairable codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2771–2775, July 2012.

[79] D. S. Papailiopoulos, A. G. Dimakis, and V. R. Cadambe. Repair optimal erasure codes through hadamard designs. *IEEE Transactions on Information Theory*, 59(5):3021–3037, April 2013.

[80] D. S. Papailiopoulos, J. Luo, A. G. Dimakis, C. Huang, and J. Li. Simple regenerating codes: Network coding for cloud storage. In *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, pages 2801–2805, March 2012.

[81] H. Park and Y.-S. Kim. Construction of fractional repetition codes with variable parameters for distributed storage systems. *Entropy*, 18(12):245–252, December 2016.

[82] S. Pawar, N. Noorshams, S. E. Rouayheb, and K. Ramchandran. Dress codes for the storage cloud: Simple randomized constructions. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2338–2342, July 2011.

[83] J. Pernas, C. Yuen, B. Gastón, and J. Pujol. Non-homogeneous two-rack model for distributed storage systems. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1237–1241, July 2013.

[84] A. Porter, S. Silas, and M. Wootters. Load-Balanced Fractional Repetition Codes. *ArXiv e-prints*, February 2018, [Online] Available: https://arxiv.org/abs/1802.00872.

[85] S. A. Prajapati and M. K. Gupta. On Some Universally Good Fractional Repetition Codes. *ArXiv e-prints*, September 2016, [Online] Available: https://arxiv.org/abs/1609.03106.

[86] N. Prakash, V. Abdrashitov, and M. Médard. The storage versus repair-bandwidth trade-off for clustered storage systems. *IEEE Transactions on Information Theory*, 64(8):5783–5805, August 2018.

[87] N. Prakash and M. N. Krishnan. The storage-repair-bandwidth trade-off of exact repair linear regenerating codes for the case $d = k = n - 1$. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 859–863, June 2015.

[88] 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*, 57(8):5227–5239, August 2011.

[89] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran. Explicit construction of optimal exact regenerating codes for distributed storage. In *Proc. of 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1243–1249, September 2009.

[90] B. Sasidaran and P. V. Kumar. On the interior points of the storage-repair bandwidth tradeoff of regenerating codes. In *Proc. of 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 788–795, October 2013.

[91] B. Sasidharan, K. Senthoor, and P. V. Kumar. An improved outer bound on the storage-repair-bandwidth tradeoff of exact-repair regenerating codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2430–2434, June 2014.

[92] B. Sasidharan, M. Vajha, and P. V. Kumar. An explicit, coupled-layer construction of a high-rate msr code with low sub-packetization level, small field size and $d < (n-1)$. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2048–2052, June 2017.

[93] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur. XORing elephants: Novel erasure codes for big data. *Proceedings of the VLDB Endowment*, pages 325–336, March 2013.

[94] K. Senthoor, B. Sasidharan, and P. V. Kumar. Improved layered regenerating codes characterizing the exact-repair storage-repair bandwidth tradeoff for certain parameter sets. In *Proc. of IEEE Information Theory Workshop (ITW)*, pages 1–5, April 2015.

[95] N. Shah, K. Rashmi, and P. Vijay Kumar. A flexible class of regenerating codes for distributed storage. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1943 –1947, June 2010.

[96] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran. Distributed storage codes with repair-by-transfer and nonachievability of interior points on the storage-bandwidth tradeoff. *IEEE Transactions on Information Theory*, 58(3):1837–1852, March 2012.

[97] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, July 1948.

[98] K. Shum. Fractional repetition codes. *IERG6120 Coding for Distributed Storage Systems*, 01 December 2016.

[99] K. W. Shum. Cooperative regenerating codes for distributed storage systems. In *Proc. of IEEE International Conference on Communications (ICC)*, pages 1–5, June 2011.

[100] K. W. Shum, H. Hou, M. Chen, H. Xu, and H. Li. Regenerating codes over a binary cyclic code. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1046–1050, June 2014.

[101] K. W. Shum and Y. Hu. Cooperative regenerating codes. *IEEE Transactions on Information Theory*, 59(11):7229–7258, November 2013.

[102] N. Silberstein. Fractional repetition and erasure batch codes. In *Proc. of The Fourth International Castle Meeting on Coding Theory and Applications (ICMCTA)*, volume 3 of *CIM Series in Mathematical Sciences*, pages 335–343. Springer, September 2014.

[103] N. Silberstein and T. Etzion. Optimal fractional repetition codes for distributed storage systems. In *Proc. of IEEE 28th Convention of Electrical Electronics Engineers in Israel (IEEEI)*, pages 1–4, December 2014.

[104] N. Silberstein and T. Etzion. Optimal fractional repetition codes and fractional repetition batch codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 2046–2050, June 2015.

[105] N. Silberstein and T. Etzion. Optimal fractional repetition codes based on graphs and designs. *IEEE Transactions on Information Theory*, 61(8):4164–4180, August 2015.

[106] R. Singleton. Maximum distanceq-nary codes. *IEEE Transactions on Information Theory*, 10(2):116–118, April 1964.

[107] Y. Su. Pliable fractional repetition codes for distributed storage systems: Design and analysis. *IEEE Transactions on Communications*, 66(6):2359–2375, June 2018.

[108] Y. S. Su. Constructions of fractional repetition codes with flexible per-node storage and repetition degree. In *Proc. of IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, December 2017.

[109] C. Tian. Rate region of the (4, 3, 3) exact-repair regenerating codes. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1426–1430, July 2013.

[110] C. Tian. Characterizing the rate region of the (4,3,3) exact-repair regenerating codes. *IEEE Journal on Selected Areas in Communications*, 32(5):967–975, May 2014.

[111] C. Tian. On the fundamental limits of coded caching and exact-repair regenerating codes. In *Proc. of International Symposium on Network Coding (NetCod)*, pages 56–60, June 2015.

[112] V. T. Van, C. Yuen, and J. Li. Non-homogeneous distributed storage systems. In *Proc of 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1133–1140, October 2012.

[113] A. Wang and Z. Zhang. Exact cooperative regenerating codes with minimum-repair-bandwidth for distributed storage. In *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, pages 400–404, April 2013.

[114] X. Wang, Y. Xu, Y. Hu, and K. Ou. MFR: Multi-loss flexible recovery in distributed storage systems. In *Proc. of IEEE International Conference on Communications*, pages 1–5, May 2010.

[115] H. Weatherspoon and J. D. Kubiatowicz. Erasure coding vs. replication: A quantitative comparison. In *Proc. of the International Workshop on Peer-to-Peer Systems*, pages 328–337, 2002.

[116] H. Weatherspoon and J. D. Kubiatowicz. Erasure coding vs. replication: A quantitative comparison. In P. Druschel, F. Kaashoek, and A. Rowstron, editors, *Peer-to-Peer Systems*, pages 328–337. Springer Berlin Heidelberg, March 2002.

[117] Y. Wu. Existence and construction of capacity-achieving network codes for distributed storage. In *Proc. of IEEE International Symposium on Information Theory (ISIT)*, pages 1150 –1154, July 2009.

[118] Y. Wu. Existence and construction of capacity-achieving network codes for distributed storage. *IEEE Journal on Selected Areas in Communications*, 28(2):277–288, February 2010.

[119] Y. Wu, R. Dimakis, and K. Ramchandran. Deterministic regenerating codes for distributed storage. In *Proc. of The Allerton Conference on Communication, Control and Computing (Urbana-Champaign)*, pages 242–249, September 2007.

[120] G. Xu, Q. Mao, S. Lin, K. Shi, and H. Zhang. Extremal graphic model in optimizing fractional repetition codes for efficient storage repair. In *Proc. of IEEE International Conference on Communications (ICC)*, pages 1–6, April 2016.

[121] Q. Yu, K. W. Shum, and C. W. Sung. Minimization of storage cost in distributed storage systems with repair consideration. In *Proc. of IEEE Global Telecommunications Conference - GLOBECOM 2011*, pages 1–5, December 2011.

[122] Q. Yu, K. W. Shum, and C. W. Sung. Tradeoff between storage cost and repair cost in heterogeneous distributed storage systems. *Transactions on Emerging Telecommunications Technologies*, 26(10):1201–1211, October 2015.

[123] Q. Yu, C. W. Sung, and T. H. Chan. Irregular fractional repetition code optimization for heterogeneous cloud storage. *IEEE Journal on Selected Areas in Communications*, 32(5):1048–1060, April 2014.

[124] H. Zhang, M. Chen, A. Parekh, and K. Ramchandran. A distributed multichannel demand-adaptive p2p vod system with optimized caching and neighbor-selection. In *Proc. SPIE Optical Engineering + Applications*, volume 81350, pages 1–19, September 2011.

[125] H. Zhang, H. Li, K. W. Shum, H. Hou, and S. R. Li. Concurrent regenerating codes. *IET Communications*, 11(3):362–369, February 2017.

[126] M. F. Zhang and S. T. Xia. Cyclic repetition erasure code. In *Proc. of 3rd International Conference on Consumer Electronics, Communications and Networks*, pages 213–216, November 2013.

[127] B. Zhu. Rethinking fractional repetition codes: New construction and code distance. *IEEE Communications Letters*, 20(2):220–223, February 2016.

[128] B. Zhu. A study on universally good fractional repetition codes. *IEEE Communications Letters*, 22(5):890–893, May 2018.

[129] B. Zhu and H. Li. Adaptive fractional repetition codes for dynamic storage systems. *IEEE Communications Letters*, 19(12):2078–2081, December 2015.

[130] B. Zhu and H. Li. Exploring node repair locality in fractional repetition codes. *IEEE Communications Letters*, 20(12):2350–2353, December 2016.

[131] B. Zhu, H. Li, H. Hou, and K. W. Shum. Replication-based distributed storage systems with variable repetition degrees. In *Proc. of Twentieth National Conference on Communications (NCC)*, pages 1–5, February 2014.

[132] B. Zhu, H. Li, and S. Y. R. Li. General fractional repetition codes from combinatorial designs. *IEEE Access*, 5(11):26251–26256, November 2017.

[133] B. Zhu, H. Li, K. W. Shum, and S.-Y. R. Li. HFR code: a flexible replication scheme for cloud storage systems. *IET Communications*, pages 2095–2100, October 2015.

[134] B. Zhu, K. W. Shum, and H. Li. Heterogeneity-aware codes with uncoded repair for distributed storage systems. *IEEE Communications Letters*, 19(6):901–904, June 2015.

[135] B. Zhu, K. W. Shum, and H. Li. On the duality of fractional repetition codes. In *Proc. of IEEE Information Theory Workshop (ITW)*, pages 56–60, November 2017.

[136] B. Zhu, K. W. Shum, and H. Li. On the Duality and File Size Hierarchy of Fractional Repetition Codes. *ArXiv e-prints*, August 2018, [Online] Available: https://arxiv.org/abs/1808.01933.

[137] B. Zhu, K. W. Shum, H. Li, and H. Hou. General fractional repetition codes for distributed storage systems. *IEEE Communications Letters*, 18(4):660–663, April 2014.

[138] B. Zhu, K. W. Shum, H. Li, and S. Y. R. Li. On low repair complexity storage codes via group divisible designs. In *Proc. of IEEE Symposium on Computers and Communications (ISCC)*, pages 1–5, June 2014.

[139] M. Zorgui and Z. Wang. On the Achievability Region of Regenerating Codes for Multiple Erasures. *ArXiv e-prints*, Jan. 2018, [Online] Available: https://arxiv.org/abs/1802.00104.