Partially collaborative storage codes for distributed storage systems

Liu, Shiqiu

2015

Liu, S. (2015). Partially collaborative storage codes for distributed storage systems.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/65824>

<https://doi.org/10.32657/10356/65824>

PARTIALLY COLLABORATIVE
STORAGE CODES FOR DISTRIBUTED
STORAGE SYSTEMS

SHIQIU LIU

SCHOOL OF PHYSICAL & MATHEMATICAL SCIENCES

2015

PARTIALLY COLLABORATIVE STORAGE CODES FOR DISTRIBUTED STORAGE SYSTEMS

SHIQIU LIU

School of Physical & Mathematical Sciences

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2015

Declaration

This thesis is a presentation of my original research work, carried out by myself, in collaboration with my supervisor Frédérique Oggier. Most of the material in this thesis has appeared in conference proceedings and been presented in various conferences.

Publications in Conference Proceedings:

1. Shiqiu Liu and Frédérique Oggier, “On Storage Codes Allowing Partially Collaborative Repairs”, Information Theory(ISIT), 2014 IEEE International Symposium on, pp.2440-2444.
2. Shiqiu Liu and Frédérique Oggier, “On the Design of Storage Orbit Codes”, Coding Theory and Applications, 4th International Castle Meeting on Coding Theory and Applications, Palmela Castle, Portugal, September 15-18, 2014. CIM Series in Mathematical Sciences, vol.3. Springer International Publishing pp. 263-271.
3. Shiqiu Liu and Frédérique Oggier, “Two Storage Code Constructions Allowing Partially Collaborative Repairs”, Information Theory and Its Applications (ISITA), 2014 International Symposium on, pp.378-382.

Publications in Journals:

1. **(Submitted to Advances in Mathematics of Communications)** Shiqiu Liu and Frédérique Oggier, “On Applications of Orbit Codes to Storage” (Status: conditional accepted).
2. **(Accepted by International Journal of Information and Coding Theory)** Shiqiu Liu and Frédérique Oggier, “Partially Collaborative Storage Codes in the Presence of an Eavesdropper”.

Acknowledgments

The four-year Ph.D study has been so far the most exciting, challenging, and interesting part of my life. Before the end of this journey, I would like to express my thanks to all the people who helped and supported me.

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Frédérique Oggier for the continuous support of my Ph.D study and research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. Many results of this thesis could not be realized without her constructive suggestions and insightful advice. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank all the examiners for examining and providing valuable suggestions and comments for my thesis.

Last but not least, I wish to thank my family and friends for their support and companionship.

Abstract

This thesis contributes to the study of designing erasure codes for distributed storage systems. Erasure codes have traditionally been studied in the context of communication theory, but have also increasingly been used to ensure the reliability of distributed storage systems, since they provide a good trade-off between storage overhead (or rate) and reliability in case of storage node failures. There is however a critical difference between erasure codes for communication and for storage application: in the context of storage, reliability needs to be ensured over time. Failures of storage nodes keep happening, and the lost data must be rebuilt, to guarantee resiliency over time. This process is often called maintenance, or repair.

Over the past years, there has been a vibrant research activity around the problem of designing good storage erasure codes, amenable to repair. A network coding inspired line of works was introduced by Dimakis et al. [13], who used a max-flow min-cut analysis to characterize the communication cost needed for repairing one failed node as a function of the code storage overhead. This setting has been generalized independently in [4, 5] to allow the simultaneous repair of several nodes failures thanks to node collaboration during the repair process, resulting in a better trade-off in terms of repair communication cost versus storage

overhead than in the case of one failure repair at a time.

The main contribution of this thesis is to introduce a family of storage codes that generalizes the framework of collaborative repairs, which subsumes the previously known constructions with either no collaboration (one failure at a time) or full collaboration (several failures at a time, involving all repair nodes in the collaboration as proposed in [4, 5]). The motivation is twofold: to allow more degrees of freedom in the code design to best suit possible storage system requirements, and to trade gain in repair bandwidth with security impairments that typically appear in the presence of collaboration. More precisely, this thesis contains a trade-off between storage overhead and repair bandwidth as a function of the collaboration degree of the repair nodes involved, and code constructions: codes for the special cases where either the storage overhead or the repair bandwidth is minimized, and codes derived from group actions on Grassmannian spaces, known as orbit codes. Finally, the degree of collaboration is studied as a function of the confidentiality one would like to obtain in the presence of eavesdroppers.

Contents

Declaration	i
Acknowledgments	iii
Abstract	v
Contents	vii
1 Introduction	1
1.1 Redundancy and Erasure Codes	1
1.2 The Notion of Repair	3
1.3 Regenerating Codes	6
1.4 Collaborative Regenerating Codes	10
1.5 Introducing Partial Collaboration	13
2 Preliminaries	17
2.1 Coding Theory	17
2.2 Information Theory	21
3 Constructions of (Fully) Collaborative Regenerating Codes	25

3.1	Construction I: t failures at MSR point	25
3.2	Construction II: t failures at MBR point	29
4	Partially Collaborative Repair	31
4.1	A Min-Cut Bound	31
4.2	Minimum Storage and Bandwidth Points	40
5	Code Constructions for Partial Collaboration	45
5.1	A Generic Code Construction	45
5.2	Codes with Minimum Storage	49
5.3	Codes with Minimum Repair Bandwidth	54
5.3.1	Data Placement and Object Recovery	54
5.3.2	Repair of $t = m$ Failures	57
5.3.3	Repair of $t = 1, 2$ Failures	60
6	Orbit Codes	65
6.1	Storage Codes from Orbit Codes	66
6.1.1	Code Length	68
6.1.2	Storage Capacity	69
6.1.3	Fault tolerance	70
6.1.4	Repair	72
6.2	Instances of Cyclic Orbit Codes using Companion Matrices	74
6.2.1	Basic Properties	74
6.2.2	Code Construction I: t Failures with Minimized Bandwidth	80
6.2.3	Code Construction II: three Failures and more	86
6.2.4	Code Construction III: Spreads	90

6.3	Instances of Cyclic Orbit Codes using Diagonal Matrices	92
6.3.1	Reed-Solomon Codes.	93
6.3.2	A Code Construction for t Failures, $t \geq 1$	94
6.3.3	Comparison	95
7	Secure File Size against a Passive Eavesdropper	99
7.1	The Case of Minimum Repair Bandwidth	100
7.1.1	A Min-Cut Bound	101
7.1.2	A Bandwidth Analysis	109
7.2	The Case of Minimum Storage	112
7.2.1	A Min-Cut Bound	113
7.2.2	Discussion	120
8	Conclusion	123
8.1	Results	124
8.2	Future Works	124
	Bibliography	127
	List of Figures	131

Chapter 1

Introduction

This thesis is devoted to the study of code design for networked distributed storage systems (DSS).

Networked distributed storage systems refer to systems that store data in a network of storage nodes, for which the interconnection among nodes plays an important role. The goal of any storage system is to store data reliably, that is, to ensure that the stored data is not lost.

1.1 Redundancy and Erasure Codes

Suppose a single copy of a data object is stored, either in a single node, or this copy is split into smaller pieces that are distributed over several nodes. In the case of a node failure, (part of) the data is lost, see Figure 1.1. It is therefore needed that the information contained in each data object is stored several times, a technique known as *redundancy*. One way to obtain redundancy is replication. Typically, data is split into pieces and each piece is stored 3 times.

Figure 1.1: A file is stored in one storage node, when this node experiences a failure, the file is lost.

It is clear that in terms of reliability the more redundancy, the better. If more replicas of an object are stored, then the system can tolerate more failures. However, the growing number of replicas increases the storage space of the system. In the context of replication, suppose we store n replicas of one object, each stored in a distinct node, the ratio of useful data per total data is $1/n$, and we say that $n/1$ is the **storage overhead** (or $1/n$ is the rate). In terms of erasure codes, n replicas means that we are using an $(n, 1)$ repetition code, a linear code of dimension 1 and length n . In order to maintain high reliability but reduce the overhead, one could replace the $(n, 1)$ repetition code by an (n, k) erasure code. An (n, k) erasure code is a k -dimensional code of length n . To use it for storage, split the data object into k pieces and encode these k pieces into n pieces such that the data object can be recovered from a subset of the n pieces. The storage overhead induced by the use of an (n, k) code is given by n/k . There has been a

lot of research done in the past few years to replace repetition codes in distributed storage systems by more efficient erasure codes. Different codes have been used by companies like Google, Windows Azure [1], Facebook [2], Cleversafe or Wuala, to name a few.

The Maximum Distance Separable (MDS) codes (see Chapter 2) are known to be optimal in terms of trade-off between rate and fault tolerance because they achieve the Singleton Bound. Consider a system with N nodes, an object of size M will be stored in $n < N$ nodes. Firstly, divide the object into k pieces, each of size M/k , then using an (n, k) -MDS code encode the object into n pieces, and each piece is stored in a distinct node. Because the storage code satisfies the ***MDS property***, any k encoded pieces out of n are enough to reconstruct the original data object.

1.2 The Notion of Repair

A storage system is meant to provide reliability over months or years, which is the main difference from a communication setting. In a storage system, failures continue to happen, thus maintenance, that is, the process of rebuilding and storing the lost data is of importance.

Consider the 3-way repetition method (or $(n, 3)$ repetition code), when one of the copies failed, one can recreate a new copy by contacting any of the other two nodes possessing a copy, thus the system can tolerate up to 2 failures of each piece, see Figure 1.2. However, when two failures occur, if a new copy is not obtained, the system can no longer tolerate any failure. Thus these two failures need to be successfully repaired before another occurs.

Figure 1.2: A file is divided into two pieces (A, B), each of the pieces is stored three times. The figure shows that when a node storing A fails, it can be recovered by contacting one of the other two nodes storing A , when two nodes storing B fail, they can be recovered by contacting the one storing B which is still alive.

Thus, besides achieving small storage overhead, efficient erasure codes for storage should also be amenable to *repair*, a notion which is a specificity of storage systems. Consider a codeword from some erasure code, and store one (or several) coefficients of this codeword across distinct network nodes. If some nodes fail, the original data will be protected by the erasure code, up to the code ability. However, if over time more nodes start failing, the data will be lost, and thus, before more failures occur, the system needs to trigger a repair process, that will make sure the data redundancy will be replenished, to keep the data protected over time. This is done by having nodes performing the repair, which concretely consists of contacting live nodes, downloading data from them, and computing the missing codeword coefficients, see Figure 1.3.

Maximum Distance Separable (MDS) codes are optimal with respect to storage

Figure 1.3: A file is encoded and stored in three nodes, each node stores A , B and $A + B$ respectively. When the node storing B fails, a new node comes, contacts the other two live nodes, and downloads A and $A + B$ respectively. After computation, the new node stores B which is the missing coefficient.

overhead versus fault tolerance. Let us see what happens from the repair point of view. For an (n, k) MDS code, in order to repair one single piece, one needs to contact k nodes to obtain a whole copy of the data object, that is at least k nodes should be alive and able to respond to the repair process, which here means decoding. This is a waste of communication and computation cost for one failure, or a small number of failures. The drawbacks of MDS codes motivated the study of codes that could provide an efficient repair process.

An efficient repair process may mean several things: one may desire it to be fast, to be computationally cheap, to reduce the number of symbols communicated, or to contact a small number of live nodes only. The storage codes aiming at reducing the *repair bandwidth*, that is, the amount of symbols communicated during repair are called *regenerating codes*, while the storage codes aiming at

reducing the *repair degree (locality)*, that is, the number of live nodes connected per repair are called *locally repairable codes* [27,30–33]. This thesis will focus on the design of regenerating codes.

1.3 Regenerating Codes

In [13], Dimakis et al. introduced a min-cut max-flow technique from network coding to the setting of distributed storage to optimize the repair bandwidth. The result of their analysis is a trade-off bound between storage and repair bandwidth, and codes achieving this bound are called “regenerating codes”. Dimakis et al. considered the scenario of only one single failure repair at a time. The main idea of regenerating codes is that they could satisfy an MDS-like property (the object can be retrieved by contacting any choice of k nodes, where k is a fixed threshold, not necessarily the code dimension), and in the repair process, many nodes could be contacted (typically all but the failed one) and less symbols downloaded, in turn reducing the cost of repairing one single failure. The data in repaired nodes could be of two forms: one is that the repaired nodes contain what they have lost exactly (*exact repair*), and the other form is that the repaired nodes do not contain exactly the lost data, but maintain overall information about the data in the storage system (*functional repair*), see Example 1.3.1. This thesis focusses on the exact repair for distributed storage systems, which is useful in many practical applications and is more difficult (it cannot be guaranteed by random coding arguments).

Example 1.3.1. A data object is encoded into three coefficients A , B and $A + B$ over a finite field $\mathbb{F}_q$, $q \neq 2$. The coefficients are stored in three nodes respectively.

Figure 1.4: The upper case corresponding to exact repair, the lower case corresponding to functional repair.

When the node containing $A + B$ fails, exact repair means that the repaired node contains $A + B$, while an example of functional repair is that the repaired node contains $\alpha A + \beta B$, $\alpha, \beta \in \mathbb{F}_q \setminus \{0\}$, see Figure 1.4.

As mentioned above, it was shown in [13] that there is an optimal tradeoff between the *storage capacity* (the amount of data stored in one node such that the object can be retrieved by contacting any choice of k nodes) and repair bandwidth for regenerating codes. There are two very important extremal points in the tradeoff:

- The *Minimum Storage Regenerating (MSR) point* is the point where the storage capacity is minimum. We say that a code is at MSR if it is such that the storage capacity is minimum. If it achieves MSR point, it is an optimal MSR code. From the property that the object can be retrieved by contacting any choice of k nodes, the storage capacity should be at least $1/k$

of the object size. Thus at MSR point, the storage capacity should be equal to $1/k$ of the object size, the repair bandwidth also achieves the minimum. The storage versus repair bandwidth trade-off tells us that the minimum storage point corresponds to an optimal repair bandwidth.

- The *Minimum Repair Bandwidth Regenerating (MBR) Point* is the point where the repair bandwidth achieves its minimum. Since the repair bandwidth needs to be no less than the storage capacity, we have at MBR point, that the repair bandwidth equals the storage capacity and they both are minimized. A code is at MBR if it is such that the storage capacity equals the repair bandwidth. If a code achieves MBR point, it is an optimal MBR code.

Let us fix some notation and terminology to be more precise. Consider a network of N nodes, and a data object $\mathbf{o}$ is stored across n of these N nodes. This object is of length M symbols. We use the simplifying assumption that only one object is stored¹. Each node is assumed to have the same storage capacity of α symbols over some alphabet, typically some finite field. Nodes that participate in the repair process are sometimes called newcomers. The newcomers could be nodes that are newly added to the system, or may be nodes that already existed, but did not store any data of the object considered. A newcomer can connect to d nodes which store encoded parts of the object, sometimes called live nodes, and download some data from each of these d nodes. The newcomer downloads β amount of data (called download repair bandwidth) from each of these d nodes, and thus downloads at most $d\beta$ amount of data in total. We denote by γ the

¹The notation of a network of N nodes, $N > n$, is here to emphasize that an object is stored over a subset of nodes

Figure 1.5: A data object $\mathbf{o}$ of size M is stored across n of N storage nodes, each node of these n nodes stores α amount of data. When a node fails, a newcomer contacts d live nodes, downloads β amount of data from each of these d nodes, after computation with the downloaded data, it finally stores α amount of data. Any k out of n nodes can retrieve the object, it is sometimes said that a *data collector* may contact any k nodes to retrieve the data.

repair bandwidth, that is the amount of data required to repair one node, see Figure 1.5.

In [13], Dimakis et al. computed the value of the storage capacity and of the repair bandwidth at MSR and MBR points for repairing one single failure:

MSR point:

$$\begin{aligned}\alpha &= \frac{M}{k}, \\ \gamma &= \frac{M}{k} \frac{d}{d-k+1}, \\ \beta &= \frac{M}{k} \frac{1}{d-k+1}.\end{aligned}$$

MBR point:

$$\alpha = \gamma = \frac{M}{k} \frac{2d}{2d - k + 1},$$

$$\beta = \frac{M}{k} \frac{2}{2d - k + 1}.$$

1.4 Collaborative Regenerating Codes

In [13], no matter how many failures occurred, the repairs are always done sequentially. Even if several failures happened, each failure will be repaired separately, one after the other, and most of the benefits are obtained when only one failure occurred among n nodes, and $n - 1$ nodes are used for repair (that is, $d = n - 1$). To reduce the cost of repair further, and to repair more than one failure simultaneously, two independent works [4, 5] introduced *collaborative (coordinated, cooperative) regenerating codes*, which allow repairs using the data not only from live nodes, but also from all nodes currently being repaired. These collaborative regenerating codes have been showed to reduce the repair cost further for more than one failure. The storage capacity and repair bandwidth still have a trade-off for collaborative regenerating codes with two extremal points, the MSR and MBR points.

In practice, a storage system may have a threshold t of number of failures, $t \geq 1$, after which the maintenance is triggered. In the event of t failures, the repair is done collaboratively among t nodes, as follows. Every node participating in the repair process downloads β data from d nodes, each of the repairing nodes exchanges β' (*exchange repair bandwidth*) data with every other $t - 1$ repair

N	N is the total number of nodes in the network
n	number of nodes storing one object
M	the size of an object
k	any choice of k nodes allow the object retrieval
α	storage capacity per node
d	number of live nodes be contacted by one newcomer
β	download repair bandwidth
γ	repair bandwidth per node
β'	exchange repair bandwidth

Table 1.1: Parameters involved in regenerating codes.

nodes, see Figure 1.6. The parameters at MSR point and MBR points for collaborative regenerating codes are:

MSR point:

$$\begin{aligned}\alpha &= \frac{M}{k}, \\ \gamma &= \frac{M}{k} \frac{d+t-1}{d-k+t}, \\ \beta &= \beta' = \frac{M}{k} \frac{1}{d-k+t}.\end{aligned}$$

MBR point:

$$\begin{aligned}\alpha &= \gamma = \frac{M}{k} \frac{2d+t-1}{2d-k+t}, \\ \beta &= \frac{M}{k} \frac{2}{2d-k+t}, \\ \beta' &= \frac{M}{k} \frac{1}{2d-k+t}.\end{aligned}$$

We summarize the parameters that have been introduced, in Table 1.1, which we will commonly use throughout the thesis.

Figure 1.6: $t = 4$ failures repaired simultaneously, and each of these four nodes exchange β' with every other nodes.

Although collaborative regenerating codes are useful to reduce the repair cost for more than one failure at the same time, their constructions are not easy. Some different code constructions have been proposed for different regimes (e.g. [7], [8]), and we will survey some constructions of collaborative regenerating codes in the following chapter.

The *security* of storage systems using collaborative codes has also been studied, and it can be seen as the resilience of the systems to active or passive attacks [21, 24]. In the case of a passive adversary that eavesdrops, the behaviour of these codes have been studied [11], where bounds on the size of a secure object have been computed. In the case of active adversaries, that is rogue nodes voluntarily corrupting the data that they transmit during repair [12], it was shown that honest nodes are critical to the repair process, since the repair bandwidth obtained to secure (from an information theoretical point of view) collaborative

regenerating codes from Byzantine attacks is worse than having no collaboration at all. This is a rational drawback of collaboration, since a single corrupted node can pollute all other honest nodes involved in the repair.

1.5 Introducing Partial Collaboration

There are two lessons to be learnt from collaborative codes:

- more failures mean more nodes collaborating, and less repair bandwidth per failure,
- collaboration endangers security when one of the participants is corrupted.

We believe it could be valuable for a system designer to have more flexibility in the number of nodes involved at the two different phases of repair, both in terms of efficacy and security.

Motivated by these considerations, we introduce in this thesis the notion of *partial collaboration* to capture the property that the nodes being repaired do collaborate by exchanging some data, but nodes do not necessarily communicate with all of the other repairing nodes. The notion of partial collaboration provides a range of different regimes between regenerating codes and collaborative regenerating codes, corresponding to *no collaboration* and *full collaboration* respectively, see Figure 1.7. From a security point of view, collaboration has been shown to be harmful, thus reducing the amount of collaboration could be a method to mitigate this threat. It is natural to call these newly introduced storage codes as *partially collaborative regenerating codes*.

Figure 1.7: A data object $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_4, \dots)$ is encoded using a generator matrix G with columns $\mathbf{g}_i$. Node i stores $\mathbf{o}_i \mathbf{g}_i, \mathbf{o}_{i+1} \mathbf{g}_i, \mathbf{o}_{i+2} \mathbf{g}_i, \mathbf{o}_{i+3} \mathbf{g}_i$, $i = 1, 2, 3, 4$, and the first subscript is understood modulo 4. This figure shows the repair of $t = 4$ failures (nodes 1,2,3,4): on the left of first line, with no collaboration, on the right of bottom line, with full collaboration. The two other figures show two intermediate modes of collaboration. The more the nodes collaborate, the less they download from live nodes, and vice-versa.

In this thesis, we mainly investigate the tradeoff of partial collaboration and the design of partially collaborative codes for exact repair. The thesis is constructed as follows. In Chapter 3, we present two constructions of collaborative regenerating codes which are prior art to show that the design is not easy. In Chapter 4, we analyze the tradeoff between storage capacity and repair bandwidth for partial collaboration by computing a min-cut bound, and give the parameter values at MSR and MBR points. In Chapter 5, we give constructions of partially collaborative regenerating codes at MSR and MBR points. In Chapter 6, we consider the applications of orbit codes to distributed storage, which can be used for partial collaboration. In Chapter 7, we evaluate to which extent partial collaboration mitigates security threats. All new results are joint work with Frédérique Oggier.

Chapter 2

Preliminaries

In this chapter, we state some basic concepts and techniques from coding theory which are very important for the design of storage codes (e.g. [22]), and some concepts from information theory useful in discussing the security of distributed storage systems (e.g. [23]). Let us begin with coding theory.

2.1 Coding Theory

Let $\mathbb{F}_q$ be the finite field with q elements, q a prime power.

Definition 2.1.1. A *linear code* of length n and rank k is a linear subspace C with dimension k of the vector space $\mathbb{F}_q^n$, where $\mathbb{F}_q$ is the finite field with q elements. The vectors in C are called *codewords*. The size of a code is the number of codewords and equals q^k .

Definition 2.1.2. A *generator matrix* G for a linear code C is a matrix whose rows form a basis for C .

Definition 2.1.3. The *dual code* $C^\perp$ of C is the orthogonal complement of the subspace C of $\mathbb{F}_q^n$, that is

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

If C is of dimension k , then $C^\perp$ has dimension $n - k$.

Definition 2.1.4. A *parity-check matrix* for a linear code C is a matrix H whose rows form a basis for the dual code $C^\perp$, that is H satisfies

$$H\mathbf{x}^T = 0, \forall \mathbf{x} \in C.$$

Definition 2.1.5. Given two vectors $\mathbf{x}$ and $\mathbf{y}$, the *Hamming distance* between $\mathbf{x}$ and $\mathbf{y}$ is the number of coefficients in which $\mathbf{x}$ and $\mathbf{y}$ differ, denoted by $d(\mathbf{x}, \mathbf{y})$.

Definition 2.1.6. Let $\mathbf{x}$ be a vector in $\mathbb{F}_q^n$, the *(Hamming) weight* of $\mathbf{x}$ is defined to be the number of non-zero coefficients of $\mathbf{x}$, denoted by $wt(\mathbf{x})$.

Definition 2.1.7. The *minimum (Hamming) distance* d_H of a linear code C is the minimum Hamming distance between any two distinct codewords, namely

$$d_H(C) = \min_{\mathbf{x} \neq \mathbf{y} \in C} d(\mathbf{x}, \mathbf{y}) = \min_{\mathbf{x} \neq \mathbf{y} \in C} wt(\mathbf{x} - \mathbf{y}).$$

Consider an (n, k) linear code over $\mathbb{F}_q$. A codeword in C , in the context of communication, is sent over an erasure channel, which will result in an erroneous codeword, where some of its components have been erased. The code should be designed so as to permit the recovery of the codeword, assuming a given erasure pattern. The minimum Hamming distance d_H of the code characterizes the erasure

recovery capacity of the code, then it can tolerate $d_H - 1$ erasures. The reason is that any two codewords differ in at least d positions, thus as long as erased positions are not more than $d - 1$, it is possible to distinguish them.

Erasure codes can be easily used for distributed storage systems: instead of sending a codeword over an erasure channel, store the n coefficients of a codeword of C into n storage nodes. Then node failures become erasures, and whether the stored encoded object can be retrieved given a failure pattern depends on d_H .

Theorem 2.1.8. (*The Singleton Bound*) *Let C be an (n, k, d_H) linear code. Then*

$$n - k \geq d_H - 1,$$

that is

$$d_H \leq n - k + 1.$$

Corollary 2.1.9. *An (n, k) linear code reaching the Singleton bound can recover up to $n - k$ erasures.*

Definition 2.1.10. A code achieving the Singleton bound is called ***maximum distance separable (MDS)*** code.

MDS codes are very useful because they achieve the maximum distance and can tolerate most erasures. Thus, when using an (n, k) MDS code in distributed storage systems, one node can be repaired by contacting any k other nodes.

Definition 2.1.11. A ***Reed-Solomon (RS)*** code C is defined as the set C of codewords defined by

$$C = \{(f(\alpha_1), \dots, f(\alpha_n)) | f \text{ is a polynomial over } \mathbb{F}_q \text{ of degree } < k\},$$

where $\alpha_1, \dots, \alpha_n$ are n distinct elements of $\mathbb{F}_q$.

More precisely, let $\alpha_1, \dots, \alpha_n$ be n distinct elements of $\mathbb{F}_q$, and consider the polynomial

$$f(X) = u_1 + u_2X + \dots + u_kX^{k-1},$$

where $u_1, \dots, u_k$ are information symbols, and $k < n$. Codewords are obtained by evaluating f at α_i for $i = 1, \dots, n$, namely

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

Obviously n is at most the cardinality of $\mathbb{F}_q$.

Theorem 2.1.12. *A Reed-Solomon code is an MDS code.*

Proof. Since any two different polynomials of degree less than k agree in at most $k-1$ points, this means that any two codewords of the Reed-Solomon code disagree in at least $n - (k-1) = n - k + 1$ positions. Furthermore, there are two polynomials that do agree in $k-1$ points but are not equal, and thus, the distance of the Reed-Solomon code is exactly $d = n - k + 1$. $\square$

The following definition and theorem are about the repair degree (locality) and its relation with the minimum distance of a linear code.

Definition 2.1.13. An (n, k) code has **repair degree (locality)** r if every codeword component in a codeword is a linear combination of at most r other components in the codeword.

Thus, when using a code of repair degree r , one node only needs to contact at most r other nodes to be repaired.

It has been shown [30,31] that there is a tradeoff between locality and minimum distance.

Theorem 2.1.14. *Let C be an (n, k) linear code with minimum distance d and repair degree (locality) r , then*

$$n - k + 1 - d \geq \lfloor \frac{k-1}{r} \rfloor.$$

The above bound explains why MDS codes are not that good with respect to the repair degree.

2.2 Information Theory

In this section, we will see some basic concepts and techniques from information theory.

Let X be a discrete random variable with alphabet $\mathcal{X}$, and $p(x) = \Pr\{X = x\}$, $x \in \mathcal{X}$ is the probability mass function. Firstly, we see a concept that describes the expected value (average) of the information contained in each message received.

Definition 2.2.1. The *entropy* $H(X)$ of a discrete random variable X is defined by

$$H(X) = - \sum_{x \in \mathcal{X}} p(x) \log p(x).$$

Since $0 \leq p(x) \leq 1$, the entropy is non-negative.

Definition 2.2.2. The *joint entropy* $H(X, Y)$ of a pair of discrete random

variables (X, Y) with a joint distribution $p(x, y)$ is defined as

$$H(X, Y) = - \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log p(x, y).$$

Definition 2.2.3. If $(X, Y) \sim p(x, y)$, then the *conditional entropy* $H(Y|X)$ is defined as

$$\begin{aligned} H(Y|X) &= \sum_{x \in \mathcal{X}} p(x) H(Y|X = x) \\ &= - \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log p(y|x) \end{aligned} \tag{2.2.1}$$

Theorem 2.2.4. (*Chain rule for entropy*)

Let $X_1, \dots, X_n$ be n discrete random variables, drawn according to $p(x_1, \dots, x_n)$.

Then

$$H(X_1, X_2, \dots, X_n) = \sum_{i=1}^n H(X_i | X_{i-1}, \dots, X_1).$$

Definition 2.2.5. Consider two random variables X and Y with a joint probability mass function $p(x, y)$ and marginal probability mass functions $p(x)$ and $p(y)$. The *mutual information* $I(X; Y)$ is defined by

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

Mutual information is a measure of the amount of information that one random variable contains with respect to another. It has a close relationship with entropy.

Theorem 2.2.6.

$$I(X; Y) = H(X) - H(X|Y),$$

$$I(X; Y) = H(Y) - H(Y|X),$$

$$I(X; Y) = H(X) + H(Y) - H(X, Y),$$

$$I(X; X) = H(X).$$

Recall the discussion about entropy, we also need to consider the conditional mutual information.

Definition 2.2.7. The *conditional mutual information* of random variables X and Y given Z is defined by

$$I(X; Y|Z) = H(X|Z) - H(X|Y, Z).$$

There is also a chain rule for mutual information.

Theorem 2.2.8.

$$I(X_1, \dots, X_n; Y) = \sum_{i=1}^n I(X_i; Y|X_{i-1}, \dots, X_1).$$

Chapter 3

Constructions of (Fully) Collaborative Regenerating Codes

The design of collaborative regenerating codes is not easy, we use the following two constructions [6], [7] to give one example at MBR point and one at MSR point respectively. For more designs and constructions, one may refer e.g. to [3], [8], [9], [10].

3.1 Construction I: t failures at MSR point

As shown in [6], consider a data object $\mathbf{o}$ of length M with coefficients in $\mathbb{F}_q$, q a prime power. Set the parameters of the collaborative regenerating code to be

$$\begin{aligned}
d &= k, & t &= r, \\
M &= kr, & \alpha &= r, \\
n &\geq d + r, \\
\gamma &= d + r - 1.
\end{aligned}$$

The parameters at MSR point should satisfy

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

The above parameters match the optimal parameters at MSR point.

Let G be a generator matrix of an (n, k) MDS code (for example a Reed-Solomon code, see Chapter 2) over $\mathbb{F}_q$, denote by $\mathbf{g}_i$ the i th column of G , $i = 1, \dots, n$, that is

$$G = [\mathbf{g}_1, \dots, \mathbf{g}_n].$$

By the MDS property, any k columns of G form an invertible matrix. We divide the object $\mathbf{o}$ into r k -dimensional row vectors, $\mathbf{o}_1, \dots, \mathbf{o}_r$, which can be represented as a matrix $\mathbf{O}$ in $\mathbb{F}_q^{r \times k}$, that is

$$\mathbf{O} = \begin{bmatrix} o_{1,1} & \dots & o_{1,k} \\ \vdots & & \vdots \\ o_{r,1} & \dots & o_{r,k} \end{bmatrix} = \begin{bmatrix} \mathbf{o}_1 \\ \vdots \\ \mathbf{o}_r \end{bmatrix}.$$

Let the i th node store $\mathbf{O}\mathbf{g}_i = \{\mathbf{o}_j \mathbf{g}_i, j = 1, \dots, r\}$, thus each of the nodes stores r

coefficients, that is $\alpha = r$.

Object recovery. The object can be recovered by contacting nodes $i_1, \dots, i_k$. The i_l -th node contains $\mathbf{O}\mathbf{g}_{i_l}$, for $l = 1, \dots, k$, for a total of kr coefficients. For each $\mathbf{o}_i$, $i = 1, \dots, r$, by the MDS property, the data object can be retrieved.

Repair of $t = r$ failures. Suppose $t = r$ nodes have failed. Without loss of generality, we label the newcomers from 1 to r .

Download phase. For $i = 1, \dots, r$, the i th newcomer connects to $d = k$ live nodes say $i_1, \dots, i_d$ and downloads $\mathbf{o}_i\mathbf{g}_{i_l}$ for $l = 1, \dots, d$. Each newcomer downloads $\beta = 1$ coefficient from each of the d nodes. By the MDS property, the i th newcomer can compute $\mathbf{o}_i$.

Collaboration phase. The newcomer i computes and sends $\mathbf{o}_i\mathbf{g}_j$ to newcomer j , for $i \neq j$. Thus each of the newcomers exchanges $r - 1$ coefficients with others.

The total number of coefficients involved in the repair for these $t = r$ failures is $kr + r(r - 1) = d + r - 1$, that is the repair bandwidth is $d + r - 1$, and the construction indeed achieves MSR point.

Example 3.1.1. Given an object $\mathbf{o}$ of size $M = 15$ with coefficients in $\mathbb{F}_q$, q a prime power. Divide the object $\mathbf{o}$ into three 5-dimensional row vectors, $\mathbf{o}_1, \dots, \mathbf{o}_3$, $\mathbf{o}$ can be represented as a matrix $\mathbf{O}$ in $\mathbb{F}_q^{3 \times 5}$, that is

$$\mathbf{O} = \begin{bmatrix} o_{1,1} & \dots & o_{1,5} \\ o_{2,1} & \dots & o_{2,5} \\ o_{3,1} & \dots & o_{3,5} \end{bmatrix} = \begin{bmatrix} \mathbf{o}_1 \\ \mathbf{o}_2 \\ \mathbf{o}_3 \end{bmatrix}.$$

Let $n = 9$, G be a generator matrix of a $(9, 5)$ -MDS code, with $G = [\mathbf{g}_1, \dots, \mathbf{g}_9]$. The object is stored in $n = 9$ nodes, and each node stores $\mathbf{O}\mathbf{g}_i = \{\mathbf{o}_j\mathbf{g}_i, j = 1, 2, 3\}$ for

$i = 1, \dots, 9$. The property of G ensures that the object can be recovered whenever $k = 5$ nodes are contacted. Then we get $\alpha = t = r = 3$, $k = d = 5$.

Suppose that node $1, \dots, 3$ fail, we label the newcomers node $1, \dots, 3$. Each of the newcomers connects to $d = k = 5$ nodes among the remaining nodes $4, \dots, 9$. Suppose these three nodes all download coefficients from node $4, \dots, 8$, then the i th node downloads $\mathbf{o}_i \mathbf{g}_4, \dots, \mathbf{o}_i \mathbf{g}_8$, for $i = 1, 2, 3$. Using the MDS property of G , the node i can compute $\mathbf{o}_i$ and $\mathbf{o}_i \mathbf{g}_i$, the missing 2 coefficients are obtained by collaboration with the other two newcomers. Namely the node i contacts nodes $i + 1, i + 2 \pmod{3}$ as follows:

- node 1 gets $\mathbf{o}_2 \mathbf{g}_1$ from node 2, gets $\mathbf{o}_3 \mathbf{g}_1$ from node 3,
- node 2 gets $\mathbf{o}_3 \mathbf{g}_2$ from node 3, gets $\mathbf{o}_1 \mathbf{g}_2$ from node 1,
- node 3 gets $\mathbf{o}_1 \mathbf{g}_3$ from node 1, gets $\mathbf{o}_2 \mathbf{g}_3$ from node 2.

Each newcomer downloads 1 coefficient from one live nodes, and gets 2 other coefficients from the other two newcomers, in total 3 coefficients used to repair one failure. We have $\gamma = \alpha = 3$.

Note that the strategy is not unique, the newcomers do not necessarily download from the same 5 remaining nodes.

3.2 Construction II: t failures at MBR point

As stated in [7], consider a data object $\mathbf{o}$ of length M with coefficients in $\mathbb{F}_q$, q a prime power. Set the parameters of the collaborative regenerating code to be

$$\begin{aligned} d &= k, \\ t &= r, \\ M &= k(k + r), \\ n &= d + r, \\ \alpha &= \gamma = 2d + r - 1. \end{aligned}$$

The parameters at MBR point should satisfy

$$(\alpha_{MBR}, \gamma_{MBR}) = \left(\frac{M}{k} \frac{2d + t - 1}{2d - k + t}, \frac{M}{k} \frac{2d + t - 1}{2d - k + t} \right)$$

The above parameters match the optimal parameters at MBR point.

The object $\mathbf{o}$ can be written as a vector $(o_0, \dots, o_{kn-1})$, each of the coefficients belongs to $\mathbb{F}_q$, since the length is $M = kn$. Divide the object vector into n groups, the j -th group contains $o_{(j-1)k}, \dots, o_{jk-1}$, denote the row vector $\mathbf{o}_j = (o_{(j-1)k}, \dots, o_{jk-1})$, then the j -th group contains $\mathbf{o}_j$ of length k , $1 \leq j \leq n$.

Let G be a generating matrix of an $(n - 1, k)$ MDS codes over $\mathbb{F}_q$, $\mathbf{g}_j$ ($j = 1, \dots, n - 1$) are the column vectors of G .

For $i = 1, 2, \dots, n$, the content of node i is constructed as follows: put the k

coefficients of the i th group $\mathbf{o}_i$ into node i and the $n - 1$ parity-check coefficients

$$\mathbf{o}_{i \oplus 1} \mathbf{g}_1, \dots, \mathbf{o}_{i \oplus (n-1)} \mathbf{g}_{n-1}$$

into node i , where $\oplus$ is the modulo- n addition defined by

$$x \oplus y := \begin{cases} x + y, & \text{if } x + y \leq n, \\ x + y - n, & \text{if } x + y > n. \end{cases}$$

Object recovery. Suppose without loss of generality that a data collector connects to nodes $1, \dots, k$. The coefficients $o_0, \dots, o_{k^2-1}$ of the vectors $\mathbf{o}_1, \dots, \mathbf{o}_k$ can be downloaded directly. The coefficients of the j -th vector ($j > k$) can be computed from $\mathbf{o}_j \cdot \mathbf{g}_{j-1}, \dots, \mathbf{o}_j \cdot \mathbf{g}_{j-k}$ by the MDS property.

Repair of $t = r$ failures. The system waits for t failures to happen, suppose without loss of generality that nodes $k + 1, \dots, n$ have failed.

Download phase. The newcomer j , $j = k + 1, \dots, n$, downloads $\mathbf{o}_i \cdot \mathbf{g}_{n+i-j}$ and $\mathbf{o}_j \cdot \mathbf{g}_{n+j-i}$ from node i , $i = 1, \dots, k$. A total of $2k(n - k) = 2kr$ coefficients have been transmitted in this phase.

Collaboration Phase. The newcomer j ($j = k+1, \dots, n$) can compute $\mathbf{o}_j$ because of the MDS property. Then the node j sends $\mathbf{o}_j \cdot \mathbf{g}_{n+j-i}$ to node i , for $i = k + 1, \dots, n, i \neq j$. Each newcomer has exchanged $r - 1$ coefficients with others.

Thus, the total number of coefficients involved in the repair for these $t = r$ failures is $2kr + r(r - 1) = r(2d + r - 1)$, that is the repair bandwidth is $2d + r - 1$, and the construction indeed achieves MBR point.

Chapter 4

Partially Collaborative Repair

As seen from Chapter 3, regenerating codes have been studied in the extreme cases of no collaboration and full collaboration. It is natural to think of giving more flexibility to their constructions, and one way to do so is by reducing the number of nodes contacted during the collaborative process. We refer to this relaxation as partial collaboration. Before constructing regenerating codes for partially collaborative repairs, we analyze the tradeoff between the storage α and the repair bandwidth γ . The tradeoff is classically obtained from a min-cut bound.

4.1 A Min-Cut Bound

A key concept in network coding theory is *information flow*: the network is seen as a directed graph, where data flows from the sources to the sinks via vertices and edges. Similarly in distributed storage systems, a data object will be stored into n nodes from a source S , then flows to some other storage nodes during the repair process, and at last flows to a data collector DC (or sink) which contacts

k storage nodes to access the object. When some nodes failed, the system can trigger a repair process by downloading data from the live nodes and exchanging some data within the repairing ones. The data collector then contacts k live nodes, including by assumption the repaired ones, to retrieve the object.

To capture the notion of storage capacities, a node x is represented as a logical pair (x_{in}, x_{out}) [13], thus the storage capacity can be represented by the edge of weight α between x_{in} and x_{out} . To repair $t \geq 2$ failures simultaneously, t newcomers each contacts a possibly different set of d live nodes, and downloads β amount of data from each of the d nodes. The newcomers can then collaborate by exchanging the downloaded data among each other. This collaborative scenario can be modeled by representing each of the repairing node as a triple $(x_{in}, x_{coord}, x_{out})$ [4], where the edge between x_{in} and x_{coord} carries the amount of data that the node temporarily stores (which is at least the storage α), the edge between x_{coord} and x_{out} has the capacity α . The download repair bandwidth β can be represented by the edge between x_{out} and x_{in} of a repairing node, while the edge between x_{in} (of a repairing node) and x_{coord} carries the exchange repair bandwidth β' . see Figure 4.1.

In [13], the authors gave a min-cut bound for a single node repair which is the no collaboration scenario:

$$\min cut(S, DC) \geq \sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\}$$

In [4, 6], a bound for t nodes repaired simultaneously, corresponding to the full

Figure 4.1: A source S with an object $\mathbf{o}$ encoded and distributed into n networked nodes. When some nodes failed, there is a fully collaborative repair. The data collector (DC) contacts k nodes to get the object.

collaboration scenario is computed:

$$\min cut(S, DC) \geq \sum_{i=0}^{g-1} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1})\beta + (t - u_i)\beta'\}$$

When t failures are repaired simultaneously, for the no collaboration scenario, each of the repairing nodes can be considered as repaired separately, since it has to download at least α data to reconstruct what it stored, the total repair cost equals to t times the repair cost for one node. However, when allowing collaboration in the repair process, the repairing nodes will communicate among themselves, so that each of them could download less than α data, then get the missing data from other repairing nodes. Collaboration has been proved [4, 6] to reduce the total repair cost.

Figure 4.2: A data object $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_4)$ is encoded using a generator matrix G with columns $\mathbf{g}_i$. Node i stores $\mathbf{o}_i \mathbf{g}_i, \mathbf{o}_{i+1} \mathbf{g}_i, \mathbf{o}_{i+2} \mathbf{g}_i, \mathbf{o}_{i+3} \mathbf{g}_i$, $i = 1, 2, 3, 4$, and the first subscript is understood modulo 4. This figure shows the repair of $t = 4$ failures (nodes 1,2,3,4): on the left of first line, with no collaboration $s = t = 4$, on the right of bottom line, with full collaboration $s = 1$. The two other figures show two intermediate modes of collaboration $s = 2, 3$. The more the nodes collaborate, the less they download from live nodes, and vice-versa.

Recall that when t failures are repaired simultaneously, for the no collaboration scenario, the repairing nodes do not communicate with each other, while in full collaboration mode, each of the repairing nodes communicate with all other repairing nodes. We consider no collaboration case as having a cooperation level 0, while full collaboration has a collaboration of level $t - 1$. Then partial collaboration will have a cooperation level between 0 and $t - 1$. Using $t - s$, $1 \leq s \leq t$ represents the cooperation level, that is, $s = 1$ corresponds to full collaboration, and $s = t$ to no collaboration. See Figure 4.2.

To compute the min-cut bound for partial collaboration, as done in [4, 6, 13], we consider an information flow graph, where the data flows from a source S to a data collector (DC). The repair process of t failures involves t nodes, which will all download β amount of data from d live nodes, and exchange β' amount of data with a subset of $t - s$ nodes, $1 \leq s \leq t$. The collaboration phase is assumed to be done in such a way that every node has both incoming and outgoing degrees $t - s$ ($s = 1$ corresponds to full collaboration, and $s = t$ to none).

Theorem 4.1.1. *Consider an information flow graph, where every node has a storage capacity of α , and repairs are performed by a group of t nodes, as described above. Suppose that a data collector DC connects to a subset of k nodes which were all involved in different phases of repairs, where each phase involves a group of u_i nodes, $1 \leq u_i \leq t$, and $k = \sum_{i=0}^{g-1} u_i$. Then a min-cut bound between the source S and the data collector DC is given by*

$$\begin{aligned} \min cut(S, DC) \geq \min_{\mathbf{u} \in P} & \left(\sum_{i \in \bar{I}} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta\} + \right. \\ & \left. \sum_{i \in I} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta'\} \right) \end{aligned} \quad (4.1.1)$$

where

$$I = \{i, t - s + 1 - u_i \geq 0\}, \quad \bar{I} = \{i, t - s + 1 - u_i < 0\}$$

and

$$P = \{\mathbf{u} = (u_0, \dots, u_{g-1}), 1 \leq u_i \leq t \text{ and } \sum_{i=0}^{g-1} u_i = k\}.$$

We will adopt the following notation. Repair nodes during the i th phase of repair are denoted by $x^{i,j}$, where j counts the nodes during the i th phase. Every

Figure 4.3: Partial collaboration for $t = 4$ failures, and exchange data with $t - s = 2$ nodes, with three different cuts: the download links, the collaboration links and the storage capacity links are cut.

node $x^{i,j}$ is seen as a logical triple $(x_{in}^{i,j}, x_{coor}^{i,j}, x_{out}^{i,j})$ formed by an incoming node, a collaborating node, and an output node see Figure 4.3, to model the storage capacity and the collaborative process, as done in [4]. The main lines of the proof are similar to that of [4], but some details are nevertheless given for the sake of completeness.

Proof. Consider a data collector DC that connects to k output nodes corresponding to a set K of nodes, say $\{x_{out}^{i,j} : (i,j) \in K\}$. We show that any cut between S and DC in the graph has a capacity that satisfies (4.1.1). Since we may assume that all the edges of the data collector have infinite capacity, we only consider the cut $(U, \bar{U})$, with $S \in U$ and $\{x_{out}^{i,j} : (i,j) \in K\} \in \bar{U}$. Let $\mathcal{C}$ denote the edges in the cut.

First Repair Phase. Let J be the set of indices such that $\{x_{out}^{0,j} : j \in J\}$ are the first output nodes in $\overline{U}$ corresponding to the first repair. The set contains $|\{x_{out}^{0,j} : j \in J\}| = u_0$ nodes. Consider a subset $M \subset J$ of size m such that $\{x_{in}^{0,j} : j \in M\} \subset U$ and $\{x_{in}^{0,j} : j \in J - M\} \subset \overline{U}$. Then m can take value between 0 and u_0 .

Consider firstly the m nodes $\{x_{in}^{0,j} : j \in M\}$. For each node $x_{in}^{0,j} \in U$, (1) either $x_{coord}^{0,j} \in U$, then $x_{coord}^{0,j} \rightarrow x_{out}^{0,j} \in \mathcal{C}$, and the contribution to the cut is α , (2) or $x_{coord}^{0,j} \in \overline{U}$, then $x_{in}^{0,j} \rightarrow x_{coord}^{0,j} \in \mathcal{C}$, and contribution to the cut is at least α .

Consider next the $u_0 - m$ other nodes $\{x_{in}^{0,j} : j \in J - M\}$. For each node the contribution comes from multiple sources.

- (1) The cut contains d edges carrying β coefficients: since $x_{out}^{0,j}$ are the first output nodes in $\overline{U}$, edges come from the output nodes in U .
- (2) When $t - s + 1 \geq u_0 - m$, the cut contains at least $t - s + 1 - u_0 + m$ edges carrying β' coefficients thanks to the coordination step: the node $x_{coord}^{0,j}$ has $t - s$ incoming edges $x_{in}^{0,k} \rightarrow x_{coord}^{0,j}$. However, since $|\{x_{in}^{0,k}\}| \cap \overline{U} = u_0 - m - 1$, the cut contains at least $(t - s) - (u_0 - m - 1)$ such edges.
- (3) When $t - s + 1 < u_0 - m$, the least number of edges carrying β' will be 0, that is the cut contains no edges carrying β' .

Therefore, the total contribution of those nodes when $t - s + 1 \geq u_0 - m$ is $c_0(m) \geq m\alpha + (u_0 - m)(d\beta + (t - s + 1 - u_0 + m)\beta')$. When $t - s + 1 < u_0 - m$, $c_0(m) \geq m\alpha + (u_0 - m)d\beta$. Since the function c_0 is concave on the interval $[0, u_0]$, Jensen's inequality yields

$$c_0(m) \geq u_0 \min\{\alpha, d\beta + (t - s + 1 - u_0)\beta'\},$$

or

$$c_0(m) \geq u_0 \min\{\alpha, d\beta\}.$$

Second Repair Phase. Let $\{x_{out}^{1,j} : j \in J\}$ be the second output nodes in $\overline{U}$ corresponding to a second repair. We repeat the same reasoning. We firstly consider the m nodes $\{x_{out}^{1,j} : j \in M\} \subset U$, where the contribution of each node is α .

Then, we consider the $u_1 - m$ nodes $\{x_{out}^{1,j} : j \in U - M\}$. For each node, we have:

(1) The cut contains at least $d - u_0$ edges carrying β : since these $\{x_{out}^{1,j}\}$ are the second output nodes in $\overline{U}$, at most u_0 edges come from output nodes in U , so at least $d - u_0$ edges come from output nodes in $\overline{U}$.

(2) Similarly to the first phase, the cut contains at least $t - s + 1 - u_1 + m$ edges carrying β' when $t - s + 1 \geq u_1 - m$; the cut contains no edges carrying β' , when $t - s + 1 < u_1 - m$.

Thus the total contribution of these nodes when $t - s + 1 \geq u_1 - m$ is $c_1(m) \geq u_1 \min\{\alpha, (d - u_0)\beta + (t - s + 1 - u_1)\beta'\}$, and $c_1(m) \geq u_1 \min\{\alpha, (d - u_0)\beta\}$ if $t - s + 1 < u_1 - m$.

In general, we have for the i th repair phase that

$$c_i(m) \geq u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta'\},$$

when $t - s + 1 \geq u_i - m$, and when $t - s + 1 < u_i - m$,

$$c_i(m) \geq u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta\}.$$

Summing these contributions leads to (4.1.1). $\square$

Using the Minimum cut-Maximum flow Theorem, we get that the initial file size M must satisfy

$$M \leq \min_{\mathbf{u} \in P} \left(\sum_{i \in I} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta'\} \right. \\ \left. + \sum_{i \in \bar{I}} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta\} \right) \quad (4.1.2)$$

where $I, \bar{I}, P$ are the parameters defined in Theorem 4.1.1.

Corollary 4.1.2. *When $s = 1$, the collaboration phase involves all the other $t - s = t - 1$ nodes, and*

$$M \leq \sum_{i=0}^{g-1} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - u_i)\beta'\},$$

for $1 \leq u_i \leq t$ such that $\sum_{i=0}^{g-1} u_i = k$ which is the known bound from [4, 5].

Proof. Set $s = 1$ in (4.1.2) to get

$$M \leq \min_{\mathbf{u} \in P} \sum_{i \in I} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - u_i)\beta'\}$$

since $\bar{I} = \{i : t - u_i < 0\}$ is now empty ($u_i \leq t$), while $I = \{i : t - u_i \geq 0\} = \{0, \dots, g-1\}$ contains all the possible indices for i . $\square$

Corollary 4.1.3. *When $s = t$, there is no collaboration, and*

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

for $u_i = 1$ and $g = k$, which is the known bound from [13].

Proof. Set $s = t$ and $u_i = 1$ in (4.1.2), to get

$$M \leq \min_{\mathbf{u} \in P} \sum_{i \in \bar{I}} u_i \min\{\alpha, (d - i)\beta\}$$

since $\bar{I} = \{i : 1 - u_i \geq 0\}$ is now empty ($u_i = 1$), while $I = \{i : 1 - u_i = 0\} = \{0, \dots, k - 1\}$ contains all the possible indices for i . $\square$

4.2 Minimum Storage and Bandwidth Points

The Minimum Storage Repair (MSR) Point At MSR point, we have $\alpha = \frac{M}{k}$. The highest contribution from β comes when there is no contribution in β' , that is when $u_i = t - s + 1$ for all i . Suppose $t - s + 1 | k$, then $g = k/(t - s + 1)$, $\bar{I}$ is empty, and (4.1.2) with equality becomes

$$\sum_{i=0}^{k/(t-s+1)-1} (t - s + 1) \min(\alpha, (d - i(t - s + 1))\beta) = M. \quad (4.2.1)$$

Since $\alpha = M/k$, each term in this sum is at most $(t - s + 1)\alpha$, and equality can be obtained only when all terms are $(t - s + 1)\alpha$, so each term which contains β should be greater or equal to $\alpha = M/k$, that is

$$(d - i(t - s + 1))\beta \geq \frac{M}{k},$$

for $i = 0, \dots, \frac{k}{t-s+1} - 1$. When $i = \frac{k}{t-s+1} - 1$, we get

$$\beta = \frac{M}{k} \frac{1}{d - k + t - s + 1}.$$

Conversely, the highest contribution from β' comes when $u_i = 1$ for all i . Then $g = k$, and i ranges from 0 to $k - 1$. Then (4.1.2) with equality gives

$$\sum_{i=0}^{k-1} \min(\alpha, (d-i)\beta + (t-s)\beta') = M \quad (4.2.2)$$

and using the same argument as above, we now need

$$(d-i)\beta + (t-s)\beta' \geq \frac{M}{k}.$$

Thus

$$\beta' \geq \frac{1}{t-s} \left(\frac{M}{k} - \beta(d-k+1) \right) = \frac{M}{k} \frac{1}{d-k+t-s+1}$$

and the repair bandwidth is

$$\gamma = d\beta + (t-s)\beta' = \frac{M}{k} \frac{d+t-s}{d-k+t-s+1}. \quad (4.2.3)$$

To summarize, we have at MSR point

$$\begin{aligned} \alpha &= \frac{M}{k}, \quad \gamma = \frac{M}{k} \frac{d+t-s}{d-k+t-s+1}, \\ \beta &= \beta' = \frac{M}{k} \frac{1}{d-k+t-s+1}. \end{aligned}$$

As a sanity check, we notice that, when $s = 1$, we indeed get the known formulas [4, 6]

$$\beta = \beta' = \frac{M}{k} \frac{1}{d-k+t}. \quad (4.2.4)$$

when $s = t$ with $t = 1$, we get [13]

$$\beta = \frac{M}{k} \frac{1}{d - k + 1}. \quad (4.2.5)$$

We check that the parameters obtained at MSR point indeed satisfy (4.1.2). Each term in the sum (4.1.2) is at most $u_i \frac{M}{k}$, and the minimum sum value is equal to M when all terms are $u_i \frac{M}{k}$. Thus each term which contains β and β' should be greater or equal to $\frac{M}{k}$. Notice that $t - s + 1 - u_i < 0$ when $i \in \bar{I}$, and replace α, β, β' in (4.1.2) with their values at MSR point to get

$$k(d - k + t - s + 1) \leq \sum_{i=0}^{g-1} u_i \left((d - \sum_{j=0}^i u_j) + (t - s + 1) \right).$$

Since $\sum_{j=0}^i u_j \leq k$ when $i < g$, the inequality holds.

The optimal total repair bandwidth is illustrated in Figure 4.4, compared with the known cases where either full collaboration or no collaboration is done. From 4.4, we can see that for t failures, when s increases from 1 to t , which respect to the cooperation level from full collaboration to no collaboration, the repair bandwidth increases, that is, the more collaboration, the less repair bandwidth.

The Minimum Bandwidth Repair (MBR) Point To optimize γ , we let α grow, and (4.2.1) gives

$$(t - s + 1) \sum_{i=0}^{k/(t-s+1)-1} (d - i(t - s + 1))\beta = M,$$

Figure 4.4: The total repair bandwidth γ as a function of $1 \leq s \leq 4$ for $t = 4$, for $k = 6$ (line below), and $k = 8$ (line above). The points at $s = 1$ correspond to the full collaboration case, while those at $s = 4$ correspond to no collaboration.

that is

$$\beta = \frac{M}{k} \cdot \frac{2}{2d - k + t - s + 1}.$$

We next compute β' . From (4.2.2), we get

$$\sum_{i=0}^{k-1} ((d - i)\beta + (t - s)\beta') = M,$$

thus

$$\beta' = \frac{M}{k} \frac{1}{2d - k + t - s + 1}.$$

Finally, at MBR point, the total repair bandwidth $\gamma = d\beta + (t - s)\beta'$ is equal to the storage capacity α , and

$$\gamma = \frac{M}{k} \frac{2d + t - s}{2d - k + t - s + 1}.$$

To summarize, at MBR point, we have

$$\begin{aligned}\alpha &= \gamma = \frac{M}{k} \frac{2d+t-s}{2d-k+t-s+1}, \\ \beta &= \frac{M}{k} \frac{2}{2d-k+t-s+1}, \quad \beta' = \frac{M}{k} \frac{1}{2d-k+t-s+1}.\end{aligned}$$

Another sanity check shows that, when $s = 1$, we indeed get the known formulas [4, 6],

$$\beta = \frac{M}{k} \frac{2}{2d-k+t}, \quad \beta' = \frac{M}{k} \frac{1}{2d-k+t}.$$

when $s = t$ with $t = 1$, we get [13]

$$\beta = \frac{M}{k} \frac{2}{2d-k+1}.$$

That the parameters at MBR point are consistent with (4.1.2) is checked similarly as was done with the parameters at MSR point, by noticing that α is always larger or equal to the bandwidth, and using $t - s + 1 - u_i < 0$ when $i \in \bar{I}$.

Chapter 5

Code Constructions for Partial Collaboration

In this chapter, we firstly propose a generic code construction for partial collaboration which is neither MSR nor MBR. However, if we change the parameters slightly, this generic code construction can achieve MSR. We give the construction which achieves MSR, after which, we also propose a code construction which can achieve the MBR point for some parameters.

5.1 A Generic Code Construction

This code construction is a slight generalization of that of [5] for full collaborative repair, which has been presented in Chapter 3.

Consider an object $\mathbf{o}$ of length kt with coefficients in the finite field $\mathbb{F}_q$, and

represent it as a matrix $\mathbf{O}$ in $\mathbb{F}_q^{t \times k}$, that is

$$\mathbf{O} = \begin{bmatrix} o_{1,1} & \dots & o_{1,k} \\ \vdots & & \vdots \\ o_{t,1} & \dots & o_{t,k} \end{bmatrix} = \begin{bmatrix} \mathbf{o}_1 \\ \vdots \\ \mathbf{o}_t \end{bmatrix}$$

where $\mathbf{o}_i$ is the i th row of $\mathbf{O}$, $i = 1, \dots, t$. Let $G \in \mathbb{F}_q^{k \times n}$ be the generator matrix of an MDS code, whose columns are denoted by $\mathbf{g}_i$, $i = 1, \dots, n$

$$G = [\mathbf{g}_1, \dots, \mathbf{g}_n].$$

The i th node stores $\mathbf{O}\mathbf{g}_i$.

Object Retrieval. If any k nodes are contacted, then the MDS property of G ensures that the object is recovered.

Repair. The system waits for t failures to have happened, before starting the repair process. Without loss of generality, we label these t nodes from 1 to t . Then t newcomers each contact $d = k$ nodes, say the i th node among those t newcomer nodes connect to nodes $i_1, \dots, i_k$ and downloads

$$\mathbf{o}_i \mathbf{g}_{i_j}, \mathbf{o}_{i+1} \mathbf{g}_{i_j}, \dots, \mathbf{o}_{i+s-1} \mathbf{g}_{i_j} \in \mathbb{F}_q, \quad j = 1, \dots, k,$$

where the indices are understood modulo t (ranging from 1 to t). Again using the MDS property of G , the i th node computes $\mathbf{o}_j$, $j = i, \dots, i + s - 1$, and after the download phase, it can compute $\mathbf{o}_j \mathbf{g}_i$, $j = i, \dots, i + s - 1$ and thus has obtained s of the t coefficients it needs to store. The $t - s$ missing coefficients are obtained through (partial) collaboration as follows: the i th node contacts nodes $i - 1, \dots, i -$

$(t - s)$. It already owns $\mathbf{o}_i \mathbf{g}_i, \dots, \mathbf{o}_{i+s-1} \mathbf{g}_i$, and gets $\mathbf{o}_{i-1} \mathbf{g}_i, \dots, \mathbf{o}_{i-(t-s)} \mathbf{g}_i$ from the other nodes.

Note that the code is such that it allows repairs in parallel, no matter the level of collaboration. This coding strategy is illustrated for $t = 4$ and $1 \leq s \leq 4$ in Figure 1.7.

Lemma 5.1.1. *The above construction provides a storage code with partial collaboration, with a storage capacity of $\alpha = t$, download bandwidth $\beta = s$, exchange bandwidth $\beta' = 1$ and total repair bandwidth*

$$\gamma = sk + (t - s), \quad 1 \leq s \leq t,$$

for a repair degree of $d = k$, and collaborative degree of $t - s$.

Proof. The size of the object is tk , thus $\alpha = tk/k = t$ corresponds to the minimum storage value that α can have. During repair, every node contacts $d = k$ live nodes, and downloads $\beta = s$ coefficients from them, accounting for ks , after which $t - s$ coefficients are exchanged, for a total of $sk + (t - s)$. $\square$

Corollary 5.1.2. *If $s = 1$, the exchange process involves $t - s = t - 1$ repair nodes, this is the collaborative case with*

$$\gamma = k + (t - 1), \quad \beta = \beta' = 1.$$

If $s = t$, then there is no collaboration, and the repair process performs t repairs of one failure:

$$\gamma = tk, \quad \beta = t.$$

Proof. Indeed, when $s = 1$, set $M = kt$, $d = k$ in (4.2.3) for γ and in (4.2.4) for $\beta = \beta'$, we get

$$\gamma = k + (t - 1), \quad \beta = \beta' = 1.$$

Again, when $s = t$, set $M = kt$, $d = k$ in (4.2.3) for γ and in (4.2.5), we get

$$\gamma = tk, \quad \beta = t.$$

□

Recall that $d = k$, $\alpha = t$ and $M = tk$. Let us compare the code parameters with those found above at MSR point, since $\alpha = t = M/k$.

$$\begin{aligned} \gamma &= \frac{M}{k} \frac{d + t - s}{d - k + t - s + 1} = \frac{t(k + t - s)}{t - s + 1}, \\ \beta &= \beta' = \frac{M}{k} \frac{1}{d - k + t - s + 1} = \frac{t}{t - s + 1}. \end{aligned}$$

Assume for now that $s \neq 1$, and $k \geq 2$. We notice that:

- $\beta = \frac{t}{t-s+1} \leq s \iff t \geq s$, with equality when $t = s$,
- $\beta' = \frac{t}{t-s+1} \geq 1 \iff s \geq 1$, with equality when $s = 1$,
- $\gamma = \frac{t(k+t-s)}{t-s+1} \leq ks + t - s \iff t \geq s$, with equality when $t = s$.

The case $s = 1$ is discussed in Corollary 5.1.2. This shows all together that the generic construction that we proposed achieves the optimal total repair bandwidth γ if and only if $s = 1$ or $s = t$, that is the full collaboration case, or the case with no collaboration. For all other values of s , the download bandwidth β and thus the overall γ is higher than the optimal one. The gap between the proposed generic

Figure 5.1: The total repair bandwidth γ as a function of $1 \leq s \leq 4$ for $k = 6$ and $t = 4$. The straight line corresponds to the generic coding strategy, while the other line is the optimal strategy.

strategy and the optimal one given by the min-cut bound analysis is shown in Figure 5.1.

5.2 Codes with Minimum Storage

Consider an object $\mathbf{o}$ of size $M = k(t - s + 1)$, $t - s \geq 1$, with coefficients in the finite field $\mathbb{F}_q$, and represent it as a matrix $\mathbf{O}$ in $\mathbb{F}_q^{(t-s+1) \times k}$, that is

$$\mathbf{O} = \begin{bmatrix} o_{1,1} & \cdots & o_{1,k} \\ \vdots & & \vdots \\ o_{t-s+1,1} & \cdots & o_{t-s+1,k} \end{bmatrix} = \begin{bmatrix} \mathbf{o}_1 \\ \vdots \\ \mathbf{o}_{t-s+1} \end{bmatrix}$$

where $\mathbf{o}_i$ is the i th row of $\mathbf{O}$, $i = 1, \dots, t - s + 1$. Let $G \in \mathbb{F}_q^{k \times n}$ be the generator matrix of an MDS code of dimension k and length n , whose columns are denoted by $\mathbf{g}_i$, $i = 1, \dots, n$

$$G = [\mathbf{g}_1, \dots, \mathbf{g}_n].$$

Data Placement. The i th node stores $\mathbf{O}\mathbf{g}_i \in \mathbb{F}_q^{(t-s+1)}$, thus $\alpha = t - s + 1$.

Object Retrieval. If any k nodes $i_1, \dots, i_k$ are contacted, then $\mathbf{O}[\mathbf{g}_{i_1}, \dots, \mathbf{g}_{i_k}]$ is obtained and the MDS property of G ensures that the object $\mathbf{O}$ is recovered.

Repair of t failures. The system waits for t failures to have happened, before starting the repair process. Without loss of generality, we label these t nodes from 1 to t . Then t newcomers each contact $d = k$ nodes, say the i th node among those t newcomer nodes connect to nodes $i_1, \dots, i_k$ and downloads

$$\mathbf{o}_i \mathbf{g}_{i_j} \in \mathbb{F}_q, \quad j = 1, \dots, k,$$

where the indices are understood modulo $t - s + 1$ (ranging from 1 to $t - s + 1$). Again using the MDS property of G , the i th node computes $\mathbf{o}_i$, and after the download phase, it can compute $\mathbf{o}_i \mathbf{g}_i$, and thus obtains one of the $t - s + 1$ coefficients it needs to store. The $t - s$ missing coefficients are obtained through (partial) collaboration as follows: the i th node contacts nodes $i + 1, \dots, i + (t - s)$ (mod t). It already owns $\mathbf{o}_i \mathbf{g}_i$, and gets $\mathbf{o}_{i+1} \mathbf{g}_i, \dots, \mathbf{o}_{i+(t-s)} \mathbf{g}_i$ from the other nodes.

Let us summarize the parameters of the above construction, given an (n, k) MDS code, a threshold of t failures, and collaboration among $t - s$ repair nodes:

$\begin{aligned} M &= k(t - s + 1) \text{ (object size)} \\ \alpha &= t - s + 1 \text{ (storage capacity per node)} \\ d &= k \text{ (number of live nodes contacted)} \end{aligned}$

This construction indeed fits the minimum storage regime, since $\alpha = M/k$. The

total repair bandwidth γ is

$$\gamma = k + (t - s), \quad 1 \leq s \leq t,$$

since every repair node downloads k amount of data, and exchanges $t - s$ of it. One may expect γ to be decreasing in $t - s$ (the collaboration increases with $t - s$), but this comparison cannot be done that way, since both the size of the object M and the storage α are functions of $t - s$.

Example 5.2.1. Given a $(9, 5)$ -MDS code, a threshold of $t = 4$, and collaboration among only $t - s = 4 - 2 = 2$ nodes, we get $M = 15$ and $\alpha = 3$. Let $G = [\mathbf{g}_1, \dots, \mathbf{g}_9]$ be the generator matrix of the MDS code, and $\mathbf{o} \in \mathbb{F}_q^{15}$ be the data object, represented as a matrix $\mathbf{O}$ in $\mathbb{F}_q^{3 \times 5}$, that is

$$\mathbf{O} = \begin{bmatrix} o_{1,1} & \dots & o_{1,5} \\ o_{2,1} & \dots & o_{2,5} \\ o_{3,1} & \dots & o_{3,5} \end{bmatrix} = \begin{bmatrix} \mathbf{o}_1 \\ \mathbf{o}_2 \\ \mathbf{o}_3 \end{bmatrix}$$

Each node stores $\mathbf{O}g_i$ $i = 1, \dots, 9$, and it is clear that the MDS property of G ensures that the object is recovered whenever any $k = 5$ nodes are contacted.

Suppose that node $1, \dots, 4$ fail. Then each of the nodes repairing them connect to the $d = k$ remaining nodes $5, \dots, 9$ and the node repairing node i downloads

$$\mathbf{o}_i \mathbf{g}_j, \quad j = 5, \dots, 9$$

for $i = 1, \dots, 4$. Again using the MDS property of G , the node repairing node i

Figure 5.2: Partially collaborative repair is illustrated based on Example 5.2.1, where $t = 4$ failures are repaired, and repair nodes collaborate by pairs.

can compute

$$\mathbf{o}_i, \mathbf{o}_i \mathbf{g}_i.$$

The 2 missing coefficients are obtained through partial collaboration with $t - s = 2$ other nodes. Namely the node repairing the i th node (we will call it the i th node below by a slight abuse of language) contacts nodes $i + 1, i + 2 \pmod{4}$ as follows:

- node 1 gets $\mathbf{o}_2 \mathbf{g}_1$ from node 2, $\mathbf{o}_3 \mathbf{g}_1$ from node 3,
- node 2 gets $\mathbf{o}_3 \mathbf{g}_2$ from node 3, $\mathbf{o}_1 \mathbf{g}_2$ from node 4,
- node 3 gets $\mathbf{o}_1 \mathbf{g}_3$ from node 4, $\mathbf{o}_2 \mathbf{g}_3$ from node 1,
- node 4 gets $\mathbf{o}_2 \mathbf{g}_4$ from node 1, $\mathbf{o}_3 \mathbf{g}_4$ from node 2.

This is illustrated in Figure 5.2.

It was show in Section 4.2, at MSR point, the parameters of the codes should

satisfy

$$\begin{aligned}\alpha &= \frac{M}{k}, \quad \gamma = \frac{M}{k} \frac{d+t-s}{d-k+t-s+1}, \\ \beta &= \beta' = \frac{M}{k} \frac{1}{d-k+t-s+1},\end{aligned}$$

The code construction that we proposed above has for object size $M = k(t - s + 1)$ and $d = k$, thus should have

$$\alpha = t - s + 1, \quad \gamma = k + t - s, \quad \beta = \beta' = 1,$$

which is indeed the case, it thus corresponds to the MSR point (unlike the similar generic construction of Section 5.1).

This code construction at MSR point proposed for partial collaboration is a generalization of the code construction for full collaboration showed in Section 3.1. In particular, the size M of the object is $M = k(t - s + 1)$, and full collaboration is obtained by setting $s = 1$.

Comparing Example 5.2.1 and Example 3.1.1, we can see that the object $\mathbf{o}$ has the same size 15 with the same value $k = 5$, that is, they are both encoded by a $(9, 5)$ -MDS code. However, Example 5.2.1 with $t = 4, s = 2$ satisfies partial collaboration, and has repair bandwidth $\gamma = 7$, while Example 3.1.1 with $t = 3, s = 1$ satisfies full collaboration, and also has repair bandwidth $\gamma = 7$. The two constructions have different repair threshold t , but have the same repair bandwidth γ . The partially collaborative construction allows the system to wait for more failures to happen without increasing the repair bandwidth.

5.3 Codes with Minimum Repair Bandwidth

Take an object $\mathbf{o} = (o_1, o_2, \dots, o_{2m})$ of size $M = 2m$, for m some positive integer, where every coefficient o_i belongs to some finite field $\mathbb{F}_q$, $i = 1, 2, \dots, 2m$. Encode this object using a single parity check code, that is, $\mathbf{o}$ is encoded into the length $2m + 1$ codeword $\mathbf{x}$ given by $\mathbf{x} = (o_1, o_2, \dots, o_{2m}, o_1 + o_2 + \dots + o_{2m})$.

5.3.1 Data Placement and Object Recovery

Data Placement. Consider a network of $n = 2m + 1$ nodes, and see it as complete graph, with thus $\frac{(2m+1)2m}{2} = m(2m + 1)$ edges. Label the edges of this graph with labels in $\{1, \dots, 2m + 1\}$ so that node i has exactly m edges labelled with i . Equivalently, this corresponds to filling up the adjacency matrix of this graph, and may be done as follows: put zeroes on the diagonal, and then label the i th row, columns $i + 1, \dots, i + m \pmod{2m + 1}$ with i :

$$\begin{bmatrix} 0 & 1 & \dots & 1 & & \\ & 0 & 2 & \dots & 2 & \\ & & \ddots & & & \\ & & & 0 & m+1 & \dots & m+1 \\ m+2 & & & & 0 & m+2 & \dots & m+2 \\ & & & & & & & 0 \\ 2m+1 & \dots & 2m+1 & & & & & \end{bmatrix}$$

Note that because the matrix is symmetric, a label in position ij gives immediately the same label in position ji . This creates m coefficients per row, for $2m + 1$ rows, and since there is no intersection when filling up the adjacency matrix, this indeed

provides a valid labeling of all the edges of the complete graph. Node i stores some of the coefficients of $\mathbf{x}$, whose index corresponds to that of its edges. Since every node has $2m$ edges, but exactly m of them have the same label, this means that every node stores $\alpha = m + 1$ coefficients from $\mathbf{x}$.

Object recovery. It should be possible to retrieve the object $\mathbf{o}$ by contacting any choice of k nodes. Take $k = m$. Every node contains $m + 1$ coefficients from $\mathbf{x}$, thus we get $k(m + 1) = m(m + 1)$ coefficients, out of which we get at least $2m$ distinct coefficients. This is indeed the case: take any row of the adjacency matrix, this gives $m + 1$ coefficients. Now, add $m - 1$ other rows, every new row will have at least one new coefficient which was not contained in the union of the previous rows, thus giving a total of $m + 1 + m - 1 = 2m$ distinct coefficients.

Repair of $t \leq m + 1$ failures. As shown above, as long as m nodes out of $2m + 1$ survive, the data may be recovered. Thus the range of possible failures to consider is for $1 \leq t \leq m + 1$.

The code parameters are summarized below, given an integer $m \geq 1$:

M	$=$	$2m$ (object size)
n	$=$	$2m + 1$ (number of nodes = codeword length)
k	$=$	m (number of nodes to retrieve the object)
α	$=$	$m + 1$ (storage capacity per node)

Since $\alpha = m + 1$, this family of codes works in the minimum repair bandwidth regime as long as the quantity γ downloaded (and exchanged) by every repair node is also $m + 1$.

Example 5.3.1. Take an object $\mathbf{o} = (o_1, o_2, o_3, o_4)$ of size $M = 4 = 2m$, $m = 2$,

where every coefficient o_i belongs to some finite field $\mathbb{F}_q$, $i = 1, 2, 3, 4$. Encode this object using a single parity check code, that is, $\mathbf{o}$ is encoded into the length 5 codeword $\mathbf{x}$ given by $\mathbf{x} = (o_1, o_2, o_3, o_4, o_1 + o_2 + o_3 + o_4)$.

Consider a network of $n = 5$ nodes, and see it as a complete graph, with thus $n(n - 1)/2 = 10$ edges. Label the edges of this graph with labels in $\{1, \dots, 5\}$ such that node i has exactly $m = 2$ edges labeled with i . Using the algorithm described above, this gives the following adjacency matrix:

$$\begin{bmatrix} 0 & 1 & 1 & 4 & 5 \\ 1 & 0 & 2 & 2 & 5 \\ 1 & 2 & 0 & 3 & 3 \\ 4 & 2 & 3 & 0 & 4 \\ 5 & 5 & 3 & 4 & 0 \end{bmatrix}. \quad (5.3.1)$$

Node i stores some of the coefficients of $\mathbf{x}$, whose index corresponds to that of its edges. Since every node has 4 edges, but exactly two of them have the same label, this means that every node stores $\alpha = 3$ coefficients from $\mathbf{x}$.

The above adjacency matrix yields the following storage allocation:

$$\begin{array}{ll} \text{node 1} & x_1, x_4, x_5 \\ \text{node 2} & x_1, x_2, x_5 \\ \text{node 3} & x_1, x_2, x_3 \\ \text{node 4} & x_2, x_3, x_4 \\ \text{node 5} & x_3, x_4, x_5 \end{array}$$

Object recovery. The object $\mathbf{o}$ can be recovered by contacting any choice of $k = 2$ nodes. Indeed, pick any choice of $k = 2$ nodes, to obtain $2\alpha = 6$ coefficients from $\mathbf{x}$, giving at least 4 distinct coefficients, and any choice of 4 distinct x_j is

enough to recover $\mathbf{o}$.

5.3.2 Repair of $t = m$ Failures

We will show next that it is always possible to repair $t = m$ failures, with repair degree $d = 2$, and collaborative repair degree $t - s = 1$. Suppose indeed that t nodes have failed. All node indices below are meant $(\bmod 2m + 1)$, ranging from 1 to $2m + 1$.

Download Phase.

- (i) A node i , such that both its neighbor node $i - 1$ and node i are still up, can clearly be repaired by contacting them, since node i overlaps each of them in m coefficients.
- (ii) If node i fails, and so does one of its neighbors, say node $i + 1$, then the process may contact $d = 2$ nodes per failure. Contacting both nodes $i - 1$ and $i + 2$ ensures to recover m of the $m + 1$ stored coefficients. The missing coefficient surely is available, because every coefficient is present in $m + 1$ nodes, and m failures occurred, so the repair can always be completed.
- (iii) In fact, this works no matter the failure patterns, and in particular the number of contiguous failed nodes: there are always at least two nodes whose contiguous neighbors are live, m coefficients can downloaded from them to repair their neighbors, and the missing one coefficient is always available for the same argument as above: every coefficient is present in $m + 1$ nodes. Iterating this process guarantees that the repair will finish.

Collaboration Phase. Finally, we argue that collaboration with one other repair node is always possible, since no matter what is the failure pattern, any

two failed nodes always intersect in at least one coefficient: this is because two contiguous nodes intersect in m coefficients, when there is one node in between them, two nodes intersect in $m - 1$ coefficients, and in general, when there are i nodes in between, they intersect in $m - i$ coefficients, so the intersection is non-empty whenever $i \leq m - 1$, which is always the case (there are $2m + 1$ nodes with m failures). Now the above download phase showed that in particular the other $m - 1$ coefficients can be retrieved.

The repair parameters for the above repair procedure are summarized below:

t	$=$	m (failure threshold that triggers repairs)
d	$=$	2 (number of live nodes contacted)
$t - s$	$=$	1 (number of collaborators)

We note that the above algorithm does not consider load balancing (trying to make the repair symmetric across repair processes), and we have not computed explicitly the amount of repair bandwidth needed here. The latter will be discussed in next Section.

Example 5.3.2. Consider for example $m = 4$, thus an object size of $2m = M = 8$, stored across $n = 9$ nodes, each storing $\alpha = 5$ coefficients of a codeword $\mathbf{x}$. The algorithm described in the paragraph Data Placement above gives the following

storage allocation:

node 1	x_1, x_6, x_7, x_8, x_9	node 6	x_2, x_3, x_4, x_5, x_6
node 2	x_1, x_2, x_7, x_8, x_9	node 7	x_3, x_4, x_5, x_6, x_7
node 3	x_1, x_2, x_3, x_8, x_9	node 8	x_4, x_5, x_6, x_7, x_8
node 4	x_1, x_2, x_3, x_4, x_9	node 9	x_5, x_6, x_7, x_8, x_9
node 5	x_1, x_2, x_3, x_4, x_5		

Suppose that $t = m = 4$ failures need to be repaired, with repair degree $d = 2$, and cooperative repair degree $t - s = 1$. We illustrate the repair of two failure patterns (we use a repair strategy which is a modification of the above algorithm, to add load balancing, see below), the first one has contiguous node failures. By abuse of notation, we will call node i below the node that is actually repairing node i .

Suppose that nodes 1, 2, 3 and 4 fail. During the downloading phase:

- node 1 gets x_6, x_7 from node 8, and x_8, x_9 from node 9,
- node 2 gets x_1, x_2 from node 5, and x_7, x_8 from node 8,
- node 3 gets x_1, x_3 from node 5, and x_8, x_9 from node 9,
- node 4 gets x_1, x_4 from node 5, and x_2, x_3 from node 6

and during the collaboration phase: node 1 sends x_9 to node 4, node 4 sends x_2 to node 3, node 3 sends x_9 to node 2, and node 2 sends x_1 to node 1.

Note that this strategy is not unique. Furthermore, node 1 could be repaired by getting $x_6, \dots, x_9$ from node 9 and x_1 from node 5 (like described in the above

algorithm), instead, the download is balanced across two nodes, and x_1 is obtained by collaboration.

Suppose a different failure pattern, where nodes 2, 3, 6 and 8 fail. During the downloading phase:

- node 2 gets x_1, x_9 from node 4, x_7, x_8 from node 9,
- node 3 gets x_2, x_3 from node 4, x_8, x_9 from node 9,
- node 6 gets x_4, x_6 from node 7, x_2, x_3 from node 5,
- node 8 gets x_5, x_8 from node 9, x_4, x_7 from node 7

and the repair is finished during the collaboration phase, where node 2 and node 3 exchange x_1 and x_2 , while node 5 and node 8 exchange x_6 and x_5 .

5.3.3 Repair of $t = 1, 2$ Failures

For small values of m , it may be interesting to see how the repairs may be done for values of $t \leq m$ such as one or two failures, even though this does not fall into the setting of partial collaboration, to evaluate the degree of flexibility of the repair process.

Repair of $t = 1$ failure. It is immediate to see that any one failure is repaired by contacting $d = m + 1$ live nodes, from which one coefficient is downloaded. This is nicely illustrated on the adjacency matrix: the failure of node i corresponds to the i th row being lost, then by symmetry, the i th column is equal to the i th row, and this column tells which node should be contacted to retrieve every missing coefficient.

We thus get the following repair parameters:

$$\boxed{t = 1, d = m + 1, t - s = 0}$$

Example 5.3.3. Consider the adjacency matrix (5.3.1) from Example 5.3.1, if node 3 fails, then the 3rd row $(1, 2, 0, 3, 3)$ is lost, but the 3rd column exactly tells us that x_1 is found in node 1, x_2 in node 2, and x_3 either in node 4 or 5.

Repair of $t = 2$ failures. Let us start by compiling some useful remarks.

Lemma 5.3.4. *Suppose $m \geq 2$.*

1. *Suppose only $t = 2$ failures occurred. Two failed contiguous nodes (meaning node i and node $i + 1 \pmod{2m + 1}$) can be repaired by contacting $d = 1$ live node, downloading m coefficients, and exchanging $t - s = 1$ coefficient.*
2. *Any two rows of the adjacency matrix intersect in at least one coefficient, but not necessarily in two.*

Proof. 1) Note that row i and row $i + 1 \pmod{2m + 1}$ have m coefficients in common, thus at least 2 coefficients in common, say x, y . The node repairing node i downloads the m coefficients from node $i - 1 \pmod{2m + 1}$ that intersect, and similarly the node repairing node $i + 1 \pmod{2m + 1}$ downloads the m intersecting coefficients from node $i + 2 \pmod{2m + 1}$. To conclude, it is enough to see that x, y were each downloaded by one of the two repairing nodes.

2) That two rows intersect in at least one coefficient follows from the symmetry of the adjacency matrix. But the 1st row, and row $m + 1$ intersect only in the coefficient 1. □

A consequence of the above lemma is that $t = 2$ failures cannot be repaired by contacting only $d = 1$ live node (but they can with $d = 2$), apart if $m = 2$. Indeed, if node 1 and node $m + 1$ fail, since they intersect in only one coefficient, it is not possible to recover $2m + 1$ coefficients by contacting only two nodes, each intersecting each of the failed nodes in m coefficients. The case $m = 2$ is special, since the parity coefficient of the code can be exploited.

Example 5.3.5. We continue Example 5.3.1 with $m = 2$, with adjacency matrix (5.3.1). We illustrate the case of two contiguous node failures with nodes 1 and 2. We apply the algorithm given in the above lemma. To repair node 1 (containing 1,4,5), download x_4, x_5 from node 5 (nodes 1 and 5 intersect in 4,5), and to repair node 2, download x_1, x_2 from node 3, then exchange x_1 and x_5 . The same algorithm applies for repairing the cases of two contiguous node failures, namely nodes 2,3, nodes 3,4, nodes 4,5 and nodes 5,1.

The situation is different if the two failed nodes are not contiguous, nevertheless the repair can be done with $d = 1$. If nodes 1 and 3 fail, start with the same algorithm as above, namely the node repairing node 1 downloads x_4, x_5 from node 5 which form the intersection of both nodes, and the node that repairs node 3 downloads x_2, x_3 from node 4:

node 1, repaired $x_4 = o_4, x_5 = o_1 + o_2 + o_3 + o_4$

node 3, repaired x_2, x_3

then node 1 gives $x_4 + x_5 = o_1 + o_2 + o_3$ so that node 3 computes $(x_4 + x_5) + (x_2 + x_3) = o_1 = x_1$, then node 3 sends x_1 and the repair is complete. It turns out that any pair of non-contiguous nodes can be repaired similarly, yielding a (fully)

collaborative repair with $d = 1$, thanks to the presence of the parity coefficient.

From Section 5.2, at MBR point, the parameters of the codes should satisfy

$$\begin{aligned} \alpha = \gamma &= \frac{M}{k} \frac{2d+t-s}{2d-k+t-s+1}, \\ \beta &= \frac{M}{k} \frac{2}{2d-k+t-s+1}, \quad \beta' = \frac{M}{k} \frac{1}{2d-k+t-s+1}. \end{aligned}$$

The generic parameters of the codes we proposed are $M = 2m$ and $k = m$, which simplifies the above to

$$\begin{aligned} \alpha = \gamma &= 2 \frac{2d+t-s}{2d-m+t-s+1}, \\ \beta &= \frac{4}{2d-m+t-s+1}, \quad \beta' = \frac{2}{2d-m+t-s+1}. \end{aligned}$$

For $d = 2$ and $t - s = 1$, we thus have that $\alpha = m + 1 \iff 2 \frac{5}{6-m} = m + 1 \iff (m - 1)(m - 4) = 0$. This shows that the code with $m = 2$ in Example 5.3.2 is optimal: indeed, $\beta = 2$, $\beta' = 1$, $\alpha = 5$, $\gamma = 5$.

The code in Example 5.3.1 is optimal as well, but with $d = 1$: $\beta = 2$, $\beta' = 1$, $\alpha = 3$, $\gamma = 3$.

This suggests that the general construction proposed may be optimal for the right value of d , a question that is left open.

Chapter 6

Orbit Codes

In this chapter, we adopt the approach of orbit codes [17], and study applications of such codes to distributed storage, and in particular to the constructions of partially collaborative regenerating codes. We show that some known storage codes can be interpreted as orbit codes as well.

As explained in Chapter 1, in distributed storage systems, maintenance is necessary. In Chapter 2, it was further detailed that, in case of node failures, it is necessary to recompute the lost data, and store it in new live nodes. This can be done using a decoding algorithm, but often, it is preferred to have more efficient ways to compute codeword coefficients, typically as a linear combination of a small number of other codeword coefficients. To better handle the maintenance process, it is often useful to consider each node storing a subspace. The reason is that a storage node with computational ability can compute a linear combination of what it stores. That is, consider an (Nm, M) linear code C , with generator matrix $(\mathbf{v}_1, \dots, \mathbf{v}_{Nm}) \in \text{Mat}_{M \times Nm}(\mathbb{F}_q)$, with columns $\mathbf{v}_i \in \mathbb{F}_q^{M \times 1}$, $i = 1, \dots, Nm$, as described in Chapter 2. Then a data object $\mathbf{o} \in \mathbb{F}_q^{1 \times M}$ is encoded into a codeword

$(\mathbf{ov}_1, \dots, \mathbf{ov}_{Nm}) \in \mathbb{F}_q^{1 \times Nm}$, and the Nm codeword coefficients are distributed into N storage nodes, each node storing m coefficients. As said above, each storage node is assumed to be equipped with computational power, therefore, a node storing $\mathbf{ov}_{i_1}, \dots, \mathbf{ov}_{i_m}$ is able to compute $\mathbb{F}_q$ -linear combinations $\sum_{j=1}^m \alpha_j \mathbf{ov}_{i_j}$. From this point of view, it makes sense to therefore model each storage node as storing an m -dimensional subspace of $\mathbb{F}_q^M$ with $\mathbb{F}_q$ -basis $\mathbf{v}_{i_1}, \dots, \mathbf{v}_{i_m}$ (we may assume without loss of generality that $\mathbf{v}_{i_1}, \dots, \mathbf{v}_{i_m}$ are linearly independent), as explicitly suggested in [14].

Under the above framework, the problem of designing storage codes then becomes that of finding N m -dimensional subspaces of $\mathbb{F}_q^M$, which naturally brings in the Grassmannian $G_q(m, M)$, which is by definition the set of all subspaces of $\mathbb{F}_q^M$ of dimension m . A first attempt to treat storage codes in this manner was presented in [15], using Grassmannian graphs, which yielded some interesting constructions. A systematic approach to design subspace codes for storage has been recently proposed in [16], which relies on Plücker embeddings. Subspace codes were in general studied from a group action point of view in [17], which introduced the concept of orbit codes. We firstly recall the definition of orbit codes.

6.1 Storage Codes from Orbit Codes

Let $\mathbb{F}_q$ be the finite field with q elements, with q a prime power, and let $G_q(m, M)$ be the set of all subspaces of $\mathbb{F}_q^M$ of dimension m , called *Grassmannian*:

$$G_q(m, M) = \{\mathcal{U} \text{ subspace of } \mathbb{F}_q^M, \dim(\mathcal{U}) = m\}.$$

We denote by $GL_M(\mathbb{F}_q)$ the set of $M \times M$ invertible matrices with coefficients in $\mathbb{F}_q$. Multiplication by elements of $GL_M(\mathbb{F}_q)$ defines a group action from the right on $G_q(m, M)$ by

$$\begin{aligned} G_q(m, M) \times GL_M(\mathbb{F}_q) &\longrightarrow G_q(m, M) \\ (\mathcal{U}, A) &\longmapsto \mathcal{U}A = \{uA, u \in \mathcal{U}\} \end{aligned}$$

since any element of $GL_M(\mathbb{F}_q)$ maps an m -dimensional subspace to an m -dimensional subspace. In fact, as pointed out in [17], since any two m -dimensional subspaces can be mapped onto each other by an element of $GL_M(\mathbb{F}_q)$, $GL_M(\mathbb{F}_q)$ acts transitively on $G_q(m, M)$.

Fix $\mathcal{U} \in G_q(m, M)$, let G be a subgroup of $GL_M(\mathbb{F}_q)$, and let $\mathcal{U}G = \{\mathcal{U}g, g \in G\}$ be the orbit of $\mathcal{U}$ under the right action of G . We will refer to $\mathcal{U}G$ as an *orbit code*, following the terminology of [17]. If furthermore G is cyclic, we call $\mathcal{U}G$ a *cyclic orbit code*.

In order to represent an m -dimensional subspace $\mathcal{U}$ of the vector space $\mathbb{F}_q^M$, we fix a basis of $\mathcal{U}$, and use an $m \times M$ matrix U whose row space $\{vU, v \in \mathbb{F}_q^m\}$ is $\mathcal{U}$.

Orbit codes form collections of m -dimensional subspaces of $\mathbb{F}_q^M$ obtained by group action. We will study their potential to provide codes for distributed storage. Let us firstly see how the storage parameters are translated into parameters for an orbit code $\mathcal{U}G$. Remember that in the context of this chapter, an object represented by a row vector is encoded by a collection of m -dimensional subspaces of $\mathbb{F}_q^M$, and the encoding is done by multiplying this row vector by the column vectors of these m -dimensional subspaces. Therefore, it is naturally to assume

that the object $\mathbf{o}$ is of length M .

6.1.1 Code Length

The number N of storage nodes is the length of the code, that is the cardinality $|\mathcal{U}G|$ of the orbit $\mathcal{U}G$, and it is well known that

$$|\mathcal{U}G| = \frac{|G|}{|Stab_G(\mathcal{U})|},$$

where $Stab_G(\mathcal{U}) = \{g \in G, \mathcal{U}g = \mathcal{U}\}$ is a subgroup of G called the stabilizer of $\mathcal{U}$. Alternatively

$$|\mathcal{U}G| = \frac{|G|}{|G \cap Stab_{GL_M(\mathbb{F}_q)}(\mathcal{U})|}.$$

Indeed, if $g \in Stab_G(\mathcal{U})$, then $g \in G$ and g is also an element of $GL_M(\mathbb{F}_q)$ such that $\mathcal{U}g = \mathcal{U}$. For the reverse inclusion, similarly, if $g \in G$ and $\mathcal{U}g = \mathcal{U}$, then $g \in Stab_G(\mathcal{U})$.

Lemma 6.1.1. *Let $\mathcal{U}_I$ be the m -dimensional subspace generated by the first m unit vectors, that is, it has $U_I = (I_m, 0_{m \times (M-m)})$ as an $\mathbb{F}_q$ -basis. Then*

$$Stab_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I) = \left\{ \begin{bmatrix} A & 0 \\ B & C \end{bmatrix}, A \in GL_m(\mathbb{F}_q), B \in \mathbb{F}_q^{(M-m) \times m}, C \in GL_{M-m}(\mathbb{F}_q) \right\}.$$

Proof. Take g in $Stab_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I)$. Note that $U_I g = (A, 0_{m \times (M-m)})$ and since A is invertible, then $(A, 0)$ generates $\mathcal{U}_I$. Then B can take any value, but C needs to be invertible, to ensure that g is. $\square$

We keep the above notation in what follows.

Lemma 6.1.2. *For any m -dimensional subspace $\mathcal{U}$, we have*

$$\text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}) = L^{-1} \text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I) L,$$

for L such that $\mathcal{U} = \mathcal{U}_I L$, $L \in GL_M(\mathbb{F}_q)$.

Proof. Note first that any m -dimensional subspace $\mathcal{U}$ is of the form $\mathcal{U} = \mathcal{U}_I L$ for some L in $GL_M(\mathbb{F}_q)$. Take $g \in \text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U})$, then by definition

$$\mathcal{U}g = \mathcal{U} \Leftrightarrow \mathcal{U}_I Lg = \mathcal{U}_I L \Leftrightarrow \mathcal{U}_I Lg L^{-1} = \mathcal{U}_I,$$

showing that $Lg L^{-1} \in \text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I)$, that is $g \in L^{-1} \text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I) L$.

Conversely, take an element $L^{-1}g L \in L^{-1} \text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I) L$, then

$$\mathcal{U} L^{-1} g L = \mathcal{U}_I L L^{-1} g L = \mathcal{U}_I g L = \mathcal{U}_I L = \mathcal{U}$$

and $L^{-1}g L$ indeed belongs to the stabilizer of $\mathcal{U}$. □

Using the above lemma, we have that

$$|\text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U})| = |L^{-1} \text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I) L| = |\text{Stab}_{GL_M(\mathbb{F}_q)}(\mathcal{U}_I)|. \quad (6.1.1)$$

6.1.2 Storage Capacity

The storage capacity (or number of stored symbols in $\mathbb{F}_q$) for every node is m , This is because a subspace $\mathcal{U}$ of dimension m means that the corresponding node actually stores $U \mathbf{o}^T \in \mathbb{F}_q^m$, where we recall that $\mathbf{o} \in \mathbb{F}_q^{1 \times M}$ is the data object to be stored. This notion was introduced in [14]. Under the hypothesis that a stored

object $\mathbf{o} \in \mathbb{F}_q^{1 \times M}$ should be recoverable from any k nodes, the minimal value of m is M/k .

6.1.3 Fault tolerance

The ability to recover a stored object $\mathbf{o} \in \mathbb{F}_q^{1 \times M}$ from any k nodes, thus providing protection against any $N - k$ failures is rephrased as:

$$\dim(\mathcal{U}_{i_1} + \dots + \mathcal{U}_{i_k}) = M.$$

Note that when $m = 1$, then, replacing ourselves in the context of the beginning of this chapter, every node stores one column of a generator matrix of a linear code, and the property of fault tolerance just discussed is that of a maximum distance separable (MDS) code.

For arbitrary k , there is no formula to evaluate $\dim(\mathcal{U}_{i_1} + \dots + \mathcal{U}_{i_k})$, apart from using iteratively the well known formula when $k = 2$ which states that

$$\dim(\mathcal{U}g^i + \mathcal{U}g^j) = \dim(\mathcal{U}g^i) + \dim(\mathcal{U}g^j) - \dim(\mathcal{U}g^i \cap \mathcal{U}g^j). \quad (6.1.2)$$

Example 6.1.3. Consider the Grassmannian $G_2(3, 5)$, and let $G = \langle g \rangle$ be the

subgroup of $GL_5(\mathbb{F}_2)$ generated by

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

Consider the orbit code $\mathcal{U}G$, where $\mathcal{U} = \mathcal{U}_I$ has for basis

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

The length N of the orbit code $\mathcal{U}G$ is

$$N = |\mathcal{U}G| = \frac{|G|}{|Stab_G(\mathcal{U})|} = 4.$$

Indeed, g has order 4

$$g^2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad g^3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad g^4 = I_5,$$

and $Stab_G(\mathcal{U})$ is trivial, since G intersects trivially $Stab_{GL_5(\mathbb{F}_2)}(\mathcal{U})$ which is formed

of matrices whose first 3×3 block is invertible (see Lemma 6.1.1). The orbit code $\mathcal{UG}$ is

$$U, \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

This corresponds to a storage code where the size of the object $\mathbf{o}$ is $M = 5$, every storage node stores $m = 3$ symbols, e.g., the first node corresponding to the subspace $\mathcal{U}$ stores $(1, 0, 0, 0, 0)\mathbf{o}^T = o_1$, $(0, 1, 0, 0, 0)\mathbf{o}^T = o_2$, $(0, 0, 1, 0, 0)\mathbf{o}^T = o_3$ for $\mathbf{o} = (o_1, \dots, o_5) \in \mathbb{F}_2^5$, giving the following storage allocation:

$$\begin{aligned} \text{Node 1 : } & o_1, o_2, o_3, & \text{Node 3 : } & \sum_i o_i, o_2, o_4 \\ \text{Node 2 : } & o_3, o_5, \sum_i o_i, & \text{Node 4 : } & o_4, o_1, o_5. \end{aligned}$$

The data object $\mathbf{o}$ may be retrieved out of any two nodes, since, from (6.1.2)

$$\dim(\mathcal{U}g^i + \mathcal{U}g^j) = 2 \dim(\mathcal{U}g^i) - 1 = 5$$

for all $i \neq j$. Indeed, two distinct subspaces intersect in a subspace of dimension 1.

6.1.4 Repair

As pointed out in the Chapter 1, a storage code should be amenable to repair, namely, the code should be such that the data stored at one node can be computed from a (small) subset of other nodes, without (necessarily) having to decode the object first. The parameters of importance are typically the repair bandwidth and locality.

- Decomposing the stored data per node into m pieces is beneficial to the communication cost: it allows a repair to download part of the stored data. The optimal repair bandwidth per failed node is m , since at least the content of a failed node must be communicated for a repair to be successful.
- To speed up repairs, contacting few live nodes is beneficial [3]. The optimal number of nodes is thus 2 in the case of one failure (since less than 2 means that a repetition code is used), or 1 in the (partial) collaborative case.

One may consider either repair of one failure at a time, or (possibly) wait for several failures to occur before repairing t of them together.

Example 6.1.4. We continue Example 6.1.3, where we have for $\mathbf{o}$ of size 5

$$\begin{aligned} \text{Node 1 : } o_1, o_2, o_3, & \quad \text{Node 3 : } \sum_i o_i, o_2, o_4 \\ \text{Node 2 : } o_3, o_5, \sum_i o_i, & \quad \text{Node 4 : } o_4, o_1, o_5. \end{aligned}$$

In case of one node failure, the subspace $\mathcal{U}g^i$ may be computed from the knowledge of a subspace of dimension 1 from each of the other $\mathcal{U}g^j$, $j \neq i$, which follows from the fact that $\dim(\mathcal{U}g^i \cap \mathcal{U}g^j) = 1$ for all $i \neq j$. For example, if Node 1 were to fail, it can be recomputed using o_1 from Node 4, o_2 from Node 3, and o_3 from Node 2. This code instance has been reported in [19], not in the language of orbit codes, but as a code whose repair bandwidth is minimal.

While the motivation example presented above is optimal for the case of one failure, most of our focus next will be on the collaborative case.

6.2 Instances of Cyclic Orbit Codes using Companion Matrices

We next propose a family of cyclic orbit codes based on companion matrices and compute the parameters of the corresponding storage codes.

Definition 6.2.1. Let $p(x) = \sum_{i=0}^{M-1} p_i x^i + x^M$ be a monic polynomial in $\mathbb{F}_q[x]$. Then the *companion matrix* M_p of $p(x)$ is defined by

$$M_p = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & & 1 \\ -p_0 & -p_1 & -p_2 & \dots & -p_{M-1} \end{bmatrix} \in \mathbb{F}_q^{M \times M}. \quad (6.2.1)$$

We first discuss basic properties of codes, namely length and fault tolerance, before proposing code constructions.

6.2.1 Basic Properties

The following is well known and is proven for the sake of completeness, for more details, see [20].

Lemma 6.2.2. Let $G = \langle g \rangle$ be the subgroup of $GL_M(\mathbb{F}_q)$ generated by the $M \times M$

companion matrix

$$g = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & & 1 \\ -p_0 & -p_1 & -p_2 & \dots & -p_{M-1} \end{bmatrix} \quad (6.2.2)$$

of a polynomial $p(x) = \sum_{i=0}^{M-1} p_i x^i + x^M \in \mathbb{F}_q[x]$, $p_0 \neq 0$. Then the order of g is the smallest positive integer l such that $p(x)$ divides $x^l - 1$.

Proof. It is well known that $p(x)$ is the minimal polynomial of M_p , that is $p(x)$ is the monic polynomial of the companion matrix M_p defined in (6.2.1), namely $p(x)$ is the monic polynomial of least degree such that $p(M_p) = 0$. Now since $p_0 \neq 0$, there exists a positive integer $e \leq q^M - 1$ such that $p(x)$ divides $x^e - 1$ (see [20, Lemma 3.1]), so take l to be the smallest positive integer with the property that $p(x) | x^l - 1$. Consequently, $M_p^l = I_M$ if and only if $p(x) | x^l - 1$. $\square$

One advantage to deal with companion matrices is that we are sure that no matter the choice of $\mathcal{U}$, the resulting code will not be trivial (by trivial, we mean containing $\mathcal{U}$ only).

Corollary 6.2.3. *Let $G = \langle g \rangle$ be the subgroup generated by the companion matrix of the polynomial $p(x) = \sum_{i=0}^M x^i \in \mathbb{F}_q[x]$. Then $|G| = M + 1$.*

Proof. We have $(x - 1) \sum_{i=0}^M x^i = x^{M+1} - 1$. $\square$

Proposition 6.2.4. *Let g be the companion matrix of $p(x)$ as in (6.2.2), and let $\mathcal{U}$ be an m -dimensional subspace in $G_q(m, M)$. Then the orbit code $\mathcal{U}\langle g \rangle$ is not*

trivial ($|\mathcal{U}\langle g \rangle| \geq 2$).

Proof. Consider g as a linear map from $\mathbb{F}_q^M$ to $\mathbb{F}_q^M$, and at $\mathbb{F}_q^M$ as an $\mathbb{F}_q[x]$ -module over the polynomial ring $\mathbb{F}_q[x]$, its invariant subspaces are given by the Smith normal form of $xI_M - g$. Therefore compute the $l \times l$ minors of

$$xI_M - g = \begin{bmatrix} x & -1 & 0 & \dots & 0 \\ 0 & x & -1 & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & & -1 \\ p_0 & p_1 & p_2 & \dots & x + p_{M-1} \end{bmatrix}$$

and let d_l denote the greatest common divisor of all the $l \times l$ minors. Then $d_1 = d_2 = \dots = d_{M-1} = \pm 1$, which shows that there is no invariant subspace of dimension m , $1 \leq m \leq M - 1$. $\square$

While any choice of $\mathcal{U}$ does give a non-trivial orbit code, different choices of $\mathcal{U}$ may give different sizes of orbit codes, as illustrated next.

Example 6.2.5. Choose $G = \langle g \rangle$ with

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

which is the companion matrix of $x^4 + x + 1 \in \mathbb{F}_2[x]$. Since $x^4 + x + 1$ is an irreducible polynomial, let ω be a root, that is $\omega^4 = \omega + 1$, and $\mathbb{F}_2[x]/(x^4 + x + 1) \simeq \mathbb{F}_2(\alpha) \simeq$

$\mathbb{F}_{2^4}$. Since ω is a primitive element of $\mathbb{F}_{2^4}$, the order of G is 15.

Consider $\mathcal{U}_I \in G_2(2, 4)$ with canonical basis $U_I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$. By Lemma 6.1.1, the stabilizer of $\mathcal{U}_I$ under G is trivial, since G intersects trivially $Stab_{GL_4(\mathbb{F}_2)}(\mathcal{U}_I)$, which is formed of matrices whose first 2×2 block is invertible. Therefore the length of the code $\mathcal{U}_I G$ is 15.

To illustrate the fact that a change in the choice of $\mathcal{U}$ may change the length of the corresponding orbit code, we look for powers of g^i whose set of invariant subspaces is non-empty. Take g^5 , given by

$$g^5 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

and as in Proposition 6.2.4, and keeping the same notation, compute the Smith normal form of $xI_4 - g^5$, to find that $d_1 = d_2 = 1$, while $d_3 = x^2 + x + 1$, and $d_4 = x^4 + x^2 + 1$. Therefore the invariant factors are

$$d_1 = 1, \quad \frac{d_2}{d_1} = 1, \quad \frac{d_3}{d_2} = x^2 + x + 1, \quad \frac{d_4}{d_3} = x^2 + x + 1,$$

revealing the presence of invariant subspaces of dimension 2. Thus, let $\mathcal{U} \in$

$G_2(2, 4)$ and $\mathcal{U} = \mathcal{U}_I L$, where

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

so that a basis is given by

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

Since $\mathcal{U}g^5 = \mathcal{U}$, $\mathcal{U}g^{10} = (\mathcal{U}g^5)g^5 = \mathcal{U}$, and the stabilizer of $\mathcal{U}$ under G has cardinality 3, given by

$$\text{Stab}_G(\mathcal{U}) = \{I_4, g^5, g^{10}\},$$

so that the orbit code $\mathcal{U}G$ with $\mathcal{U} = \mathcal{U}_I L$, is of length 5, while $\mathcal{U}_I G$ is of length 15.

Lemma 6.2.6. *Let g be the companion matrix of $p(x)$ with g and $p(x)$ as in (6.2.2), $G = \langle g \rangle$, and let $\mathcal{U}_I \in G_q(m, M)$ be the subspace generated by a canonical basis as above. If $q = 2$, then the stabilizer $\text{Stab}_G(\mathcal{U}_I)$ is trivial. Consequently the size of the orbit code is the order of g .*

Proof. Let l be the order of g , and suppose that the stabilizer of $\mathcal{U}_I$ under $G = \langle g \rangle$ is not trivial, that is, there exists an s , $0 < s < l$, such that $\mathcal{U}_I g^s = \mathcal{U}_I$. Then by Lemma 6.1.1, there exist $A \in GL_m(\mathbb{F}_q)$, $B \in \mathbb{F}_q^{(M-m) \times m}$, $C \in GL_{M-m}(\mathbb{F}_q)$ such that

$$g^s = \begin{bmatrix} A & 0 \\ B & C \end{bmatrix}.$$

Note that $g^s \cdot g = g \cdot g^s$, therefore the first m rows of $g^s \cdot g$ and $g \cdot g^s$ are equal,

and given respectively by

$$\begin{bmatrix} 0_{m \times 1} & A & 0_{m \times (M-m-1)} \end{bmatrix} = \begin{bmatrix} a_2 & 0 \\ a_3 & 0 \\ \vdots & \vdots \\ a_m & 0 \\ b_1 & c_1 \end{bmatrix},$$

where a_i, b_i, c_i denote the respective i th row of A, B and C . Equating coefficient-wise shows that

$$A = a_{11}I_m, \quad b_1 = 0, \quad c_1 = (a_{11}, 0, \dots, 0).$$

Repeating the same computation by noting that the first m rows of $g^{s+1} \cdot g$ equal those of $g \cdot g^{s+1}$, we get

$$b_2 = 0, \quad c_2 = (0, a_{11}, 0, \dots, 0),$$

and more generally, comparing the first m rows of $g^{s+t} \cdot g$ and $g \cdot g^{s+t}$, $t = 1, \dots, M-m-1$, yields

$$b_j = 0, \quad c_j = (0, \dots, a_{11}, \dots, 0), \quad j = 1, \dots, M-m,$$

where only the j th coordinate of c_j is non-zero.

Therefore

$$g^s = a_{11}I_M,$$

and since $q = 2$, the only choice is $a_{11} = 1$. This is a contradiction, since g has

order $l > s$. □

Note that the same argument breaks at $g^s = a_{11}I_M$ for $q \neq 2$. For example, when $q = 3$, $p(x) = x^3 + 2$, then g has order 6, and $g^3 = 2I_3$.

6.2.2 Code Construction I: t Failures with Minimized Bandwidth

We saw above that under the same group action, $\mathcal{U}G$ and $\mathcal{U}_I G$ may yield different code lengths, due to different stabilizers. In this section, we thus consider the canonical case $\mathcal{U}_I G$.

Lemma 6.2.7. *Take $\mathcal{U}, \mathcal{U}' \in G_q(M-1, M)$, $\mathcal{U} \neq \mathcal{U}'$. Then $\mathcal{U}$ and $\mathcal{U}'$ intersect in a subspace of dimension $M-2$.*

Proof. Since by (6.1.2)

$$\dim(\mathcal{U} + \mathcal{U}') = 2(M-1) - \dim(\mathcal{U} \cap \mathcal{U}') \leq M,$$

it must be that

$$M-2 \leq \dim(\mathcal{U} \cap \mathcal{U}')$$

which concludes the proof, since $\mathcal{U}$ and $\mathcal{U}'$ are distinct. □

The following lemma describes the behaviour of intersection of three subspaces, and shows that the considered orbit codes exhibit a particular structure, namely any three subspaces intersect in a subspace of dimension $M-3$.

Lemma 6.2.8. *Take $\mathcal{U}, \mathcal{U}', \mathcal{U}''$ three distinct elements of $G_q(M-1, M)$. Then $\mathcal{U}, \mathcal{U}'$ and $\mathcal{U}''$ intersect in a subspace of dimension either $M-2$ or $M-3$. When*

$\mathcal{U}$, $\mathcal{U}'$, $\mathcal{U}''$ belong to an orbit code $\mathcal{U}\langle g \rangle$ with g the companion matrix of $p(x)$ as in (6.2.2), the dimension of their intersection is $M - 3$.

Proof. Since by (6.1.2)

$$\dim(\mathcal{U} \cap \mathcal{U}' \cap \mathcal{U}'') = \dim(\mathcal{U}) + \dim(\mathcal{U}' \cap \mathcal{U}'') - \dim(\mathcal{U} + (\mathcal{U}' \cap \mathcal{U}')),$$

and we know that $\dim(\mathcal{U}) = M - 1$ and $\dim(\mathcal{U}' \cap \mathcal{U}'') = M - 2$ by the above lemma, we have that

$$\dim(\mathcal{U} \cap \mathcal{U}' \cap \mathcal{U}'') = 2M - 3 - \dim(\mathcal{U} + (\mathcal{U}' \cap \mathcal{U}')).$$

Now either $(\mathcal{U}' \cap \mathcal{U}'') \subset \mathcal{U}$ and $\dim(\mathcal{U} + (\mathcal{U}' \cap \mathcal{U}')) = M - 1$, or $(\mathcal{U}' \cap \mathcal{U}'') \not\subset \mathcal{U}$ and $\dim(\mathcal{U} + (\mathcal{U}' \cap \mathcal{U}')) = M$, which proves the first claim.

Consider now the case where the intersection is $M - 2$. Suppose there exist three subspaces A, B, C which intersect in a subspace of dimension $M - 2$. Let D be another subspace. Then D intersects B in a subspace of dimension $M - 2$. Either this subspace is $A \cap B \cap C$, or not. We will suppose not, and show that this yields a contradiction. Then since B and D each are $(M - 1)$ -dimensional, it must be that $B \cap D$ is formed by an $(M - 3)$ -dimensional subspace of $A \cap B \cap C$, and a one-dimensional subspace that belongs to B , and not to A , or to C . Now D also intersects C in an $(M - 2)$ subspace, and it already intersects it in a $(M - 3)$ -subspace, therefore the intersection of C and D further contains a one dimensional subspace that belongs to C , and not to A and B . But by now, the $(M - 1)$ -dimensional subspace D is fully determined, it is made of a $(M - 3)$ -dimensional subspace that belongs to $A \cap B \cap C$, and two other one-dimensional

subspaces, one, say c , belonging to C (and no to A or B) and one, say b , belonging to B (and not to C or A). Finally, D must intersect A in a $(M - 2)$ -dimensional subspace. So either b or c must belong to A . But this not possible, because if b belongs to A , then the intersection between B and A becomes of dimension $M - 1$, and similarly for if c were to belong to A , then the intersection of A and C would be $n - 1$.

This shows that if there exist three subspaces that intersect in a subspace of dimension $M - 2$, then any other subspace will intersect them in the same subspace of dimension $M - 2$. Clearly this configuration cannot happen for the orbit code generated from a companion matrix. To see this, it is enough to consider $\mathcal{U}g$, $\mathcal{U}g^2$ and $\mathcal{U}g^3$ to see that $\mathcal{U}g^3$ contains a one-dimensional subspace which is common to $\mathcal{U}g^2$ but not to $\mathcal{U}g$. $\square$

We propose next a simple construction which will turn out to provide the optimal repair bandwidth, since it requires per failure the download of exactly the stored amount per node.

Proposition 6.2.9. *Let g be the companion matrix of $p(x)$ as in (6.2.2) and let U_I contain a canonical basis of $\mathcal{U}_I \in G_2(M - 1, M)$. Then the orbit code $\mathcal{U}G$ has the following properties:*

1. *it is of length l which is the order of g ,*
2. *the object is retrieved by contacting any two nodes,*

Furthermore, for $M \geq 4$, if $p(x)$ is chosen as in Corollary 6.2.3, then the repair of t failures, $t \geq 2$, is done by downloading $M - 2$ elements of $\mathbb{F}_2$ from one node for each failure, and exchanging one element of $\mathbb{F}_2$ between the two repair nodes.

Proof. The claim on the length comes from Lemma 6.2.6. Label the l storage nodes from 0 to $l - 1$. By assumption, the i th node stores the element $\mathcal{U}g^i$ of the orbit $\mathcal{U}G$ of $\mathcal{U}$ under the right action of G , $i = 0, 1, \dots, l - 1$.

To retrieve a stored object $\mathbf{o} \in \mathbb{F}_2^{1 \times l}$ from any 2 nodes, or equivalently to protect against any $l - 2$ failures, we need that for a choice of 2 subspaces from the code $\mathcal{U}G$,

$$\dim(\mathcal{U}_{i_1} + \mathcal{U}_{i_2}) = M,$$

and the claim follows from Lemma 6.2.7.

Recall Subsection 6.1.2 for counting the size of elements downloaded/repaired.

Suppose two nodes have failed, say node s and node t , download from node i the subspace $A = \mathcal{U}g^i \cap \mathcal{U}g^s$ of dimension $M - 2$, and from node j the subspace $B = \mathcal{U}g^j \cap \mathcal{U}g^t$ also of dimension $M - 2$, by the above lemma. If $\dim(A \cup B) = M$, the repair process can be completed by collaboration, by exchanging the missing basis vector at each repair node. Indeed, since $\dim(A \cup B) = M$, write the missing basis vector a at node s in a basis $\{v_1, \dots, v_M\}$ of $A \cup B$ as $a = \sum_{i=1}^M a_i v_i$. If $v_1, \dots, v_l \in A$, ask the symbol $\sum_{i=l+1}^M a_i v_i$ from B . Iterate this process for the missing basis vector at node t .

We are left to show that $\dim(A \cup B) = M$, or in fact, since

$$\dim(A \cup B) = \dim(A) + \dim(B) - \dim(A \cap B) = 2(M - 2) - \dim(A \cap B),$$

to show that $\dim(A \cap B) = M - 4$.

Write $\mathcal{U}g^i = \langle g_1, \dots, g_{M-2}, g_{i_0} \rangle$, $\mathcal{U}g^j = \langle g_1, \dots, g_{M-2}, g_{j_0} \rangle$. Then $A = \mathcal{U}g^s \cap \mathcal{U}g^i = \langle g_1, \dots, g_{s_0-1}, g_{s_0+1}, \dots, g_{M-2}, g_{i_0} \rangle$, otherwise the nodes i , j and s would intersect in

the same subspace of dimension $M - 2$, which is not possible by Lemma 6.2.8, and for the same reason $B = \mathcal{U}g^t \cap \mathcal{U}g^j = \langle g_1, \dots, g_{t_0-1}, g_{t_0+1}, \dots, g_{M-2}, g_{j_0} \rangle$. Without loss of generality, suppose that $s_0 < t_0$. Then

$$A \cap B = \langle g_1, \dots, g_{s_0-1}, g_{s_0+1}, \dots, g_{t_0-1}, g_{t_0+1}, \dots, g_{M-2} \rangle$$

which has dimension $M - 4$.

Suppose that t nodes have failed, say node w , $w = 0, \dots, t - 1$, which each downloads from node $w + t$ the subspace $\mathcal{U}g^w \cap \mathcal{U}g^{w+t}$ of dimension $M - 2$ (the labeling of the nodes is done for simplicity and without loss of generality). Each repaired node, after the download phase, is missing only one 1-dimensional subspace. By the above claim, any pair $A := \mathcal{U}g^w \cap \mathcal{U}g^{w+t}$, $B := \mathcal{U}g^{w'} \cap \mathcal{U}g^{w'+t}$ has the property that $\dim((\mathcal{U}g^w \cap \mathcal{U}g^{w+t}) + (\mathcal{U}g^{w'} \cap \mathcal{U}g^{w'+t})) = M$, thus every repaired node can contact one other node to complete the repair process. Indeed, since $\dim(A+B) = M$, write the missing basis vector a at node w in a basis $\{v_1, \dots, v_M\}$ of $A + B$ as $a = \sum_{i=1}^M a_i v_i$. If $v_1, \dots, v_l \in A$, ask the symbol $\sum_{i=l+1}^M a_i v_i$ from B in node w' . Iterate this process for the missing basis vector at other nodes. $\square$

We remark that this code requires the download of $M - 1$ symbols per failure, for a storage of $M - 1$ symbols per node, thus is using the minimal possible bandwidth. It is a collaborative code for $t = 2$, since two nodes are participating to the repair, and they are exchanging data among each other, and a partially collaborative code for $t \geq 3$, since then each repaired node only needs to communicate with one other node.

Example 6.2.10. Consider $G_2(3, 4)$, and the subspace $\mathcal{U}$, with basis

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

Let $G = \langle g \rangle$ be the cyclic subgroup of $GL_4(\mathbb{F}_q)$ generated by

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

The order of g is 5 and the elements of the orbit codes are explicitly given by

$$U, \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

We store them in node 0 to 4. Assume that node 1 and node 4 failed. To repair node 1 and 4, we have

Download phase.

- node 1 gets $\{(0010), (0001)\}$ from node 2,
- node 4 gets $\{(1000), (0100)\}$ from node 0.

Collaboration phase. The node repairing node 1 can compute (0011) and send it, and the node repairing node 4 sends (0100) in exchange.

Note that the strategy is not unique. To repair node 1, download alternatively $\{(0001), (0110)\}$ from node 3, to repair node 4, get $\{(1100), (1111)\}$ from node 2, then the node repairing node 1 can get (0011) and the one repairing node 4 can get (0111) from the collaboration.

6.2.3 Code Construction II: three Failures and more

We propose a construction that handles more than two failures.

Proposition 6.2.11. *For $M \geq 3$, we consider g as the companion matrix of the polynomial $p(x)$ as in (6.2.2), and $p(x)$ is a primitive polynomial in $\mathbb{F}_2[x]$ which has the order $2^M - 1$. Let U_I contain a canonical basis of $\mathcal{U}_I \in G_2(M-1, M)$. Then the repair of three failures is done by connecting two nodes and downloading $M-2$ elements of $\mathbb{F}_2$ from each node for each failure, and exchanging one element of $\mathbb{F}_2$ with every other repair node.*

Proof. A node repairing the i th node which contains $\mathcal{U}_I g^i$ connects to two nodes j, k , respectively storing $\mathcal{U}_I g^j$ and $\mathcal{U}_I g^k$, and downloads $A = \mathcal{U}_I g^i \cap \mathcal{U}_I g^j$ of dimension $M-2$ from node j , $B = \mathcal{U}_I g^i \cap \mathcal{U}_I g^k$ of dimension $M-2$ from node k . If $A \neq B$, then $A \cup B = \mathcal{U}_I g^i$, and the i th node is recovered. So we only need to consider $A = B$.

There are $2^M - 1$ elements of $G_2(M-1, M)$, the order of g is $2^M - 1$, the size of $\mathcal{U}_I \langle g \rangle$ is $2^M - 1$, so it is possible to download the same A and B . Also, there are at most three subspaces in $G_2(M-1, M)$ which intersect in the same subspace of dimension $M-2$.

Suppose we have three failures, nodes i, j, k . After the download process, the three nodes repairing them all have downloaded a subspace of dimension

$M - 2$, denoted by A, B, C respectively. Suppose that node i downloaded from the subspaces A_1, A_2 , node j downloaded from B_1, B_2 , node k downloaded from C_1, C_2 . The dimension of the intersection subspaces of two of A, B, C , can take only three possibilities:

(1) If two of A, B, C intersect in a subspace of dimension $M - 2$, for example $\dim(A \cap B) = M - 2$, then we have $\dim(A_1 \cap A_2 \cap B_1 \cap B_2) = M - 4$, which is impossible.

(2) If two of A, B, C intersect in a subspace of dimension $M - 4$, say $\dim(A \cap B) = M - 4$, then $\dim(A \cup B) = M$, then after the exchange process, the repair completed.

(3) We are left to consider the situation where any two of A, B, C intersect in a subspace of dimension $M - 3$. Without loss of generality, let

$$A = \{g_1, \dots, g_{M-2}\},$$

$$B = \{g_1, \dots, g_{s-1}, g_{s+1}, \dots, g_{M-2}, y_{B_0}\},$$

$$C = \{g_1, \dots, g_{t-1}, g_{t+1}, \dots, g_{M-2}, y_{C_0}\},$$

then $\dim(A \cap B) = M - 3$, $\dim(A \cap C) = M - 3$, we need $\dim(B \cap C) = M - 3$ as well. From the assumption of A, B, C , and any two of $\mathcal{U}_I g^i, \mathcal{U}_I g^j, \mathcal{U}_I g^k$ intersect in a subspace of dimension $M - 2$, we can let

$$\mathcal{U}_I g^i = \{g_1, \dots, g_{M-2}, y_A\},$$

$$\mathcal{U}_I g^j = \{g_1, \dots, g_{s-1}, g_{s+1}, \dots, g_{M-2}, y_A, y_B\},$$

$$\mathcal{U}_I g^k = \{g_1, \dots, g_{t-1}, g_{t+1}, \dots, g_{M-2}, y_A, y_C\},$$

If $s = t$, $\dim(B \cap C) = M - 3$, we must have $y_{B_0} \neq y_{C_0}$, then

$$A \cup B \cup C = \{g_1, \dots, g_{M-2}, y_{B_0}, y_{C_0}\},$$

so $\dim(A \cup B \cup C) = M$, each of the nodes exchanges one element with every other two nodes and repair is completed.

If $s \neq t$, $\dim(B \cap C) = M - 3$, we must have $y_{B_0} = y_{C_0}$. When $y_{B_0} = y_{C_0} = y_A$, we get

$$\begin{aligned} \mathcal{U}_I g^i \cap \mathcal{U}_I g^j &= \{g_1, \dots, g_{s-1}, g_{s+1}, \dots, g_{M-2}, y_A, \} \\ &= B_1 \cap B_2, \end{aligned}$$

which is impossible. When $y_{B_0} = y_{C_0} \neq y_A$, we can let

$$B_1 = \{g_1, \dots, g_{s-1}, g_{s+1}, \dots, g_{M-2}, y_{B_0}, y_{B_1}\},$$

$$B_2 = \{g_1, \dots, g_{s-1}, g_{s+1}, \dots, g_{M-2}, y_{B_0}, y_{B_2}\},$$

$$C_1 = \{g_1, \dots, g_{t-1}, g_{t+1}, \dots, g_{M-2}, y_{B_0}, y_{C_1}\},$$

$$C_2 = \{g_1, \dots, g_{t-1}, g_{t+1}, \dots, g_{M-2}, y_{B_0}, y_{C_2}\},$$

B_1, B_2, C_1, C_2 also being subspaces in $G_2(M-1, M)$. Then using $\dim(B_1 \cap C_1) = M - 2$, we have $y_{B_1} = y_{C_1}$, since $\dim(B_1 \cap C_2) = M - 2$, we get $y_{B_1} = y_{C_2}$, and we have $y_{C_1} = y_{C_2}$, which is a contradiction.

So we conclude that when any two of A, B, C intersect in a subspace of dimen-

sion $M - 3$, $\dim(A \cup B \cup C) = M$, then the repair can be realized. $\square$

Corollary 6.2.12. *The code construction and repair strategy in Proposition 6.2.11 still hold for more than three failures.*

Proof. From Proposition 6.2.11, any three repairing nodes can span the whole space $\mathbb{F}_2^M$, so for the exchange process, exchanging with two other repairing nodes is enough. $\square$

Example 6.2.13. Consider $G_2(2, 3)$, and the subspace $\mathcal{U}$, with basis

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

Let $G = \langle g \rangle$ be the cyclic subgroup of $GL_3(\mathbb{F}_q)$ generated by the companion matrix g of the primitive polynomial $p(x) = x + 1$, that is

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

The order of g is 7 and the elements of the orbit codes are explicitly given by

$$U, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

We store them in node 0 to 7. Assume that node 1, node 2 and node 4 failed. To repair these nodes, we have

Download phase.

- node 1 gets $\{(010), (100)\}$ from node 0 and node 7,
- node 2 gets $\{(110), (101)\}$ from node 3 and node 6,
- node 4 gets $\{(111), (011)\}$ from node 6 and node 3,

Collaboration phase. The node repairing node 2 can send (101) to node 4, the node repairing node 4 can compute (001) and send it to node 1, the node repairing node 1 can send (001) to node 2.

The strategy is not unique. To repair node 1, download alternatively $\{(101), (010)\}$ from node 3 and node 5, to repair node 2, get $\{(110), (100)\}$ from node 0 and node 7, to repair node 4, get $\{(101), (100)\}$ from node 3 and node 0, then the node repairing node 4 can get (010) from node 1, the node repairing node 2 can get (111) from node 4, and the node repairing node 1 can get (001) from node 2.

6.2.4 Code Construction III: Spreads

In this section, we interpret a known family of storage codes [18] in the language of orbit codes, using the result from [17] that shows the connection between spreads and orbit codes.

Consider the finite fields $\mathbb{F}_q \subset \mathbb{F}_{q^m} \subset \mathbb{F}_{q^n}$, with $m|M$. It is well known that $\mathbb{F}_{q^M}$ is an M -dimensional vector space over $\mathbb{F}_q$, of which $\mathbb{F}_{q^m}$ and its cosets are m -dimensional subspaces. They thus form a spread, that is a collection of disjoint subspaces whose union is $\mathbb{F}_{q^M}$. Spreads to design storage codes have been investigated in [18], and can be obtained as a particular case of the proposed construction of orbit codes using companion matrices. For this, two ingredients are needed: (1) g must be the companion matrix of a primitive polynomial $p(x)$,

meaning that $p(x)$ is monic and irreducible of degree M , and of order $l = q^M - 1$, namely, roots of $p(x)$ are primitive elements of $\mathbb{F}_{q^M}$, they generate the cyclic group $\mathbb{F}_{q^M}^*$ and (2) $\mathcal{U}$ is chosen so as to contain a basis for $\mathbb{F}_{q^m}$.

Proposition 6.2.14. *Suppose that $m|M$, and consider the field extension $\mathbb{F}_{q^M}/\mathbb{F}_{q^m}$. Let U contain a basis of $\mathbb{F}_{q^m}$, and g be the companion matrix of a primitive polynomial $p(x)$ such that $p(\omega) = 0$ and $\langle \omega \rangle = \mathbb{F}_{q^m}^*$. Then $\mathcal{U}\langle g \rangle$ is an orbit code of length $\frac{q^M-1}{q^m-1}$ such that $\mathcal{U}g^i \cap \mathcal{U}g^j$ is trivial whenever $i \neq j$, and the union of $\mathcal{U}g^i$, $i = 1, \dots, \frac{M}{m}$ is $\mathbb{F}_{q^M}$.*

Proof. Note that $\mu := \omega^{M/m}$ is a generator of $\mathbb{F}_{q^m}^*$, and that $\{1, \mu, \dots, \mu^{m-1}\}$ forms an $\mathbb{F}_q$ -basis of $\mathbb{F}_{q^m}$. Form a matrix U where each row contains $\mu^i = \omega^{iM/m}$ written in an $\mathbb{F}_q$ -basis of $\mathbb{F}_{q^M}$, $i = 0, \dots, m-1$. Then the action of g corresponds to multiplication by ω , and the claims follow from the fact that as a $\mathbb{F}_{q^M}^*$ can be written as the union of $\omega^i \mathbb{F}_{q^m}^*$, $i = 1, \dots, \frac{q^M-1}{q^m-1}$. $\square$

More discussion about spread codes is also available in [17].

Example 6.2.15. Take $p(x) = x^4 + x + 1 \in \mathbb{F}_2[x]$, it is a primitive polynomial, and let g be its companion matrix

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

Let ω be a root of $p(x)$, that is $\omega^4 = \omega + 1$, also ω has order 15 and is a primitive

element of $\mathbb{F}_{16}$. Therefore ω^5 is a generator of $\mathbb{F}_4^*$, and an $\mathbb{F}_2$ -basis for $\mathbb{F}_4$ is

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

Then the orbit code $\mathcal{U}\langle g \rangle$ forms a spread, it has length 5, and we obtain a known storage code [18, Example 1], namely

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$

6.3 Instances of Cyclic Orbit Codes using Diagonal Matrices

We describe in this section cyclic orbit codes based on diagonal matrices, and obtain Reed-Solomon codes as a particular case, and a family of storage codes which will turn out to give optimal parameters.

Let ω be the primitive element of $\mathbb{F}_q$. Consider the cyclic group $G = \langle g \rangle$ of $GL_M(\mathbb{F}_q)$ generated by the $M \times M$ diagonal matrix

$$g = \begin{bmatrix} 1 & & & & \\ & w & & & \\ & & w^2 & & \\ & & & \ddots & \\ & & & & w^{M-1} \end{bmatrix} \quad (6.3.1)$$

where $M < |\mathbb{F}_q|$. Then the order of g is that of w , which is $|\mathbb{F}_q| - 1$. Indeed, for g^i to be the identity for some i , we at least need w^i to be 1, but then all other powers $(w^j)^i$ will be 1 as well, $j = 2, \dots, M - 1$.

6.3.1 Reed-Solomon Codes.

Consider the orbit code $\mathcal{U}G$ where $\mathcal{U}$ has for basis

$$U = [1, w, \dots, w^{M-1}].$$

The length N of the orbit code $\mathcal{U}G$ is

$$N = |\mathcal{U}G| = \frac{|G|}{|\text{Stab}_G(\mathcal{U})|} = |\mathbb{F}_q| - 1.$$

In fact, the orbit code $\mathcal{U}G$ is given by $\mathcal{U}G = \{Ug^i, i = 0, \dots, |\mathbb{F}_q| - 2\}$ where

$$\begin{aligned} Ug^i &= [1, w, \dots, w^{M-1}] \begin{bmatrix} 1 & & & & \\ & w^i & & & \\ & & w^{2i} & & \\ & & & \ddots & \\ & & & & w^{(M-1)i} \end{bmatrix} \\ &= [1, w^{i+1}, w^{2+2i}, \dots, w^{(M-1)+i(M-1)}]. \end{aligned}$$

This corresponds to a storage allocation where each node stores the inner product of the object $\mathbf{o}$ with a column of a generator matrix of a Reed-Solomon code.

6.3.2 A Code Construction for t Failures, $t \geq 1$.

A variation of the above construction can be obtained by tensor products. Consider the orbit code $\mathcal{U}G$ where $G = \langle g \otimes I_m \rangle$ for g the diagonal matrix given by

$$g = \begin{bmatrix} 1 & & & & \\ & w & & & \\ & & w^2 & & \\ & & & \ddots & \\ & & & & w^{k-1} \end{bmatrix},$$

$k \geq 1$, and I_m the m -dimensional identity matrix, and take for a basis of $\mathcal{U}$

$$U \otimes I_m = [1, w, \dots, w^{k-1}] \otimes I_m.$$

Note that the generator g chosen for this construction is similar to (6.3.1), but for the dimension $k \geq 1$. Since M is the size of the object, we choose k and m such that $km = M$. Since the order of $g \otimes I_m$ is still $|\mathbb{F}_q| - 1$, this does not change the size N of the orbit code $\mathcal{U}G$.

Proposition 6.3.1. *Let g be the k -dimensional diagonal matrix above, and let $U \otimes I_m = [1, w, \dots, w^{k-1}] \otimes I_m$ be a basis of the m -dimensional subspace $\mathcal{U} \in G_q(m, M)$, $M = km$. Then the orbit code $\mathcal{U}G$ has the following properties:*

1. *the object is retrieved by contacting any k nodes,*
2. *the repair of t failures, $t \geq 1$, is done by contacting any k nodes.*

Proof. 1. If any k nodes $i_1, \dots, i_k$ are contacted, then we obtain k columns of a Vandermonde matrix, each tensored by I_m , thus a $k \times k$ invertible submatrix

tensored by I_m , and the data object $\mathbf{o}$ can be recovered.

2. Without loss of generality, we label the t failed nodes from 1 to t . Then t live nodes each contact k nodes, say the i th node among those t live nodes connects to nodes $i_1, \dots, i_k$ and downloads only the one-dimensional subspace, corresponding to the i th row of $Ug^{i_j} \otimes I_m$, $j = 1, \dots, k$. Again using the fact that we obtain k columns of a Vandermonde matrix, the i th node computes a part of the object, given by the coefficients $\mathbf{o}_i, \mathbf{o}_{m+i}, \dots, \mathbf{o}_{(k-1)m+i}$, and after the download phase, it can compute the inner product of the vector $(\mathbf{o}_i, \mathbf{o}_{m+i}, \dots, \mathbf{o}_{(k-1)m+i})$ with the vector Ug^i , and thus obtains one of the m coefficients it needs to store. The $m - 1$ missing coefficients are obtained through (partial) collaboration as follows: the i th node contacts nodes $i + 1, \dots, i + m - 1$ (the indices are understood modulo the number of nodes). It already owns $(\mathbf{o}_i, \mathbf{o}_{m+i}, \dots, \mathbf{o}_{(k-1)m+i})(Ug^i)^T$, and gets $(\mathbf{o}_{i+1}, \mathbf{o}_{m+i+1}, \dots, \mathbf{o}_{(k-1)m+i+1})(Ug^{i+1})^T$ from node $i + 1$, and similarly from the other nodes, until $(\mathbf{o}_{i+m-1}, \mathbf{o}_{m+i+m-1}, \dots, \mathbf{o}_{(k-1)m+i+m-1})(Ug^{i+m-1})^T$ from node $i + m - 1$.

□

6.3.3 Comparison

Recall that we encode an object $\mathbf{o}$ of size M over $\mathbb{F}_q$ into Nm pieces and distribute them into N storage nodes, each of the nodes stores m coefficients in $\mathbb{F}_q$.

Let us consider the code construction of Subsection 6.2.2 for $t = 2$. An object $\mathbf{o}$ of length M is stored in N nodes, each of the nodes storing one codeword of the orbit code $\mathcal{U}_I G$, $\mathcal{U}_I \in G_2(M - 1, M)$ with canonical basis U_I , thus $\alpha = M - 1$. The

group G acting is $G = \langle g \rangle$ with g the companion matrix of the polynomial $p(x)$ whose order is l , so that $N = l$. When $p(x) = \sum_{i=0}^M x^i$, we know from Proposition 6.2.9 that $k = 2$, $\beta = M - 2$, and $\beta' = 1$. We notice that the total amount of data downloaded for repair is $2(M - 2) + 2 = 2M - 2$, which is $M - 1$ per node, suggesting that the code is at MBR. At MBR point, we should have, using $d = 1$, $k = 2$, $t - s = 1$:

$$\alpha = \gamma = \frac{3M}{4}, \beta = \frac{M}{2}, \beta' = \frac{M}{4}.$$

When $M = 4$, the code in Subsection 6.2.2 has

$$\alpha = \gamma = M - 1 = 3 = \frac{3M}{4}, \beta = M - 2 = 2 = \frac{M}{2}, \beta' = 1 = \frac{M}{4},$$

which is indeed at MBR point (α is minimized). When $M > 4$, the code has parameters:

$$\alpha = \gamma = M - 1 > \frac{3M}{4}, \beta = M - 2 > \frac{M}{2}, \beta' = 1 < \frac{M}{4},$$

which, as mentioned above, satisfies $\alpha = \gamma$, but α is not minimized.

Let us consider the partially collaborative code construction of Subsection 6.2.2 for $t \geq 3$. As discussed above for the collaborative code construction, this partially collaborative code construction only changes the value of t , but $t - s$ still equals to 1. We know then that the repair bandwidth is still $M - 1$, which suggests that the code is at MBR. Since the optimal values of α, γ at MBR point are related to the value of $t - s$ which is not changed, the minimized α is obtained by $M = 4$ (which indeed is a collaborative phase since three failures cannot happen when $M = 4$). For $M \geq 4$, still $\alpha = \gamma$, but α is not minimized, while the number

Figure 6.1: The storage code proposed in Subsection 6.2 is optimal with respect to the minimum repair bandwidth, but not with respect to the storage capacity α . This shows the optimal value of α with respect to the storage needed for the construction, as a function of M . When $M = 4$, our construction is optimal. We set $d = 1$, $k = 2$ and $t - s = 1$.

of contacted nodes stays optimal. The gap between the optimal storage and the value of α for this construction is shown on Figure 6.1, for different values of M between 4 and 10. We remark that optimal code constructions at MBR point for partial collaboration are not known as of now. This suggests that a deeper study of automorphism groups as a future work to get new/better codes for partial collaboration.

Then consider the code construction of Subsection 6.2.3. The object $\mathbf{o}$ of length M is stored in N nodes, each of the nodes storing one codeword of the orbit code $\mathcal{U}_I G$, $\mathcal{U}_I \in G_2(M - 1, M)$ with canonical basis U_I , thus $\alpha = M - 1$. The group G acting is $G = \langle g \rangle$ with g the companion matrix of the polynomial $p(x)$ whose order is $2^M - 1$, so that $N = 2^M - 1$. We know from Proposition 6.2.11 that $k = 2$, $d = 2$, $t - s = 2$, $\beta = M - 2$, and $\beta' = 1$. We notice that the total amount of data downloaded for repairing one node is $2(M - 2) + 2 = 2M - 2$,

which suggesting that the code is not at MBR. We consider whether the code is at MSR. At MSR point, we should have, using $d = 2$, $k = 2$, $t - s = 2$:

$$\alpha = \frac{M}{2}, \gamma = \frac{2M}{3}, \beta = \beta' = \frac{M}{6}.$$

So when $M \geq 3$, the code cannot be at MSR, and

$$\alpha = M - 1 > \frac{M}{2}, \gamma = 2M - 2 > \frac{2M}{3}, \beta = M - 2 > \frac{M}{6}, \beta' = 1$$

This code construction however provides an example of partial collaborative repair, a setting for which code constructions are very sparse, and the trade-off between storage and repair bandwidth not (yet) well understood beyond MSR and MBR points.

Consider finally the code given in Subsection 6.3.2. Set m to be $t - s + 1$, which sets the level of collaboration, namely there is collaboration among $t - s$ repair nodes. Then the size of the object is $M = k(t - s + 1)$, each node has a storage capacity of $\alpha = t - s + 1$, $t - s \geq 1$. This construction can be seen as a new interpretation of the construction of Section 5.2, which fits the minimum storage regime (MSR), since $\alpha = M/k$. The total repair bandwidth γ is

$$\gamma = k + (t - s), \quad 1 \leq s \leq t,$$

since every repair node downloads k amount of data, and exchanges $t - s$ of it.

Chapter 7

Secure File Size against a Passive Eavesdropper

In this chapter, we calculate two upper bounds on the secure file size (at MBR and MSR points respectively) for a passive eavesdropper model. From the bounds, we can see that partial collaboration brings security advantages with respect to full collaboration for distributed storage systems.

In [25], Pawar et al. considered a *passive eavesdropper* (the eavesdropper can only read the data but can not modify the data) model, where the eavesdropper can observe the data on nodes that have been repaired (the number l of nodes it observed is less than k). They computed a bound on the *secure file size* (the maximum amount of data that can be stored safely) assuming that regenerating codes. In [26], Shah et al. gave a bound on the secure file size considering a passive eavesdropper model that generalizes the model in [25], in that the eavesdropper can access not only the nodes that have been repaired but also the nodes being repaired. This makes a difference, since more data may be in transit in

nodes being repaired than that will actually be stored. Rawat et al. [27] tighten the bound given in [26], Koyluoglu et al. [29] extend the bound in [27] to the scheme for (full) collaboration. Using this model, we can evaluate whether the partial collaboration reduces the harm to security caused by collaboration. Now we generalize the bounds to partial collaboration. In the following context, the repair process is of partial collaboration.

Suppose that a data object $\mathbf{o}$ of size M has to be stored over n nodes, across a networked distributed storage system. We assume the alphabet to be the finite field $\mathbb{F}_q$ of cardinality q , q a prime power, that is $\mathbf{o} \in \mathbb{F}_q^M$. It is encoded into n pieces $\mathbf{s}_1, \dots, \mathbf{s}_n$, each piece is stored at a distinct node, we label the nodes such that node i stores $\mathbf{s}_i$, $i = 1, \dots, n$. We label the nodes participating to the repair process by $n + j$, $j \geq 1$.

In the presence of an eavesdropper, data objects are encrypted for ensuring their confidentiality. We therefore distinguish the secure file $\mathbf{o}^s$, of size $\mathcal{M}^s$, from its encrypted version, the file $\mathbf{o}$ of size $\mathcal{M}$. Encoding is then done on top of the encrypted file $\mathbf{o}$.

7.1 The Case of Minimum Repair Bandwidth

Consider an (n, k) storage code made to minimize the repair bandwidth γ , that is data objects are encoded, and the encoded pieces are stored across n nodes, such that any k of them are enough to recover the data object. The repair bandwidth per node must at least be the content of the storage node α , so the minimum repair bandwidth is obtained when $\gamma = \alpha = d\beta + (t - s)\beta'$.

Now an eavesdropper may spy on the content of some nodes, but also on the

incoming links of the nodes. However for a storage code that minimizes the repair bandwidth, the contents of the node and that of the incoming links are the same.

Definition 7.1.1. Let $\mathbf{o}^s$ be a file of size $\mathcal{M}^s$, which is encrypted into a secure file $\mathbf{o}$, of size $\mathcal{M}$. A networked distributed storage system is said to *achieve a secure file size of $\mathcal{M}^s$ against an l_1 -eavesdropper*, $l_1 \leq k$, if, for any set ε_1 of size l_1

$$I(\mathbf{o}^s; \mathbf{e}) = 0,$$

where $\mathbf{e} = \{\mathbf{s}_i, i \in \varepsilon_1\}$ is the eavesdropper observation vector.

7.1.1 A Min-Cut Bound

To understand the effect of the presence of an eavesdropper, we use known techniques, in particular a max-flow min-cut bound in the storage network, and a standard upper bound on the leaked information [26, 28, 29], as detailed next.

Note that by definition of mutual information

$$I(\mathbf{o}^s; \mathbf{e}) = 0 \iff H(\mathbf{o}^s | \mathbf{e}) = H(\mathbf{o}^s).$$

Suppose that over time, k nodes have failed and been repaired at nodes $n + 1, \dots, n + k$ by g consecutive groups, each of size u_j , $j = 0, \dots, g - 1$, that is $\sum_{j=0}^{g-1} u_j = k$, and $1 \leq u_j \leq t$. Suppose that $\varepsilon_1 \subseteq \{n + 1, \dots, n + k\}$, then when a

data collector contacts these k nodes, we have

$$\begin{aligned} H(\mathbf{o}^s|\mathbf{e}) &= H(\mathbf{o}^s|\mathbf{s}_i, i \in \varepsilon_1) \\ &= H(\mathbf{o}^s|\mathbf{s}_i, i \in \varepsilon_1) - H(\mathbf{o}^s|\mathbf{s}_j, j = n+1, \dots, n+k) \end{aligned} \quad (7.1.1)$$

$$= I(\mathbf{o}^s; \mathbf{s}_j, j \in \{n+1, \dots, n+k\} \setminus \varepsilon_1 | \mathbf{s}_i, i \in \varepsilon_1) \quad (7.1.2)$$

$$\leq H(\mathbf{s}_j, j \in \{n+1, \dots, n+k\} \setminus \varepsilon_1 | \mathbf{s}_i, i \in \varepsilon_1) \quad (7.1.3)$$

$$= \sum_{j \in \{n+1, \dots, n+k\} \setminus \varepsilon_1} H(\mathbf{s}_j | \mathbf{s}_{n+1}, \dots, \mathbf{s}_{j-1}, \mathbf{s}_i, i \in \varepsilon_1) \quad (7.1.4)$$

Indeed, (7.1.1) holds because $H(\mathbf{o}^s|\mathbf{s}_j, j = n+1, \dots, n+k) = 0$ since the knowledge of any k encoded pieces is enough to recover $\mathbf{o}$. Then (7.1.2) comes from the definition of conditional mutual information $I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$. Alternatively, one may develop the conditional mutual information as $I(X;Y|Z) = H(X,Z) - H(Z) - H(X,Y,Z) + H(Y,Z) = H(X,Z) - H(X,Y,Z) + H(Y|Z) \leq H(Y|Z)$, which justifies (7.1.3). Finally, (7.1.4) holds because $H(X_1, \dots, X_n|Y) = \sum_{i=1}^n H(X_i|X_1, \dots, X_{i-1}, Y)$. The corresponding definitions and properties have been stated in Chapter 2.

Proposition 7.1.2. *For an (n, k) partially collaborative storage code at the minimum repair bandwidth point (MBR), we have that the secure file size $\mathcal{M}^s$ is upper bounded by*

$$\mathcal{M}^s \leq (k - l_1)(2d + t - k - l_1 - s + 1) \frac{\beta}{2} \quad (7.1.5)$$

in the presence of an l_1 -eavesdropper spying the stored content of l_1 nodes with $0 \leq l_1 \leq k$. We further have $\beta = 2\beta'$ and the optimal file size is reached with

equality in (7.1.5) and

$$\beta = \frac{\mathcal{M}^s}{k - l_1} \frac{2}{2d + t - s + 1 - k - l_1}, \quad \alpha = \frac{\mathcal{M}^s}{k - l_1} \frac{2d + t - s}{2d + t - s + 1 - k - l_1}.$$

Proof. We want $H(\mathbf{o}^s|\mathbf{e}) = H(\mathbf{o}^s)$, and we have the upper bound

$$H(\mathbf{o}^s) \leq \sum_{j \in \{n+1, \dots, n+k\} \setminus \varepsilon_1} H(\mathbf{s}_j | \mathbf{s}_{n+1}, \dots, \mathbf{s}_{j-1}, \mathbf{s}_i, \quad i \in \varepsilon_1)$$

from (7.1.4). To upper bound $H(\mathbf{s}_j | \mathbf{s}_{n+1}, \dots, \mathbf{s}_{j-1}, \mathbf{s}_i, \quad i \in \varepsilon_1)$, as is standardly done [4, 6, 13, 29], we consider a cut in the information flow graph that represents the flow of information from a data owner which uploads the data, to the data collector which gathers it, via different repair phases. The cut involves the k nodes $n + j, j = 1, \dots, k$, that have been repaired, among which $l_1 = |\varepsilon_1|$ are observed by the eavesdropper. The k repaired nodes are grouped into g consecutive groups of size $u_i, i = 0, \dots, g - 1$. Each group of u_i nodes is partitioned into four groups:

- $l_1^{i,1}$ nodes have their content eavesdropped,
- $m_i - l_1^{i,1}$ nodes have their content that contribute $(m_i - l_1^{i,1})\alpha$ to the cut,
- $l_1^{i,2}$ nodes have their outgoing edges eavesdropped,
- $u_i - m_i - l_1^{i,2}$ nodes have their outgoing edges that contribute $(d - \sum_{j=0}^{i-1} u_j)\beta$ from the download phase, and $(t - s + 1 - u_i)\beta'$ from the collaborative phase (when $t - s + 1 \geq u_i$) to the cut,

with $l_1^i = l_1^{i,1} + l_1^{i,2}$, and $\sum_{i=0}^{g-1} (l_1^{i,1} + l_1^{i,2}) = \sum_{i=0}^{g-1} l_1^i = l_1$.

Set $J = \{i, t - s + 1 - u_i \geq 0\}$ and $\bar{J} = \{i, t - s + 1 - u_i < 0\}$ then

$$\begin{aligned}
H(\mathbf{o}^s) &\leq \sum_{j \in R} H(\mathbf{s}_j | \mathbf{s}_{n+1}, \dots, \mathbf{s}_{j-1}, \mathbf{s}_i, i \in \varepsilon_1) \\
&\leq \sum_{i \in J} \{(m_i - l_1^{i,1})\alpha + (u_i - m_i - l_1^{i,2})[(d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta']\} \\
&\quad + \sum_{i \in \bar{J}} \{(m_i - l_1^{i,1})\alpha + (u_i - m_i - l_1^{i,2})(d - \sum_{j=0}^{i-1} u_j)\beta\} \\
&\leq \sum_{i \in J} (u_i - l_1^i) \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta'\} \\
&\quad + \sum_{i \in \bar{J}} (u_i - l_1^i) \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta\}
\end{aligned}$$

using Jensen's inequality. Since at MBR point $\alpha = \gamma = d\beta + (t - s)\beta'$

$$H(\mathbf{o}^s) \leq \sum_{i \in J} (u_i - l_1^i) [(d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta'] + \sum_{i \in \bar{J}} (u_i - l_1^i) (d - \sum_{j=0}^{i-1} u_j)\beta. \quad (7.1.6)$$

To evaluate β , we note from (7.1.6) that the smallest contribution in β' occur when $u_i = t - s + 1$, $t - s + 1 | k$, in which case (7.1.6) becomes

$$\begin{aligned}
H(\mathbf{o}^s) &\leq \sum_{i=0}^{g-1} (u_i - l_1^i) (d - \sum_{j=0}^{i-1} u_j) \beta \\
&= [d(k - l_1) - (t - s + 1) \sum_{i=0}^{g-1} i(u_i - l_1^i)] \beta \\
&= [d(k - l_1) - \frac{k(k - t + s - 1)}{2} + (t - s + 1) \sum_{i=0}^{g-1} i l_1^i] \beta
\end{aligned}$$

since $\sum_{i=0}^{g-1} u_i = \sum_{i=0}^{g-1} (t - s + 1) = k$ means that $g = \frac{k}{t-s+1}$. The minimum

is obtained for $t - s + 1 | l_1$ and for the values of l_1^i such that $l_1^i = u_i$ for $i = 0, \dots, \frac{l_1}{t-s+1} - 1$, and $l_1^i = 0$ for $i \geq \frac{l_1}{t-s+1}$. We then get

$$\begin{aligned}
H(\mathbf{o}^s) &\leq [d(k - l_1) - \frac{k(k - t + s - 1)}{2} + (t - s + 1)^2 \sum_{i=0}^{\frac{l_1}{t-s+1}-1} i] \beta \\
&= [d(k - l_1) - \frac{k(k - t + s - 1)}{2} + \frac{l_1(l_1 - (t - s + 1))}{2}] \beta \\
&= [d(k - l_1) + \frac{(t - s + 1)(k - l_1) - (k - l_1)(k + l_1)}{2}] \beta \\
&= \frac{(k - l_1)}{2} [2d + t - s + 1 - k - l_1] \beta.
\end{aligned}$$

Conversely, the smallest contribution in β happens when $u_i = 1$, thus combining (7.1.6) with the minimal value of β , namely

$$\beta = \frac{H(\mathbf{o}^s)}{k - l_1} \frac{2}{2d + t - s + 1 - k - l_1}$$

yields

$$H(\mathbf{o}^s) \leq \frac{(2d + 1 - k - l_1)H(\mathbf{o}^s)}{2d + t - s + 1 - k - l_1} + (k - l_1)(t - s)\beta'$$

that is

$$\beta' \geq \frac{H(\mathbf{o}^s)}{k - l_1} \frac{1}{2d + t - s + 1 - k - l_1}$$

and the minimal values of β and β' satisfy $\beta = 2\beta'$.

When $g = k$, $u_i = 1$, $i = 0, \dots, g - 1$, and we have

$$H(\mathbf{o}^s) \leq \frac{(k - l_1)(2d - k - l_1 + 1)\beta}{2} + (k - l_1)(t - s)\beta'.$$

When $g = 1$, $t \geq k$, then $u_0 = k$, $l_1^0 = l_1$, if $t \leq k + s - 1$, the bound will be

$$H(\mathbf{o}^s) \leq (k - l_1)[d\beta + (t - s + 1 - k)\beta'];$$

if $t < k + s - 1$, the bound will be

$$H(\mathbf{o}^s) \leq (k - l_1)d\beta.$$

When $1 < g < k$, since we want minimize the bound, we need to minimize the size of J , or alternatively to maximize u_i . Let $t < k$, $u_i = t, i = 0, \dots, g - 2$, $g = \lfloor k/t \rfloor + 1$, then $u_{g-1} = k - \lfloor k/t \rfloor t$, we get the bound

$$\begin{aligned} H(\mathbf{o}^s) \leq \min_{l_1^i \leq t} \{ & \sum_{i=0}^{g-2} (t - l_1^i)(d - it)\beta \\ & + (k - \lfloor k/t \rfloor t - l_1^{g-1})[(d - \lfloor k/t \rfloor t)\beta + (t - s + 1 - k + \lfloor k/t \rfloor t)\beta'] \}. \end{aligned}$$

The tightest bound is obtained when $g = k$, $u_i = 1, i = 0, \dots, g - 1$.

The relation $\beta = 2\beta'$ in (7.1.6) gives the bound (7.1.5).

Recalling that $\alpha = \gamma = d\beta + (t - s)\beta'$ finally gives

$$\alpha = 2d\beta' + (t - s)\beta' = \beta'(2d + t - s).$$

□

The upper bound on the secure file size (7.1.5) of the above proposition is illustrated in Figure 7.1, as a function of s , $1 \leq s \leq 5$, for fixed l_1 , $d = k = t = 5$,

Figure 7.1: The secure file size $\mathcal{M}^s$ as a function of $1 \leq s \leq 5$ with $d = k = t = 5$, the lines from top to bottom represent the eavesdropper number l_1 from 1 to 4 respectively.

and $\beta = 2$:

$$\mathcal{M}^s \leq (5 - l_1)(10 - l_1 - s + 1). \quad (7.1.7)$$

The upper bound is monotonously decreasing, with extremes on the boundary points, that is maximum when $s = 1$ and minimum when $s = t = 5$. This shows that no collaboration ($s = t$) has the smallest secure file size, see Figure 7.1. Increasing the number of eavesdropper decreases the secure file size.

We recover the following particular cases:

Corollary 7.1.3. [29] *For a fully collaborative storage code operating at the MBR point, the secure file size $\mathcal{M}^s$ is upper bounded by*

$$\mathcal{M}^s \leq (k - l_1)(2d + t - k - l_1) \frac{\beta}{2}.$$

Proof. Full collaboration corresponds to $s = 1$. Thus set $s = 1$ into $\mathcal{M}^s \leq (k - l_1)(2d + t - k - l_1 - s + 1)\frac{\beta}{2}$. $\square$

Corollary 7.1.4. *For a storage code with no collaboration operating at the MBR point, the secure file size $\mathcal{M}^s$ is upper bounded by*

$$\mathcal{M}^s \leq (k - l_1)(2d - k - l_1 + 1)\frac{\beta}{2}.$$

Proof. No collaboration corresponds to $s = t$. Then set $s = t$ into $\mathcal{M}^s \leq (k - l_1)(2d + t - k - l_1 - s + 1)\frac{\beta}{2}$. $\square$

Corollary 7.1.5. *For a partially collaborative code with no eavesdropper at the minimum bandwidth repair (MBR) point, we have*

$$\alpha = \frac{\mathcal{M}}{k} \frac{2d + t - s}{2d - k + t - s + 1}, \beta = \frac{\mathcal{M}}{k} \frac{2}{2d - k + t - s + 1}, \beta' = \frac{\mathcal{M}}{k} \frac{1}{2d - k + t - s + 1}$$

for a total repair bandwidth of

$$\gamma = \frac{\mathcal{M}}{k} \frac{2d + t - s}{2d + t - k - s + 1}.$$

Proof. From the above proposition, we have $\beta = 2\beta'$ and

$$\beta = \frac{\mathcal{M}^s}{k - l_1} \frac{2}{2d + t - s + 1 - k - l_1}, \alpha = \frac{\mathcal{M}^s}{k - l_1} \frac{2d + t - s}{2d + t - s + 1 - k - l_1}.$$

Just set $l_1 = 0$. $\square$

Observation 7.1.1. *It is also possible to use the result of Section 4.2 to claim that $\beta = 2\beta'$ at MBR, in which case the proof of the above proposition can be*

simplified by using $\beta = 2\beta'$ inside. Then Corollary 7.1.5 is not a corollary any more, rather, the proposition becomes a corollary of the results from Section 4.2. We preferred the more general approach which gives the result of Section 4.2 at MBR as a corollary of the present analysis.

7.1.2 A Bandwidth Analysis

To understand the effect of the different parameters on the repair bandwidth, we consider the normalized repair bandwidth $\frac{\gamma}{\mathcal{M}^s}$, introduced in [29]. Since we have computed upper bounds on $\mathcal{M}^s$, we will be able to provide a lower bound $\bar{\gamma}$ on the normalized repair bandwidth:

$$\frac{\gamma}{\mathcal{M}^s} \geq \bar{\gamma}(s, t) = \frac{2d + t - s}{(k - l_1)(2d + t - s - k - l_1 + 1)}.$$

Note that the regime of no collaboration holds only for $d \geq k$, and that $d \leq n - t$ (once t nodes have failed, only $n - t$ nodes are still live and at most all of them can be contacted). We thus have $k \leq d \leq n - t$.

A first observation is that $\bar{\gamma}(s, t)$ is always greater when $l_1 > 0$ compared to $l_1 = 0$, that is having an eavesdropper does increase $\bar{\gamma}(s, t)$.

Suppose first that all live nodes participate to the repair process, that is $d = n - t$:

$$\frac{\gamma}{\mathcal{M}^s} \geq \bar{\gamma}(s, t) = \frac{(2n - t - s)}{(k - l_1)(2n - t - s - k - l_1 + 1)}.$$

The two extreme cases for s are no collaboration ($t = 1$ and $s = t$), for which

$$\bar{\gamma}(s = t = 1) = \frac{1}{k - l_1} \frac{2n - 2}{2n - k - l_1 - 1},$$

and full collaboration ($t = t_0 \geq 2$ and $s = 1$):

$$\bar{\gamma}(s = 1, t = t_0) = \frac{1}{k - l_1} \frac{2n - t_0 - 1}{2n - k - l_1 - t_0}.$$

We have

$$\begin{aligned} & \bar{\gamma}(s = t = 1) \leq \bar{\gamma}(s, t) \\ \iff & (2n - k - l_1 - t - s + 1)(2n - 2) \leq (2n - t - s)(2n - k - l_1 - 1) \\ \iff & (t + s - 2)(k + l_1 - 1) \geq 0 \end{aligned}$$

which always holds for $t \geq 2$, corresponding to the case where collaboration does not reduce the normalized repair bandwidth with respect to no collaboration, no matter the degree of collaboration $1 \leq s \leq t$. If we compare instead partial collaboration and full collaboration, then

$$\bar{\gamma}(s, t) \leq \bar{\gamma}(s = 1, t = t_0) \iff t_0 \geq t + s - 1.$$

The interpretation is that the best strategy is to repair one failure at a time. In case of correlated failures, the system is forced to repair several failures in parallel. If there is a coding strategy that allows full collaboration for a small threshold t_0 which can take care of the correlated failures, then this will give the least normalized repair bandwidth (if $t_0 = t$, then $t_0 \geq t + s - 1$ exactly when $s = 1$

which is full collaboration). If however there exists a partial collaboration for a choice of s, t such that $t_0 \geq t + s - 1$, which can happen when the full collaborative strategy exists for a large threshold t_0 , then partial collaboration is better. This holds irrespectively of the value of l_1 .

Consider next the case $n > d + t \iff d < n - t$. Comparing no collaboration with partial collaboration (and in particular full collaboration), the reverse conclusion is obtained in that partial collaboration reduces the normalized repair bandwidth with respect to no collaboration:

$$\bar{\gamma}(s = t = 1) = \frac{1}{k - l_1} \frac{2d}{2d - k - l_1 + 1} > \bar{\gamma}(s, t) = \frac{1}{k - l_1} \frac{2d + t - s}{2d - k - l_1 + t - s + 1}$$

for $t - s \geq 1$. Next comparing partial collaboration and full collaboration, we have

$$\bar{\gamma}(s, t) \leq \bar{\gamma}(s = 1, t = t_0) \iff t - s + 1 \geq t_0.$$

The interpretation in this case is that it is best to delay to repair until several failures happen. If there is a coding strategy that allows full collaboration for a large threshold t_0 , then this will give the least normalized repair bandwidth (if $t_0 = t$, then $t - s \geq t_0 - 1$ exactly when $s = 1$ which is full collaboration). If however there exists a partial collaboration for a choice of s, t such that $t - s + 1 \geq t_0$, which can happen when the full collaborative strategy exists for a small threshold t_0 , then it is best to delay further the repair, and partial collaboration is better in terms of normalized bandwidth. This holds irrespectively of the value of l_1 .

7.2 The Case of Minimum Storage

Consider a storage code that minimizes the storage capacity α . The storage capacity must be at least $\mathcal{M}/k$, since the object $\mathbf{o}$ has size $\mathcal{M}$, and any k nodes allow the retrieval of $\mathbf{o}$. We thus have $\alpha = \mathcal{M}/k$.

Now an eavesdropper may spy on the content of some nodes, but also on the incoming links of the nodes. It is important to distinguish both cases, because the amount of information that is transmitted to a node during the repair process could be more than what is actually stored, especially at the minimum storage capacity $\alpha = \mathcal{M}/k$. In fact, it was observed in [27], in the case of $t = 1$ failure, that at MSR point, the node that performs the repair has actually access to more data ($\mathcal{M}/k(d-k+1)$) than what it can store ($\mathcal{M}/k$). This is not the case at MBR point, as seen in the above computations. This has consequences in the presence of an eavesdropper, who can access the data that was downloaded during the repair: the eavesdropper gains more information.

The same is true in the partially collaboration setting. We know from Section 4.2 that every node in the set of t nodes that participate in the repair of t failures at MSR point access to more data ($\mathcal{M}(d+t-s)/k(d-k+t-s+1)$) than what it can store ($\mathcal{M}/k$), whenever $k \geq 2$.

Suppose that we use a linear code to encode the data file $\mathbf{o}$, so that $\mathbf{s}_i \in \mathbb{F}_q^\alpha$ may then be written as

$$\mathbf{s}_i = (\mathbf{o}^T \mathbf{g}_i^1, \dots, \mathbf{o}^T \mathbf{g}_i^\alpha)$$

for some column vectors $\mathbf{g}_i^1, \dots, \mathbf{g}_i^\alpha$.

Let $\mathcal{S}_i$ be the subspace spanned by the vectors $\{\mathbf{g}_i^1, \dots, \mathbf{g}_i^\alpha\}$, $i = 1, \dots, n$, so that the content of each node may be seen as a vector space of dimension α . Let

$\mathbf{d}_{i,j} \in \mathbb{F}_q^\beta$ be the downloaded data from node i to node j , which may be written as

$$\mathbf{d}_{i,j} = (\mathbf{o}^T \mathbf{g}_{i,j}^1, \dots, \mathbf{o}^T \mathbf{g}_{i,j}^\beta),$$

where $\mathbf{g}_{i,j}^r \in \mathcal{S}_i$, $r = 1, \dots, \beta$, and we refer to $\mathcal{D}_{i,j}$ as the subspace spanned by the column vectors $\{\mathbf{g}_{i,j}^1, \dots, \mathbf{g}_{i,j}^\beta\}$.

We collect all the downloaded data to the node j in $\mathbf{d}_j \in \mathbb{F}_q^{d\beta}$. Similarly, let $\mathbf{d}_{A,B}$ be the set of all the downloaded data from a set of nodes indexed by A to a set of nodes indexed B , and let $\mathcal{D}_{A,B}$ be the subspace spanned by the vectors $\mathbf{g}_{i,j}$, $i \in A, j \in B$.

Let $\mathbf{d}'_j \in \mathbb{F}_q^{\beta'(t-s)}$ be all the data obtained during the exchange phase at node j , and $\mathbf{d}'_{A,B}$ be all the data exchanged from the set of nodes indexed by A to the set of nodes indexed by B .

Definition 7.2.1. Let $\mathbf{o}^s$ be a secret file of size $\mathcal{M}^s$, which is encrypted into a secure file $\mathbf{o}$, of size $\mathcal{M}$. A networked distributed storage system is said to *achieve a secure file size of $\mathcal{M}^s$ against an (l_1, l_2) eavesdropper*, if, for any disjoint sets ε_1 and ε_2 of size l_1 and l_2 , respectively, $|l_1 \cup l_2| \leq k$,

$$I(\mathbf{o}^s; \mathbf{e}) = 0,$$

where $\mathbf{e} = \{\mathbf{s}_i, i \in \varepsilon_1, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2\}$ is the eavesdropper observation vector.

7.2.1 A Min-Cut Bound

Suppose that over time, k nodes have failed and been repaired at nodes $n + 1, \dots, n + k$ by g consecutive groups, each of size $u_j, j = 0, \dots, g - 1$, that is

$\sum_{j=0}^{g-1} u_j = k$, and $1 \leq u_j \leq t$. Suppose that $(\varepsilon_1 \cup \varepsilon_2) \subseteq \{n+1, \dots, n+k\}$, let $R = \{n+1, \dots, n+k\} \setminus (\varepsilon_1 \cup \varepsilon_2)$, then when a data collector contacts these k nodes, we have

$$\begin{aligned}
H(\mathbf{o}^s | \mathbf{e}) &= H(\mathbf{o}^s | \mathbf{s}_i, i \in \varepsilon_1, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2) \\
&= H(\mathbf{o}^s | \mathbf{s}_i, i \in \varepsilon_1, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2) - H(\mathbf{o}^s | \mathbf{s}_i, i \in \varepsilon_1, \mathbf{s}_R, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2) \\
&= I(\mathbf{o}^s; \mathbf{s}_R | \mathbf{s}_i, i \in \varepsilon_1, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2) \\
&\leq H(\mathbf{s}_R | \mathbf{s}_i, i \in \varepsilon_1, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2) \\
&= \sum_{j \in R} H(\mathbf{s}_j | \mathbf{s}_{n+1}, \dots, \mathbf{s}_{j-1}, \mathbf{s}_i, i \in \varepsilon_1, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2)
\end{aligned}$$

using the same arguments as in the previous section, with the slight variation that $H(\mathbf{o}^s | \mathbf{s}_i, i \in \varepsilon_1, \mathbf{s}_R, \mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2) = 0$ because the knowledge of $\mathbf{d}_j, \mathbf{d}'_j, j \in \varepsilon_2$ gives that of $\mathbf{s}_j, j \in \varepsilon_2$.

Proposition 7.2.2. *For an (n, k) partially collaborative storage code at the minimum storage repair point (MSR), we have that the secure file size $\mathcal{M}^s$ is upper bounded by*

$$\mathcal{M}^s \leq \sum_{i=1}^{k-l_1-l_2} (\alpha - I(\mathbf{s}_i; \mathbf{d}_{i, \varepsilon_2})) \quad (7.2.1)$$

If $I(\mathbf{s}_i; \mathbf{d}_{i, \varepsilon_2}) \geq \beta' = \beta$, then (7.2.1) becomes

$$\mathcal{M}^s \leq (k - l_1 - l_2)(\alpha - \beta), \quad (7.2.2)$$

in the presence of an (l_1, l_2) eavesdropper spying the stored content of l_1 nodes and the downloaded and exchanged data of l_2 nodes, where $l_1 + l_2 \leq k$. Furthermore

$\beta = \beta'$ and the optimal secure file size is reached with equality in (7.2.2), with

$$\beta = \beta' = \frac{\alpha}{d - k + t - s + 1}.$$

Proof. Recall that nodes in ε_1 are eavesdropped on what they store only, while nodes in ε_2 are eavesdropped not only on what they store, but also on what they download and exchange. Each group of u_i nodes is partitioned into the following two groups:

- $l_1^{i,1}, l_2^{i,1}$ nodes are having what they store eavesdropped, among which, $m_i - l_1^{i,1} - l_2^{i,1}$ of them belong to R , which are denoted by h_i . They may leak information to the nodes in ε_2 , so they contribute $(m_i - l_1^{i,1} - l_2^{i,1})\alpha - \dim \mathcal{D}_{h_i; \varepsilon_2}$ to the cut.
- $l_1^{i,2}, l_2^{i,2}$ nodes are being eavesdropped during the download phase, among which, $u_i - m_i - l_1^{i,2} - l_2^{i,2}$ of them are in R , contributing either $(d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta'$ or $(d - \sum_{j=0}^{i-1} u_j)\beta$ to the cut,

with $l_1^i = l_1^{i,1} + l_1^{i,2}$, $l_2^i = l_2^{i,1} + l_2^{i,2}$ and $\sum_{i=0}^{g-1} l_1^i = l_1$, $\sum_{i=0}^{g-1} l_2^i = l_2$.

By Jensen's inequality, we have

$$\begin{aligned} \mathcal{M}^s \leq & \sum_{i \in J} (u_i - l_1^i - l_2^i) \min \left\{ \alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i}, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - s + 1 - u_i)\beta' \right\} \\ & + \sum_{i \in \bar{J}} (u_i - l_1^i - l_2^i) \min \left\{ \alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i}, (d - \sum_{j=0}^{i-1} u_j)\beta \right\}. \end{aligned} \quad (7.2.3)$$

The highest contribution from β comes from no contribution in β' , that is when $u_i = t - s + 1$, for $i = 0, \dots, k/(t - s + 1) - 1$. Let $u_i = t - s + 1$, for $i = 0, \dots, \lfloor k/(t - s + 1) \rfloor - 1$, and $u_i = k - \lfloor k/(t - s + 1) \rfloor (t - s + 1)$, for

$i = \lfloor k/(t-s+1) \rfloor$. Then $g = \lfloor k/(t-s+1) \rfloor + 1$, $\bar{J}$ is empty, and (7.2.3) becomes

$$\begin{aligned} \mathcal{M}^s &\leq (u_{g-1} - l_1^{g-1} - l_2^{g-1}) \min\left\{\alpha - \frac{\dim \mathcal{D}_{h_{g-1}, \varepsilon_2}}{u_{g-1} - l_1^{g-1} - l_2^{g-1}}, \right. \\ &\quad \left. (d - \sum_{j=0}^{g-2} u_j)\beta + (t-s+1 - u_{g-1})\beta'\right\} \\ &\quad + \sum_{i=0}^{g-2} (u_i - l_1^i - l_2^i) \min\left\{\alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i}, (d - i(t-s+1))\beta\right\}. \end{aligned}$$

Each term in the sum is at most

$$(u_i - l_1^i - l_2^i)\left(\alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i}\right) = (u_i - l_1^i - l_2^i)\alpha - \dim \mathcal{D}_{h_i, \varepsilon_2},$$

and equality can be obtained only when

$$(k - l_1 - l_2)\alpha - \dim \mathcal{D}_{R, \varepsilon_2} = \mathcal{M}^s.$$

Thus each term which contains β should be no less than $\alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i}$.

For $i = 0, \dots, g-2$,

$$(d - i(t-s+1))\beta \geq \alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i},$$

and since $\dim \mathcal{D}_{h_i, \varepsilon_2} \geq 0$,

$$\beta \geq \frac{\alpha}{d - i(t-s+1)}.$$

When $i = g-2 = \lfloor k/(t-s+1) \rfloor - 1$,

$$\beta = \frac{\alpha}{d + t - s + 1 - \lfloor k/(t-s+1) \rfloor (t-s+1)},$$

and since $\lfloor k/(t-s+1) \rfloor (t-s+1) \leq k$, we have that the minimum value of β is

$$\beta = \frac{\alpha}{d-k+t-s+1}. \quad (7.2.4)$$

For $i = g-1$,

$$(d - (g-1)(t-s+1))\beta + (t-s+1-u_{g-1})\beta' \geq \alpha - \frac{\dim \mathcal{D}_{h_{g-1}, \varepsilon_2}}{u_{g-1} - l_1^{g-1} - l_2^{g-1}},$$

then taking into account that $\dim \mathcal{D}_{h_{g-1}, \varepsilon_2} \geq 0$ and (7.2.4),

$$\begin{aligned} \beta' &\geq \frac{\alpha - (d - (g-1)(t-s+1))\beta}{t-s+1-u_{g-1}} \\ &= \frac{\alpha(t-s+1-k+u_{g-1})}{(d-k+t-s+1)(t-s+1-u_{g-1})} \\ &= \frac{\alpha}{d-k+t-s+1}. \end{aligned} \quad (7.2.5)$$

Conversely, the highest contribution from β' comes when $u_i = 1$ for all $i = 0, \dots, g-1$, so that $g = k$ and $i = 0, \dots, k-1$, then (7.2.3) becomes

$$\mathcal{M}^s \leq \sum_{i=0}^{k-1} (1 - l_1^i - l_2^i) \min \left\{ \alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{u_i - l_1^i - l_2^i}, (d-i)\beta + (t-s)\beta' \right\}.$$

Here we still have that each term is at most

$$(1 - l_1^i - l_2^i) \left(\alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{1 - l_1^i - l_2^i} \right),$$

and equality is actually obtained when each term is equal to it.

Thus, each term that contains β and β' should be no less than $\alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{1-l_1^i-l_2^i}$,

that is for $i = 0, \dots, k-1$,

$$(d-i)\beta + (t-s)\beta' \geq \alpha - \frac{\dim \mathcal{D}_{h_i, \varepsilon_2}}{1 - l_1^i - l_2^i}.$$

From (7.2.4) and $\dim \mathcal{D}_{h_i, \varepsilon_2} \geq 0$, we get

$$\begin{aligned} \beta' &\geq \frac{\alpha - (d-i)\beta}{t-s} \\ &= \frac{\alpha(t-s+1-k+i)}{(d-k+t-s+1)(t-s)}, \end{aligned}$$

so take $i = k-1$, and

$$\beta' \geq \frac{\alpha}{d-k+t-s+1}. \quad (7.2.6)$$

Combining (7.2.5) and (7.2.6), we get the minimum value of β' is that

$$\beta' = \frac{\alpha}{d-k+t-s+1},$$

considering the minimum value β from (7.2.4), we get $\beta = \beta'$.

When $g = k$, $u_i = 1, i = 1, \dots, g-1$, we thus the the upper bound

$$\mathcal{M}^s \leq \sum_{i=0}^{g-1} (1 - l_1^i - l_2^i) \left(\alpha - \frac{\dim \mathcal{D}_{n+i+1, \varepsilon_2}}{1 - l_1^i - l_2^i} \right).$$

Suppose R is composed of the set of nodes $\{n+j, j = 1, \dots, k-l_1-l_2\}$, then we have

$$\mathcal{M}^s \leq \sum_{i=0}^{k-l_1-l_2-1} (\alpha - I(\mathbf{s}_i; d_{i, \varepsilon_2}))$$

thus, we get the bound (7.2.1).

At the minimum storage point, we can bound $\dim \mathcal{D}_{n+i+1, \varepsilon_2} \geq \beta' = \beta$, then

the above bound becomes that of [29], that is

$$\mathcal{M}^s \leq (k - l_1^i - l_2^i)(\alpha - \beta).$$

When $g = 1$, $t \geq k$, then $u_0 = k$, $l_1^0 = l_1$ and $l_2^0 = l_2$. If $t - s + 1 < k$, then $J = \emptyset$ and $\alpha < d\beta$, the minimum value is $\alpha - \frac{\dim \mathcal{D}_{h_i, d_{\varepsilon_2}}}{k - l_1^i - l_2^i}$. If $t - s + 1 \geq k$, $\bar{J} = \emptyset$, the minimum value will still be $\alpha - \frac{\dim \mathcal{D}_{h_i, d_{\varepsilon_2}}}{k - l_1^i - l_2^i}$, and the bound is

$$\mathcal{M}^s \leq (k - l_1 - l_2)(\alpha - \frac{\dim \mathcal{D}_{h_i, d_{\varepsilon_2}}}{k - l_1^i - l_2^i}) = (k - l_1 - l_2)\alpha - \dim \mathcal{D}_{R, \varepsilon_2}.$$

We can also use the bound $\dim \mathcal{D}_{R, \varepsilon_2} \geq \beta' = \beta$, yielding

$$\mathcal{M}^s \leq (k - l_1 - l_2)\alpha - \beta.$$

When $1 < g < k$, since we want minimize the upper bound, we need to minimize the size of J . Then set $u_i = t, i = 0, \dots, g - 2$, $g = \lfloor k/t \rfloor + 1$, so that $u_{g-1} = k - \lfloor k/t \rfloor t$. For $u_i = t$, $t - s + 1 - u_i \leq 0$, the minimum value comes from α ; for u_{g-1} , no matter whether $t - s + 1 - k + \lfloor k/t \rfloor t \geq 0$, or $t - s + 1 - k + \lfloor k/t \rfloor t < 0$, the minimum value always comes from α . We then get the upper bound

$$\begin{aligned} H(\mathbf{o}^s) \leq \min_{l_1^i \leq t} \{ & \sum_{i=0}^{g-2} (t - l_1^i - l_2^i)(\alpha - \frac{\dim \mathcal{D}_{h_i, d_{\varepsilon_2}}}{t - l_1^i - l_2^i}) \\ & + (k - \lfloor k/t \rfloor t - l_1^{g-1} - l_2^{g-1})(\alpha - \frac{\dim \mathcal{D}_{h_{g-1}, d_{\varepsilon_2}}}{k - \lfloor k/t \rfloor t - l_1^{g-1} - l_2^{g-1}}) \}. \end{aligned}$$

Also, $\dim \mathcal{D}_{h_i, d_{\varepsilon_2}}, i = 0, \dots, g - 1$, can be bounded by $\beta' = \beta$, yielding

$$\mathcal{M}^s \leq (k - l_1 - l_2)\alpha - g\beta = (k - l_1 - l_2)\alpha - (\lfloor k/t \rfloor + 1)\beta.$$

Comparing the above three conditions, we get the tightest bound when $g = k$, $u_i = 1$, $i = 0, \dots, 1$, which gives the result, matching the bound of (7.2.2). $\square$

Note that for the fully collaborative case $s = 1$, and the values of β becomes

$$\beta = \beta' = \frac{\alpha}{d - k + t}$$

as proven in [29].

The total repair bandwidth, based on the above proposition is then

$$d\beta + (t - s)\beta' = (d + t - s)\beta = \frac{\alpha(d + t - s)}{d - k + t - s + 1}.$$

Corollary 7.2.3. *For a partially collaborative code with no eavesdropper at the minimum storage point, we have*

$$\alpha = \frac{\mathcal{M}}{k}, \quad \beta = \beta' = \frac{\mathcal{M}}{k} \frac{1}{d - k + t - s + 1},$$

for a total repair bandwidth of

$$\gamma = \frac{\mathcal{M}}{k} \frac{d + t - s}{d + t - k - s + 1}.$$

7.2.2 Discussion

From (7.2.1) in Proposition 7.2.2, we get that the upper bound on the secure file size $\mathcal{M}^s$ depends on $\sum_{i=1}^{k-l_1-l_2} I(\mathbf{s}_i; \mathbf{d}_{i,\varepsilon_2})$. This sum is involving each node whose content $\mathbf{s}_i$ is not eavesdropped ($k - l_1 - l_2$ of them), and data originating from each of these nodes but being eavesdropped when it is downloaded or exchanged in the

set ε_2 of nodes. Therefore, for each $\mathbf{s}_i$, at most $(t-s)\beta'$ amount of data is involved in the collaboration process and at most $l_2\beta$ amount of data is downloaded later. For the same value t , and since $\beta = \beta'$, we thus have

$$I(\mathbf{s}_i; \mathbf{d}_{i,\varepsilon_2}) \leq (t-s)\beta' + l_2\beta = (t-s+l_2)\beta'.$$

Note that since $1 \leq s \leq t$, $t-s+l_2 \geq 0$, so apart if $t=s$ and $l_2=0$, we always have $t-s+l_2 \geq 1$ and thus for the range

$$\beta' \leq I(\mathbf{s}_i; \mathbf{d}_{i,\varepsilon_2}) \leq (t-s+l_2)\beta'$$

We get from from Proposition 7.2.2 the bound (7.2.2), and the secure file size $\mathcal{M}^s$ is upper bounded by

$$\mathcal{M}^s \leq (k-l_1-l_2)(\alpha-\beta) \tag{7.2.7}$$

with

$$\beta = \frac{\alpha}{d-k+t-s+1}.$$

We have

$$\alpha - \beta = \alpha \left(1 - \frac{\alpha}{d-k+t-s+1} \right) = \alpha \left(\frac{d-k+t-s}{d-k+t-s+1} \right).$$

We consider fixed α , $k-l_1-l_2$. Then, since both denominators are positive:

$$\frac{d-k+t-s}{d-k+t-s+1} \geq \frac{d-k+t-s_0}{d-k+t-s_0+1} \iff t-s \geq t-s_0.$$

Thus the secure file size is the largest when $t - s = t - 1$, that is, $s = 1$ (full collaboration), while it is the smallest when $t - s = 0$, that is $t = s$, and there is no collaboration.

Chapter 8

Conclusion

In this chapter, we summarize the problems studied and the results achieved in this thesis. We also give some directions for future work in this area.

Erasure codes are by now widely used in distributed storage systems. These systems need to be reliable for years, thus the repair of failures over time is of importance. The major issues of maintenance are the speed of repair but also its communication cost. With respect to repair, MDS erasure codes have some limitation. One approach to study storage codes, from the communication cost view point was inspired by network coding and gave rise to the concept of regenerating codes and collaborative regenerating codes.

The (fully) collaborative regenerating codes have been proved to reduce the communication cost. To make the design of regenerating codes more flexible, we introduced the notion of partial collaboration. From network coding, we know that gains in throughput gives security threat to the network, the more devices (nodes) communicated, the less security. Partial collaboration which reduces the number of nodes communicated may reduce the security threat, and provide a

good trade-off between reduction in repair bandwidth and security.

8.1 Results

In this thesis, we consider the analysis and the design of partially collaborative regenerating codes for distributed storage systems. We obtained the following results:

- We computed a min-cut bound, which gives a tradeoff between storage capacity and repair bandwidth. From this bound, we computed the parameters' value at minimum storage repair (MSR) point and minimum repair bandwidth repair (MBR) point.
- We designed codes for partial collaboration at MSR point and MBR point.
- We studied the application of orbit codes to distributed storage systems, which can be used to construct storage codes for constraints from no collaboration to full collaboration.

Also we investigated the security effect of partial collaboration against a passive eavesdropper.

8.2 Future Works

We propose a list of further open research questions.

- Optimal MBR codes: our approaches to construct codes can be further improved. In fact, the construction for MBR point in Section 5.3 varies for

different number of live nodes contacted, and the threshold of the failures could vary. A general code construction needs to be considered.

- Orbit codes: we only considered the orbit codes constructed from companion and diagonal matrices, and the construction for partially collaboration is not yet efficient enough. Thus using other matrices to construct orbit codes is a natural way to follow. Furthermore, one should look more closely at the automorphism group behind orbit codes to better characterize these codes. The bounds and dual codes of orbit codes could also be considered.
- Security: the systems need to be robust against adversaries not only passive, but also active. We only investigated the partial collaboration against passive ones. What are the performances against active adversaries need to be considered.

Bibliography

- [1] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, “Erasure coding in windows azure storage”, in *USENIX Annual Technical Conference (USENIX ATC)*, 2012.
- [2] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, “XORing elephants: Novel erasure codes for big data”, in *Proceedings of the VLDB Endowment*, 2013.
- [3] F. Oggier, A. Datta. “Coding Techniques for Repairability in Networked Distributed Storage Systems”, in the series *Foundations and Trends in Communications and Information Theory*, vol. 9, no. 4, June 2013.
- [4] A.-M. Kermarrec, N. Le Scouarnec, G. Straub, “Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes”, *The 2011 International Symposium on Network Coding (NetCod 2011)*.
- [5] K. Shum, “Cooperative Regenerating Codes for Distributed Storage Systems”, *The International Conference on Communications (ICC 2011)*, Kyoto. Available at <http://arxiv.org/abs/1101.5257>

- [6] K. W. Shum, Y. Hu, “Cooperative Regenerating Codes”, preprint, available at <http://arxiv.org/abs/1101.5257>
- [7] K. W. Shum and Y. Hu, “Exact Minimum-Repair-Bandwidth Cooperative Regenerating Codes for Distributed Storage Systems”, *The International Symposium on Information Theory (ISIT 2011)*, Saint-Petersburg.
- [8] N. Le Scouarnec, “Exact Scalar Minimum Storage Coordinated Regenerating Codes”, *The International Symposium on Information Theory (ISIT 2012)*, Boston. Available at <http://arxiv.org/abs/1202.0457>
- [9] J. Chen and K. W. Shum, “Repairing Multiple Failures in the Sub-Ramchandran Regenerating Codes”, *The International Symposium on Information Theory (ISIT 2013)*, Istanbul. Available at <http://arxiv.org/abs/1302.1256>
- [10] S. Jieka and N. Le Scouarnec, “CROSS-MBCR: Exact Minimum Bandwidth Coordinated Regenerating Codes”, *CoRR*, vol. abs/1207.0854, July 2012.
- [11] O. O. Koyluoglu, A. S. Rawat, S. Vishwanath, “The secrecy capacity of minimum bandwidth cooperative regenerating codes”, *The International Symposium on Information Theory (ISIT 2013)*, Istanbul. Available at <http://arxiv.org/abs/1210.3664>.
- [12] F. Oggier, A. Datta, “Byzantine Fault Tolerance of Regenerating Codes”, *The 11th IEEE International Conference on Peer-to-Peer Computing (P2P 2011)*, Kyoto. Available at <http://arxiv.org/abs/1106.2275>.

- [13] A.G. Dimakis, P.B. Godfrey, Y. Wu, M.J. Wainwright, K. Ramchandran, “Network Coding for Distributed Storage Systems”, *IEEE Trans. on Information Theory*, vol. 56, no. 9, Sept. 2010.
- [14] H.D.K. Hollmann, “Storage codes – coding rate and repair locality”, in the proceedings of the *International Conference on Computing, Networking and Communications (ICNC 13)*, San Diego, 2013.
- [15] F. Oggier, “Some Constructions of Storage Codes from Grassmann Graphs”, in the Proceedings of the *International Zurich Seminar on Communications, Zürich*, 2014.
- [16] N. Raviv, T. Etzion, “Distributed Storage Systems based on Equidistant Subspace Codes”, preprint, <http://arxiv.org/abs/1406.6170>.
- [17] A.-L. Trautmann and F. Manganiello and M. Braun and J. Rosenthal, “Cyclic Orbit Codes”, *IEEE Transactions on Information Theory*, vol. 59, no. 11, 2013.
- [18] F. Oggier, A. Datta, “Self-Repairing Codes for Distributed Storage - A Projective Geometric Construction”, in the proceedings of *IEEE Information Theory Workshop (ITW)*, 2011.
- [19] K.V. Rashmi, N.B. Shah, P.V. Kumar, K. Ramchandran, “Explicit construction of optimal exact regenerating codes for distributed storage”, in the Proceedings of the *Allerton conference on Communication, Control, and Computing*, 2009.
- [20] R. Lidl and H. Niederreiter, “Introduction to Finite Fields and their Applications”, Cambridge University Press, Cambridge, London, 1986.

- [21] O. Goldreich, “Foundations of Cryptography: Volume II, Basic Applications”, Cambridge University Press, Cambridge, London, 2004.
- [22] S. Ling and C.P. Xing, “Coding Theory: A First Course”, Cambridge University Press, Cambridge, London, 2004.
- [23] T. M. Cover and J. A. Thomas, “Elements of Information Theory”, 2nd ed. Wiley-Interscience, 2006.
- [24] H. Delfs and H. Knebl, “Introduction to Cryptography: Principles and Applications”, 2nd ed. Springer, 2007.
- [25] S. Pawar, S. El Rouayheb, K. Ramchandran, “Securing Dynamic Distributed Storage Systems against Eavesdropping and Adversarial Attacks”, *IEEE Trans. on Information Theory (Special Issue on Facets of Coding Theory: from Algorithm to Networks)*, vol. 57, no. 9, 2011.
- [26] N. B. Shah, K. V. Rashmi, P.V. Kumar, “Information Theoretically Secure Regenerating Codes for Distributed Storage”, in Proceedings of *IEEE Globecom*, Dec. 2011.
- [27] A. S. Rawat, O. O. Koyluoglu, N. Silberstein and S. Vishwanath, “Optimal Locally Repairable and Secure Codes for Distributed Storage Systems”, *IEEE Trans. Information Theory*, vol. 60, no. 1, pp. 212–236, Jan. 2014.
- [28] S. Pawar, S. El Rouayheb, K. Ramchandran, “On Secure Distributed Data Storage Under Repair Dynamics”, in the proceedings of the *2010 IEEE International Symposium on Information Theory (ISIT 2010)*, Austin.

- [29] O. O. Koyluoglu, A. S. Rawat and S. Vishwanath, “Secure Cooperative Regenerating Codes for Distributed Storage Systems”, Available at <http://arxiv.org/abs/1210.3664>.
- [30] P. Gopalan, C. Huang, H. Simitci and S. Yekhanin, “On the Locality of Codeword Symbols”, *IEEE Transactions on Information Theory*, vol. 58, pp. 6925-6934, Nov. 2012.
- [31] I. Tamo and A. Barg, “A Family of Optimal Locally Recoverable Codes”, *IEEE Transactions on Information Theory*, Vol. 60 , pp. 4661-4676, July 2014.
- [32] N. Silberstein, A. S. Rawat, O. O. Koyluoglu and S. Vishwanath, “Optimal Locally Repairable Codes with Local Minimum Storage Regeneration via Rank-Metric Codes”, Available at <http://arxiv.org/abs/1301.6331>.
- [33] G. M. Kamath, N. Prakash, V. Lalitha and P. V. Kumar, “Codes with Local Regeneration and Erasure Correction”, *IEEE Transactions on Information Theory*, Vol. 60 , pp. 4637 - 4660, July 2014.

List of Figures

- 1.1 A file is stored in one storage node, when this node experiences a failure, the file is lost. 2

- 1.2 A file is divided into two pieces (A, B), each of the pieces is stored three times. The figure shows that when a node storing A fails, it can be recovered by contacting one of the other two nodes storing A , when two nodes storing B fail, they can be recovered by contacting the one storing B which is still alive. 4

- 1.3 A file is encoded and stored in three nodes, each node stores A , B and $A + B$ respectively. When the node storing B fails, a new node comes, contacts the other two live nodes, and downloads A and $A + B$ respectively. After computation, the new node stores B which is the missing coefficient. 5

- 1.4 The upper case corresponding to exact repair, the lower case corresponding to functional repair. 7

1.5	A data object $\mathbf{o}$ of size M is stored across n of N storage nodes, each node of these n nodes stores α amount of data. When a node fails, a newcomer contacts d live nodes, downloads β amount of data from each of these d nodes, after computation with the downloaded data, it finally stores α amount of data. Any k out of n nodes can retrieve the object, it is sometimes said that a <i>data collector</i> may contact any k nodes to retrieve the data.	9
1.6	$t = 4$ failures repaired simultaneously, and each of these four nodes exchange β' with every other nodes.	12
1.7	A data object $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_4, \dots)$ is encoded using a generator matrix G with columns $\mathbf{g}_i$. Node i stores $\mathbf{o}_i\mathbf{g}_i, \mathbf{o}_{i+1}\mathbf{g}_i, \mathbf{o}_{i+2}\mathbf{g}_i, \mathbf{o}_{i+3}\mathbf{g}_i$, $i = 1, 2, 3, 4$, and the first subscript is understood modulo 4. This figure shows the repair of $t = 4$ failures (nodes 1,2,3,4): on the left of first line, with no collaboration, on the right of bottom line, with full collaboration. The two other figures show two intermediate modes of collaboration. The more the nodes collaborate, the less they download from live nodes, and vice-versa.	14
4.1	A source S with an object $\mathbf{o}$ encoded and distributed into n networked nodes. When some nodes failed, there is a fully collaborative repair. The data collector (DC) contacts k nodes to get the object.	33

4.2	A data object $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_4)$ is encoded using a generator matrix G with columns $\mathbf{g}_i$. Node i stores $\mathbf{o}_i\mathbf{g}_i, \mathbf{o}_{i+1}\mathbf{g}_i, \mathbf{o}_{i+2}\mathbf{g}_i, \mathbf{o}_{i+3}\mathbf{g}_i$, $i = 1, 2, 3, 4$, and the first subscript is understood modulo 4. This figure shows the repair of $t = 4$ failures (nodes 1,2,3,4): on the left of first line, with no collaboration $s = t = 4$, on the right of bottom line, with full collaboration $s = 1$. The two other figures show two intermediate modes of collaboration $s = 2, 3$. The more the nodes collaborate, the less they download from live nodes, and vice-versa.	34
4.3	Partial collaboration for $t = 4$ failures, and exchange data with $t - s = 2$ nodes, with three different cuts: the download links, the collaboration links and the storage capacity links are cut.	36
4.4	The total repair bandwidth γ as a function of $1 \leq s \leq 4$ for $t = 4$, for $k = 6$ (line below), and $k = 8$ (line above). The points at $s = 1$ correspond to the full collaboration case, while those at $s = 4$ correspond to no collaboration.	43
5.1	The total repair bandwidth γ as a function of $1 \leq s \leq 4$ for $k = 6$ and $t = 4$. The straight line corresponds to the generic coding strategy, while the other line is the optimal strategy.	49
5.2	Partially collaborative repair is illustrated based on Example 5.2.1, where $t = 4$ failures are repaired, and repair nodes collaborate by pairs.	52

6.1	The storage code proposed in Subsection 6.2 is optimal with respect to the minimum repair bandwidth, but not with respect to the storage capacity α . This shows the optimal value of α with respect to the storage needed for the construction, as a function of M . When $M = 4$, our construction is optimal. We set $d = 1$, $k = 2$ and $t - s = 1$	97
7.1	The secure file size $\mathcal{M}^s$ as a function of $1 \leq s \leq 5$ with $d = k = t = 5$, the lines from top to bottom represent the eavesdropper number l_1 from 1 to 4 respectively.	107