ABSTRACT

Title of dissertation: Coding Schemes for Distributed Storage Systems

Min Ye, Doctor of Philosophy, 2017

Dissertation directed by: Professor Alexander Barg
Department of Electrical and
Computer Engineering
and Institute for System Research

This thesis is devoted to problems in error-correcting codes motivated by data integrity problems arising in large-scale distributed storage systems. We study properties and constructions of Maximum Distance Separable (MDS) codes, which are widely used in storage applications since they provide the maximum failure tolerance for a given amount of storage overhead.

Among the parameters of the code that are important for storage applications are: the amount of data transferred in the system during node repair (the repair bandwidth), which characterizes the network usage, and the volume of accessed data, which corresponds to the number of disk I/O operations. Therefore, recent research on MDS codes for distributed storage has focused on codes that can minimize these two quantities. A lower bound on the repair bandwidth of a code, called the cut-set bound, was proved by Dimakis et al. in 2010, and codes that attain this bound are said to have the optimal repair property. Explicit optimal-repair low-rate (rate $\leq 1/2$) MDS codes were constructed by Rashmi et al. in 2011. At the same time, large-scale distributed systems such as the

Google File System and Hadoop Distributed File System, employ high-rate (rate $> 1/2$) MDS codes due to the need of reducing storage overhead. Until recently, except for some particular cases, no general explicit constructions of high-rate optimal-repair MDS codes were known.

In this thesis, we present the first explicit constructions of optimal-repair MDS codes, thereby providing a solution to the general construction problem of such codes for the high-rate regime.

More specifically, we construct explicit MDS codes that can repair any number of failed nodes from any number of helper nodes with the smallest possible amount of downloaded/accessed data. For the particular case of repairing a single node failure, we further present an explicit family of MDS codes that minimize the amount of accessed data during the repair. This family of codes has an additional favorable property that the node size (the amount of information stored in the node) is also the smallest possible. Reducing the node size directly translates into reducing the complexity of storage systems.

While most studies on MDS codes with optimal repair bandwidth focus on array codes, the repair problem of widely used scalar codes such as Reed-Solomon codes has also recently attracted attention of researchers. It has been an open problem whether scalar linear MDS codes can achieve the cut-set bound. In this thesis, we answer this question in the affirmative by giving explicit constructions of Reed-Solomon codes that can be repaired at the cut-set bound. We also prove a lower bound on the node size of optimally repairable scalar MDS codes, showing that the node size of our RS codes is close to the best possible for scalar linear codes.

Finally, we extend the concept of repair bandwidth from erasure correction to error

correction, which forms a new problem in coding theory. We prove a bound on the amount of downloaded information for this problem and present explicit code families that attain this bound for a wide range of parameters.

Coding Schemes for Distributed Storage Systems

by

Min Ye

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2017

Advisory Committee:
Professor Alexander Barg, Chair/Advisor
Professor Prakash Narayan
Professor Sennur Ulukus
Professor Armand Makowski
Professor Lawrence Washington

Acknowledgments

First I would like to thank my advisor Sasha Barg for his support and guidance during my PhD studies. Sasha gave me the freedom to pursue whatever interests me, and he was always available when I needed to talk with him.

I would also like to thank Itzhak Tamo. Part of this thesis results from collaboration with Itzhak. He is such a smart person, and he can always enlighten me. Most importantly, we had lots of fun working together.

I owe my deepest thanks to my family for their love and support.

I thank Yi-Peng Wei for his help and friendship when I had difficulties. In particular, he lent me his car when I had a car accident.

I had opportunities to discuss my research with several faculty members of University of Maryland, and has benefited a lot from these discussions. In particular, I am grateful to Professors Prakash Narayan and Leonid Koralov for their useful feedback.

I thank my roommates, undergraduate classmates, high school classmates and friends for their companionship and valuable friendship.

Finally, I am grateful to the members of my committee for their time and interest in my work.

Contents

Main notation and acronyms

Notation	Meaning
$[n]$	the set $\{1, 2, \ldots, n\}, n \in \mathbb{N}$
(n, k, l) array code	array code of code length n and dimension k, each codeword coordinate is a vector of length l
r	$r := n - k$ is the number of parity nodes of the code
l	the sub-packetization value of the code
$\mathcal{F}$	the subset of failed nodes
$\mathcal{R}$	the subset of helper nodes
$\text{GRS}_F(n, k, \Omega, v)$	(n, k) Generalized Reed-Solomon code with evaluation points $\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\} \subseteq F$ $\{(v_1 f(\omega_1), \ldots, v_n f(\omega_n)) \in F^n : f \in F[x], \deg f \leq k-1\}$, $v = (v_1, \ldots, v_n) \in (F^*)^n$
$\text{RS}_F(n, k, \Omega)$	(n, k) Reed-Solomon code over field F with evaluation points Ω
$\mathcal{C}^\perp$	the dual code of the code $\mathcal{C}$
$\text{tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}$	trace function $\text{tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\beta) = \beta + \beta^q + \beta^{q^2} + \cdots + \beta^{q^{m-1}}$
MDS codes	Maximum distance separable codes
MSR codes	Minimum-storage regenerating codes
LRC	Locally recoverable code
RS code	Reed-Solomon code

Chapter 1: Introduction

Distributed storage systems, such as those run by Google [18] and Facebook [14], find numerous applications ranging from social networks to file and video sharing. Currently deployed systems are formed of thousands of individual drives (nodes), and drive failures occur on a daily basis. A common approach to enhancing reliability of the data against frequent drive failures is to introduce redundancy in order to recover the unavailable contents. Many existing systems adopt a straightforward solution, storing multiple copies of the data [14, 18]. An obvious drawback of replicating the contents on multiple storage nodes is large storage overhead, which becomes unsustainable with the exponential increase of the volume of the information in large-scale storage systems. For this reason, companies utilizing or providing distributed storage solutions have increasingly turned to erasure (error) correcting coding for the efficient data recovery. The tradeoff between failure tolerance and storage overhead is an important measure of the performance of the system. In terms of code design, this amounts to constructing high-rate codes with large distance, which has been extensively studied.

At the same time, distributed storage systems introduce additional requirements for the code design. Measurements performed in real-world distributed storage systems show that single node failures occur far more frequently than failures of multiple nodes [41, Section 6.6]. This fact together with the distributed nature of the system present new challenges in the code design that are related to the repair efficiency of a single node failure or of a small number of failed nodes. Two important performance metrics of efficient repair that have emerged are the *locality* and *repair bandwidth*: locality is the number of nodes accessed during the repair procedure, and bandwidth is the amount of data communicated between the nodes. Efficient operation of the system often amounts to minimizing one of these two quantities; therefore, recent research on coding for distributed storage has focused on codes with either small bandwidth or small locality. Theoretical studies of codes with small locality, called locally recoverable codes (LRCs), were initiated by Gopalan et al. [19]. Bounds and several constructions of optimal LRCs are given in particular in the following works: [36, 57, 59, 60]. Codes with small repair bandwidth, introduced by Dimakis et al. [12], are also called *regenerating codes*. Regenerating codes form the central topic of this thesis, and we will say more about them later.

Decoding tasks with communication constraints

Node failures in distributed storage correspond to erasures of codeword coordinates, and most coding schemes for distributed storage are designed for erasure correction. At the same time, error correction (global decoding) is also a central focus of coding theory,

Metric \ Task	Repair single (several) erasure(s)	Global decoding (error correction)
Bandwidth	Regenerating codes	Fractional decoding
Locality	Locally Recoverable Codes (LRC)	LDPC codes

Table 1.1: Communication-constrained coding problems

especially for codes designed for data transmission. For such codes, small locality can also result in significant reduction of the (global) decoding complexity. In particular, codes in the well-known class of Low Density Parity Check (LDPC) codes [17] possess small locality of codeword coordinates which supports efficient and nearly optimal decoding algorithms. In this thesis we introduce the "fractional decoding" problem, which, we believe, is a new problem in coding theory. In this problem, we aim to design codes that minimize the bandwidth during the global decoding procedure in distributed systems. Equivalently, for a given bandwidth, we want to design optimal codes that maximize the number of correctable errors. Table 1.1 summarizes our discussion of various kinds of repair problems under communication constraints.

Throughout this thesis we adopt the point of view of codes that stems from distributed storage applications. In particular, in accordance with current literature, we use the terminology motivated by such applications however unusual it may feel in the general coding-theoretic context. In this vein, code's coordinates are called nodes, correcting erasures is referred to as repairing (or recovering) failed nodes, and the process of obtaining information stored at some other nodes of the codeword for the repair of the failed node(s) is described as "downloading data from the helper nodes". As usual in distributed storage applications, we assume that a code of length n formed of k information coordinates and $r = n - k$ parity coordinates is spread across n different nodes of the storage cluster. Each node of the cluster stores a coordinate of the code.

Referring to Table 1.1, in this thesis, we focus on the bandwidth problems. Today's large-scale distributed storage systems tend to use Maximum Distance Separable (MDS) codes since they provide the maximum failure tolerance for a given amount of storage overhead. Suppose that an (n, k) MDS code with k information nodes and $r = n - k$ parity nodes is used to encode data in a distributed storage system. Suppose further that the repair task of failed nodes is performed over some finite field F which we call the base field. The data stored in each node is viewed as a vector over F. The dimension l of this vector over F is called *sub-packetization*, and it plays a key role in our considerations. Since sub-packetization measures the size of each node, which in practice cannot be too large, we would like this parameter to be as small as possible.

Optimal-repair MDS codes

The following simple but important result of [8, 12] forms the starting point of our constructions.

Proposition 1.1. *Suppose that the data is encoded with an MDS code $\mathcal{C}$. Suppose that h*

out of the n nodes are inaccessible and d surviving (helper) nodes are used to recover the lost data. The minimum amount of information downloaded from all the helper nodes, or the repair bandwidth, satisfies the following inequality which forms a necessary condition for successful recovery:

$$\text{Repair bandwidth} \geq \frac{hdl}{d+h-k} \quad \text{symbols of } F. \tag{1.1}$$

If this lower bound is achieved for the repair of any h erased nodes from any d helper nodes, we say that the MDS code has the (h,d)-*optimal repair* property. We note that MDS codes with optimal repair bandwidth are also called *minimum storage regenerating* (MSR) codes in the literature [12], and we will use this term below alongside the more mainstream nomenclature.

MDS codes with optimal access

Storage applications further entail the following specialization of the coding tasks. The data downloaded in the repair process can form some functions such as linear combinations of the data stored in the helper nodes. Therefore, it may happen that we access a large portion of the data on the helper nodes, perform some transformation whose range is more limited, and download its outcome whose volume may still satisfy the bound on the repair bandwidth (1.1). This leads us to isolate a subclass of optimal-repair MDS codes for which the amount of information accessed in the helper nodes equals the amount of information downloaded for the repair task. We say that an (n,k) MDS code has the (h,d)-*optimal access* property if the repair of any h erased nodes using any d helper nodes can be accomplished by accessing the amount of data that meets the lower bound (1.1).

Repairing Reed-Solomon codes

Most studies of MDS codes with optimal repair bandwidth in the literature are concerned with a particular subclass of codes known as MDS *array codes* [4], also called vector codes because every node is a vector over the base field. Vector codes provide more flexibility in the code design compared with scalar linear codes. At the same time, it is of interest to study limitations of the repair bandwidth of scalar codes such as Reed-Solomon (RS) codes, which are widely used in applications including distributed storage systems. Following this line of thought, Guruswami and Wootters [26] initiated the study of repair bandwidth of RS codes and other linear scalar codes. They characterized linear repair schemes of such codes and showed that RS codes can be repaired using a smaller bandwidth than under the trivial approach.

1.1 Overview of the thesis

A summary of the contents of individual chapters is as follows.

Chapter 2: In this chapter, we formally introduce the basic concepts used in this thesis and give an overview of related works in the literature.

Chapter 3: ([77]) While constructions of optimal-repair codes of low rate ($R := k/n < 1/2$) have been found early on [42], explicit codes with optimal repair bandwidth for the practically important high-rate case were unknown. In this chapter, we present several explicit constructions of MDS array codes with optimal repair bandwidth for any k and n [77]. In particular, we resolve the most basic question of constructing such codes for the repair of a single failed node, or $(1, n-1)$-optimal repair MDS codes. Extending this result, for any r and n, we present explicit constructions of codes with the (h, d)-optimal repair property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously. The encoding, decoding, repair of failed nodes, and update procedures of these codes all have low complexity. Moreover, the codes that we construct are resilient toward errors during the repair process, i.e., they can correct erroneous information provided by the helper nodes, and their repair bandwidth meets the corresponding lower-bound extension of (1.1) given in (3.2) below. One of the code families presented in this chapter, in addition to the above, affords the optimal access property.

Chapter 4: ([78]) Some of the code families presented in Chapter 3 support the optimal access property. At the same time, according to [62], for $(1, n-1)$-optimal-access MDS codes, a necessary condition for the sub-packetization value l is $l \geq r^{(k-1)/r}$, and the constructions of Chapter 3 fall short of attaining or approaching this bound. Here we construct an explicit family of $(1, n-1)$-optimal-access MDS codes with $l = r^{\lceil n/r \rceil}$, which differs from the lower bound by at most a factor of r^2. These codes can be constructed over any finite field F as long as $|F| \geq n + r$, and afford low-complexity encoding and decoding procedures.

We also define a version of the repair problem that bridges the context of regenerating codes and LRCs, which we call *group repair with optimal access*. In this variation, we assume that the set of $n = sm$ nodes is partitioned into m repair groups of size s, and require that the amount of accessed data for repair is the smallest possible whenever the $d = s + k - 1$ helper nodes include all the other $s - 1$ nodes from the same group as the failed node. We construct an explicit family of codes with the group optimal access property. These codes exist over any field F of size $|F| \geq n$, and also afford low-complexity encoding and decoding procedures [78].

Chapter 5: ([64], [75]) Paper [26] and the follow-up papers stopped short of constructing RS codes (or any scalar MDS codes) that meet the lower bound (1.1) with equality, which has been an open problem in coding theory. Taking up this problem, for any n, k and $k \leq d \leq n-1$, we construct an (n, k) RS code with sub-packetization $l = \exp((1 + o(1))n \log n)$ that has $(1, d)$-optimal repair property [64]. We also prove an almost matching lower bound on l, showing that the super-exponential scaling is both necessary and sufficient for achieving the lower bound (1.1) using linear repair schemes. More precisely, we prove that for scalar MDS codes (including the RS codes) with optimal repair property, the sub-packetization l must satisfy $l \geq \exp((1 + o(1))k \log k)$.

While sub-packetization scaling slower than this is not possible for optimally repairable RS codes, it is possible to approximate this goal by allowing slightly

greater repair bandwidth in exchange for much smaller sub-packetization. Along these lines, for any n and r, we construct an $(n, k = n - r)$ RS code with sub-packetization $l = r^n$ that asymptotically achieves (1.1) for the repair of any single failed node from all the other $n - 1$ surviving nodes [75].

Chapter 6: ([63]) Coding-theoretic considerations in the literature, including the works discussed above, have accounted for three of the four possibilities in Table 1.1. In this chapter we target the fourth remaining possibility, introducing a "fractional decoding" problem, which extends the concept of repair bandwidth to the error correction setting. We derive an upper bound on the number of correctable errors under a given amount of information downloaded during the decoding process. We also present explicit code constructions that achieve this bound.

Research work not included in this thesis

While working toward my PhD degree at the University of Maryland, I also worked on problems other than those included in this thesis, specifically on *discrete distribution estimation* under locally differential privacy and on *polar codes*.

For the private estimation problem, we gave a new privatization scheme that outperforms the existing schemes in the literature [79, 80]. In these works we also proved a lower bound on the minimax estimation loss which showed that our scheme is order optimal. In a later work [76], we proved a more refined lower bound on estimation loss that enabled us to strengthen the order optimality claim by showing that the scheme of [79, 80] is in fact asymptotically optimal.

My research on polar codes includes constructing polar coding schemes for distributed hierarchical source coding [73] and universal source coding [72]. In [74] we also proposed new polar coding schemes using dynamic kernels that account for improved performance of the codes. Finally, in [23], we resolved the main question related to the construction of polar codes for alphabets other than binary. The latter case has been analyzed in [39], but their technique did not extend to nonbinary code alphabets. We proposed a new method that enabled us to estimate the capacity loss due to the construction algorithm, and subsequently to estimate the overall complexity of constructing polar codes.

Chapter 2: Preliminaries and Related Work

In this chapter, we begin with introducing the main notation and general background material used throughout this thesis. We then provide a high-level review of the related literature. More detailed discussion of related works is provided in the subsequent chapters.

2.1 Notation and definitions

An (n, k, l) MDS array code has k information nodes and $r = n - k$ parity nodes in each codeword with the property that any k out of n nodes can recover the codeword. Every node is a column vector in F^l, where F is some finite field, reflecting the fact that the system views a large data block stored in one node as one coordinate of the codeword. The parameter l is called sub-packetization.

Definition 2.1 (Repair bandwidth)**.** *Given an (n, k, l) MDS array code $\mathcal{C}$ and two disjoint subsets $\mathcal{F}, \mathcal{R} \subseteq [n]$ such that $|\mathcal{F}| \leq r$ and $|\mathcal{R}| \geq k$, we define $N(\mathcal{C}, \mathcal{F}, \mathcal{R})$ as the smallest number of symbols[1] of F one needs to download from the helper nodes $\{C_i, i \in \mathcal{R}\}$ in order to recover the erased nodes $\{C_i, i \in \mathcal{F}\}$. The* (h, d)-repair bandwidth *of the code $\mathcal{C}$ equals*

$$\max_{\substack{\mathcal{F}\subseteq[n],|\mathcal{F}|=h \\ \mathcal{R}\subseteq[n]\setminus\mathcal{F},|\mathcal{R}|=d}} N(\mathcal{C}, \mathcal{F}, \mathcal{R}).$$

where $[n] := \{1, 2, \ldots, n\}$.

Definition 2.2 (Cut-set bound [8, 12])**.** *Let $\mathcal{C}$ be an (n, k, l) MDS array code. For any two disjoint subsets $\mathcal{F}, \mathcal{R} \subseteq [n]$ such that $|\mathcal{F}| \leq r$ and $|\mathcal{R}| \geq k$, we have the following inequality:*

$$N(\mathcal{C}, \mathcal{F}, \mathcal{R}) \geq \frac{|\mathcal{F}||\mathcal{R}|l}{|\mathcal{F}| + |\mathcal{R}| - k}. \tag{2.1}$$

If (2.1) is achieved, we say that $\mathcal{C}$ can optimally repair *nodes $\{C_i, i \in \mathcal{F}\}$ using nodes $\{C_i, i \in \mathcal{R}\}$.*

Definition 2.3 (Optimal repair/access)**.** *If the (h, d)-repair bandwidth of $\mathcal{C}$ is $\frac{hdl}{h+d-k}$, meeting the lower bound in (2.1), we say that $\mathcal{C}$ has* (h, d)-optimal repair *property. We further say that the code has the* d-optimal repair *property if $h = 1$, omitting the reference to h, and say that the code has the* optimal repair *property if $h = 1$ and $d = n - 1$.*

[1] these symbols can be some functions of the contents of the nodes $\{C_i, i \in \mathcal{R}\}$.

*Similarly, we say that an (n, k, l) MDS array code $\mathcal{C}$ has (h, d)-*optimal access *property if the repair of any h erased nodes using any d helper nodes can be accomplished by accessing the amount of data that meets the lower bound in (2.1). If $h = 1$ and $d = n - 1$, We simply say that the code has the* optimal access *property.*

In this thesis we need to use Generalized Reed-Solomon (GRS) codes. Let us recall the definition of GRS codes and dual codes.

Definition 2.4. *Let F be a finite field. A* Generalized Reed-Solomon code $\mathrm{GRS}_F(n, k, \Omega, v) \subseteq F^n$ *of dimension k over F with evaluation points $\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\} \subseteq F$ is the set of vectors*

$$\{(v_1 f(\omega_1), \ldots, v_n f(\omega_n)) \in F^n : f \in F[x], \deg f \leq k - 1\}$$

where $v = (v_1, \ldots, v_n) \in (F^*)^n$ are some nonzero elements. If $v = (1, \ldots, 1)$, then the GRS code is called a Reed-Solomon code and is denoted as $\mathrm{RS}_F(n, k, \Omega)$. (If the symbol field F is clear from the context or not important, we sometimes omit it in the subscript.)

Definition 2.5 (Dual code)**.** *The dual code of a linear code $\mathcal{C} \subseteq F^n$ is the linear subspace of F^n defined by*

$$\mathcal{C}^{\perp} = \Big\{x = (x_1, \ldots, x_n) \in F^n \Big| \sum_{i=1}^{n} x_i c_i = 0 \quad \forall c = (c_1, \ldots c_n) \in \mathcal{C}\Big\}.$$

It is well known [35, p.304] that

$$(\mathrm{RS}_F(n, k, \Omega))^{\perp} = \mathrm{GRS}_F(n, n - k, \Omega, v), \tag{2.2}$$

where $v_i = \prod_{j \neq i} (\omega_i - \omega_j)^{-1}, i = 1, \ldots, n$. (The dual of an RS code is a GRS code.)

2.2 General code construction of MSR array codes

In Chapter 3-4, we will construct various families of MDS array codes with optimal repair/access property. All the constructions in these two chapters are defined within the following general framework. Let $\mathcal{C} \in F^{ln}$ be an (n, k, l) array code with nodes $C_i \in F^l, i = 1, \ldots, n$, where each C_i is a column vector with coordinates $C_i = (c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})^T$. Throughout the next two chapters we consider codes defined by the following r parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0,\ t = 0, 1, \ldots, r - 1\}, \tag{2.3}$$

where $A_{t,i}, 0 \leq t \leq r - 1, 1 \leq i \leq n$ are $l \times l$ matrices over F. The specific code families in Chapter 3-4 are obtained by choosing different forms of these matrices.

2.3 Related literature

In this section, we provide a brief overview of earlier works on regenerating codes and related coding problems.

Regenerating codes

Among the works on coding for distributed storage, papers on regenerating codes are most closely related to this thesis. Regenerating codes were introduced by Dimakis et al. [12] with the aim of optimizing the amount of data downloaded during the repair procedure. In particular, the authors of [12] derived a trade-off relation between storage capacity per node and the optimal repair bandwidth, and they also showed the existence of codes achieving this trade-off. Two end points of this trade-off curve have special meaning. The point that corresponds to the minimum storage overhead is called the Minimum Storage Regenerating (MSR) point and opposite corner of the curve is called the Minimum Bandwidth Regenerating (MBR) point. As we mentioned before, MSR codes are MDS codes with optimal repair bandwidth, and they are of particular interest because MDS codes also optimize the tradeoff between failure tolerance and storage overhead.

It is worth mentioning that the results in [12] are derived under the so-called *functional repair* model, in which one does not need to recover the exact contents stored on the failed nodes. Later on, most research, including this thesis, has focused on a stronger requirement called *exact repair*, wherein we need to recover the exact contents during the repair procedure. Both MSR and MBR points are achievable under the exact repair requirement for any given parameters: see [42] for MBR constructions and low rate MSR constructions; see [77] (also Chapter 3 of this thesis) for high-rate MSR constructions. At the same time, the intermediate points of the trade-off curve derived in [12] are not achievable for exact repair [53, 65]. A number of subsequent papers presented code constructions and tighter lower bounds for intermediate points [15, 16, 20, 40, 50].

Below we only consider the exact repair problem. A large part of this thesis is devoted to the construction of MSR codes and bounds on sub-packetization of scalar linear MSR codes. Other relevant papers devoted to the existence as well as explicit constructions of MSR codes include [6, 7, 37, 54, 58, 61]. In the following table we list papers that contain state-of-the-art results on MSR codes and some related problems of importance to us. (In the "Contributions" column we only list the relevant results. Some of the cited papers also contain other results, which were surpassed by later works.)

Papers	Contributions
Rashmi et al., 2011 [42]	gave explicit constructions of low-rate (rate $\leq 1/2$) MSR codes with optimal sub-packetization value
Ye and Barg, 2017 [77] (Chapter 3 of this thesis)	This is the first paper to give explicit constructions of high-rate (rate $> 1/2$) MSR codes with more than 3 parities. This is also the only paper that contains explicit constructions of universal MDS codes with the (h, d)-optimal repair property for all possible values of h and d simultaneously
Ye and Barg, 2017 [78] (Chapter 4 of this thesis)	gave explicit constructions of MDS codes with optimal access property and nearly optimal sub-packetization $l = r^{\lceil n/r \rceil}$. This is the best known explicit construction of optimal-access MDS codes
Sasidharan et al., 2016 [51]; Li et al., 2017 [33]	independently gave explicit constructions of optimal-access MDS codes whose parameters (sub-packetization and field size) are exactly the same as the codes presented in [78]. Constructions in both papers are very similar to the constructions in [78]. Both papers [51] and [33] appeared after the initial release of [78]
Wang et al., 2016 [68]	gave an existence proof of MSR codes with sub-packetization $l = r^{\lceil n/(r+1) \rceil}$, which is the best known achievable sub-packetization value of MSR codes. For the case $r = 2$, [68] gave an explicit construction of MSR codes with the best known sub-packetization value $l = 2^{\lceil n/3 \rceil}$
Raviv et al., 2017 [44]	For the case $r = 3$, gave an explicit construction of MDS codes with sub-packetization $l = r^{k/4}$ that can optimally repair any single systematic node. This construction can be transformed into an MSR code with sub-packetization $l = r^{\lceil n/4 \rceil}$. This is the best known sub-packetization value of explicit MSR codes for the case $r = 3$
Tamo et al., 2014 [62]	showed that for an MDS code with the optimal access property, the sub-packetization satisfies the lower bound $l \geq r^{(k-1)/r}$
Goparaju et al., 2014 [22]	showed that the sub-packetization of an MSR code satisfies the lower bound $l \geq 2^{\sqrt{k/(2r-1)}}$, indicating that for a fixed r, the growth rate of l is super-polynomial in n
Guruswami and Rawat, 2017 [24]; Rawat et al., 2017 [48]	constructed MDS codes with nearly optimal repair bandwidth whose sub-packetization level l is independent of n for a fixed r
Sasidharan et al., 2017 [52]	For the case $r \mid k$ and $2 \mid r$, gave an explicit construction of MSR code with $d = n - (r/2 + 1)$ and $l = \frac{1}{2} r^{n/r}$. This is the best known explicit construction (in terms of l) for this particular choice of parameters n, k and d

We remark that the (non-explicit) codes presented in [68] can only optimally repair systematic nodes, and the sub-packetization value of their construction is $l = r^{k/(r+1)}$. Applying a simple transformation, one can obtain the (non-explicit) MSR codes listed in the table.

Improving repair bandwidth of well-known codes

The repair problem applies to any code family, and the ideas developed in the study of regenerating codes proved useful in analyzing the repair bandwidth of Reed-Solomon (RS) codes. Apart from theoretical interest, the problem of repairing RS codes has strong connections to applications because, for instance, Facebook uses the $(14, 10)$ RS code to protect data on its storage clusters. The first paper to study repair of RS codes was [26]. This paper not only initiated this line of research, but also characterized essentially all

possible linear repair schemes of scalar codes. Subsequently, [10,11,64,75] extended this line of work. We address the question of repairing RS codes in Chapter 5 of this thesis.

Finally we note that binary MDS array codes such as EVEN-ODD and RDP codes [3, 9] are widely used in storage systems, and the repair problem of these codes was studied in [66,71].

Communication efficient secret sharing

Secret sharing is one of the basic cryptographic primitives that enters a large number of secure data storage or exchange protocols. One of the standard constructions of secret sharing schemes, due to Shamir [55] essentially makes use of Reed-Solomon codes (or MDS codes in a broad sense). Therefore, the idea of regenerating codes can naturally be used to construct communication-efficient (low-bandwidth) secret sharing schemes [2, 29, 30]. Although we do not investigate the secret sharing problem in this thesis, we believe the ideas and techniques that we introduce might be useful for constructing communication-efficient secret sharing schemes.

Regenerating codes with security constraints

Several works considered regenerating codes with adversaries [28,38,43,45]. One of the most common adversarial settings for the repair problem assumes that a certain limited amount of helper nodes are compromised, and provide incorrect information for the repair task. Papers [38] and [43] considered the *error resilience* capability in the repair process of a single node failure, and we will use this concept in Chapter 3.

Chapter 3: Explicit constructions of high-rate MSR codes for all parameters

In this chapter, we study high-rate MSR codes. Until recently, explicit constructions of such codes in the literature were only available for the cases of at most 3 parity nodes. Moreover, the only case analyzed in previous research was optimal repair of a single node failure by accessing all the surviving nodes.

In this chapter, we resolve the construction problem of explicit high-rate MSR codes. The results that we present constitute the first and the only known constructions of such codes for general parameters in the literature. As our first result, we present an explicit construction of $(1, n-1)$-optimal repair MDS codes, thereby addressing the most basic case of MSR codes. Extending this result, for any r and n, we present two explicit constructions of MDS array codes with the (h, d)-optimal repair property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously. Codes in the first family can be constructed over any base field F as long as $|F| \geq sn$, where $s = \text{lcm}(1, 2, \ldots, r)$. The encoding, decoding, repair of failed nodes, and update procedures of these codes all have low complexity. Codes in the second family have the optimal access property and can be constructed over any base field F as long as $|F| \geq n + 1$. Moreover, both code families have the optimal error resilience capability when repairing failed nodes. We also construct several other related families of MDS codes with the optimal repair property.

The results presented in this chapter were published in [77].

3.1 Introduction

The main objective of this chapter is to construct MDS codes with the optimal repair/access property. Recall from Chapter 2 that an (n, k, l) MDS array code over the field F has the (h, d)-optimal repair/access property if the repair of any h erased nodes using any d helper nodes can be accomplished by downloading/accessing $\frac{hdl}{h+d-k}$ symbols of F; see (2.1).

Developing the above setting, we also consider an adversarial variant of the repair problem, which assumes that a certain limited amount of helper nodes are compromised, and provide incorrect information for the repair task. In [38, 43], the authors introduced the *error resilience* capability in the repair process of a single node failure. We generalize this concept to the problem of repairing multiple node failures. Given an (n, k, l) MDS array code $\mathcal{C}$, a nonnegative integer t, and two disjoint subsets $\mathcal{F}, \mathcal{R} \subseteq [n]$ such that $|\mathcal{F}| \leq r$ and $|\mathcal{R}| \geq k + 2t$, define $N(\mathcal{C}, \mathcal{F}, \mathcal{R}, t)$ as the smallest number of symbols of F one needs to download from nodes $\{C_i, i \in \mathcal{R}\}$ such that the erased nodes $\{C_i, i \in \mathcal{F}\}$ can be recovered from these symbols as long as the number of erroneous nodes in $\{C_i, i \in \mathcal{R}\}$

is no larger than t. It is shown in [38, 43] that for any (n, k, l) MDS array code $\mathcal{C}$,

$$N(\mathcal{C}, \mathcal{F}, \mathcal{R}, t) \geq \frac{|\mathcal{F}||\mathcal{R}|l}{|\mathcal{F}| + |\mathcal{R}| - 2t - k}, \tag{3.1}$$

where t is a nonnegative integer and $\mathcal{F}, \mathcal{R} \subseteq [n]$ are disjoint subsets such that $|\mathcal{F}| = 1$ and $|\mathcal{R}| \geq k + 2t$. The method in [38, 43] can be straightforwardly generalized to show that (3.1) also holds for any $\mathcal{F}$ such that $|\mathcal{F}| \leq r$. We arrive at the following definition.

Definition 3.1. *We say that an (n, k, l) MDS array code $\mathcal{C}$ has the* universally error-resilient (UER) (h, d)-optimal repair *property if*

$$N(\mathcal{C}, \mathcal{F}, \mathcal{R}, t) = \frac{h(d + 2t)l}{h + d - k} \tag{3.2}$$

for any nonnegative integer t and any two disjoint subsets $\mathcal{F}, \mathcal{R} \subseteq [n]$ such that $|\mathcal{F}| = h$ and $|\mathcal{R}| = d + 2t$. When $h = 1$, we omit it from the notation. We define the UER (h, d)-optimal access *property in a similar way.*

For $k \leq (n + 1)/2$ (the low rate regime), MDS array codes with d-optimal repair property were constructed in [42, 54, 58, 70], and MDS array codes with the UER d-optimal repair property were constructed in [43]. For arbitrary code rate, [8] proved that there exists a family of codes for which the bound (2.1) is asymptotically achieved when $l \to \infty$. For finite l and $k > (n + 1)/2$ (the high-rate regime) papers [6, 7, 37, 61, 68] showed that for large enough base field F there exist MDS array codes that can optimally repair any single systematic node failure using all the surviving nodes, and [67] showed the same for all rather than only systematic nodes.

Among the very recent additions to the literature, [47] showed existence of MDS array codes with d-optimal repair property for any single value of d in the range $k \leq d \leq n - 1$. Paper [21] showed existence of MDS array codes that can optimally repair any single systematic node failure by accessing any subset of the surviving nodes as long as the number of accessed nodes is no less than k. Finally [69] showed existence of MDS array codes which can optimally repair any h systematic node failures using all the $n - h$ surviving nodes for any single value of h in the range $1 \leq h \leq r$ (some special cases for $h = 2$ and 3 are discussed in [46]). These papers essentially rely on existential lemmas in large finite fields, e.g., the Schwartz-Zippel lemma or Combinatorial Nullstellensatz. At the same time, all the known explicit constructions in the high-rate regime are obtained only for the case of at most 3 parity nodes, and are further limited to repairing $h = 1$ failed node by accessing all the $n - 1$ surviving nodes; see [37, 44, 61, 67, 68].

Remark 3.1. *After the release of the results of this chapter (as arXiv:1604.00454, see [77]), papers [33, 51, 52, 78] proposed several explicit constructions of $(1, n-1)$-optimal repair MDS codes with smaller sub-packetization for general values of n and r. At the same time, there papers did not consider the general case of arbitrary h and d.*

3.1.1 Overview of the chapter

In this chapter, given any n and k, we present several explicit constructions of (n, k, l) MDS array codes. Our constructions are grouped around two main code families. Codes in the first family have the UER (h, d)-optimal repair property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously, and the encoding, decoding, repair and update procedures of these codes all have low complexity. Codes in the second family have the UER (h, d)-optimal access property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously, and can be constructed over any base field F as long as $|F| \geq n + 1$.

This chapter is organized as follows. In Sections 3.3-3.6 we present the first code family and its variants. Sections 3.7-3.10 are devoted to the second family and related results. Finally, in Sect. 3.11 we present a new class of MDS array codes with optimal repair and smaller l than the other constructions. In the next few paragraphs we give a more detailed overview of our results.

In Section 3.3, we present our first construction of $(n, k, l = r^n)$ MDS array codes with the optimal repair property using any field F of size $|F| \geq rn$. The encoding, decoding, and repair of a single failed node involve only simple operations with $r \times r$ matrices over F, and thus have low complexity. An additional property of the proposed codes is optimal update, i.e., the need to change only the minimum possible number of coordinates in the parity nodes if one coordinate in systematic node is updated. In our construction we rely on a (non-systematic) parity-check representation of the codes as opposed to the systematic generator form used in most earlier works. This representation does not distinguish between systematic nodes and parity nodes, and leads naturally to the optimal repair of all nodes. The parity-check form combined with the block Vandermonde structure [68] and the idea of using r-ary expansions [6, 61] makes the explicit construction for larger number of parity nodes possible. Note that exponentially large l is necessary for optimal repair bandwidth: Indeed, according to a result of [22], $l \geq 2^{\sqrt{k/(2r-1)}}$ is necessary for any code with the optimal repair property. It is further shown in [62] that $l \geq r^{(k-1)/r}$ for any code with the optimal access property.

In Section 3.4, we extend the construction of Section 3.3 to obtain (n, k, l) MDS array codes with the UER d-optimal repair property for any positive integers n, k, d, l such that $k \leq d < n, l = (d + 1 - k)^n$ using any field F of size $|F| \geq (d + 1 - k)n$. We first observe that we only need to know a $1/(d + 1 - k)$ fraction of data stored in each of the surviving nodes $C_j, j \neq i$ in order to recover the erased node C_i. We then use a novel method to prove that these data form an $(n - 1, d, l/(d + 1 - k))$ MDS array code, and thus establish the UER d-optimal repair property.

In Section 3.5, we present another extension, constructing MDS codes with d-optimal repair property for several values of d simultaneously. Moreover, we show that (n, k, r^n) MDS array codes constructed in Section 3.3 will automatically have the d-optimal repair property for all d such that $(d + 1 - k)|(n - k)$. In Section 3.6 we further extend our construction to obtain (n, k, l) MDS array codes with the UER (h, d)-optimal repair property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously, where $l = s^n, s = \text{lcm}(1, 2, \ldots, r)$. These codes also have the optimal update property. Moreover, the encoding, decoding, and repair procedures only require operations with matrices of size not greater than $n \times n$.

In Section 3.7 we develop the idea of using permutation matrices [6, 61] to obtain an explicit family of $(n, n-r, r^{n-1})$ MDS array codes with the optimal access property, which can be constructed over any base field F such that $|F| \geq n+1$. In Sections 3.8-3.10, we combine the ideas in Section 3.7 and in Sections 3.4-3.6 to obtain an explicit family of (n, k, l) MDS array codes with the UER (h, d)-optimal access property for all $h \leq r$ and $k \leq d \leq n-h$ simultaneously, where $l = s^n, s = \text{lcm}(1, 2, \ldots, r)$. These codes can be constructed over any base field F as long as $|F| \geq n+1$.

In Section 3.11 we introduce a new class of MDS array codes, *Generalized Reed-Solomon Array Codes*, and use their properties to extend the construction of Section 3.7 to obtain MDS array codes with the d-optimal repair property for several values of d simultaneously. These codes also only require the underlying field size $|F| \geq n+1$ as well as a smaller l compared to the other code families in this chapter.

3.2 General code construction

Given positive integers r and n, define an $(n, k = n-r, l)$ array code $\mathcal{C}$ by setting in (2.3)

$$A_{t,i} = A_i^t, 0 \leq t \leq r-1, 1 \leq i \leq n, \tag{3.3}$$

where $A_1, A_2, \ldots, A_n$ are some $l \times l$ matrices. (We use the convention $A^0 = I$.) The specific code families in Section 3.3-3.11 are obtained by choosing different forms of the matrices $A_1, A_2, \ldots, A_n$.

3.3 Construction of MDS array codes with optimal repair property

3.3.1 Code construction

Construction 3.1. *Let F be a finite field of size $|F| \geq rn$, and let $l = r^n$. Let $\{\lambda_{i,j} : i \in [n], j = 0, 1, \ldots, r-1\}$ be rn distinct elements in F. Consider the code family given by (2.3)-(3.3), where we take*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_a^T, \ i = 1, \ldots, n.$$

Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F, and a_i is the i-th digit from the right in the representation of a in the r-ary form, $a = (a_n, a_{n-1}, \ldots, a_1)$.

Each A_i is a diagonal matrix, where the diagonal entry in the a-th row is λ_{i,a_i}. Using this observation, we can write out the parity-check equations (2.3) coordinatewise. Let $c_{i,a}$ denote the a-th coordinate of the column vector C_i for all $a = 0, \ldots, l-1$, i.e., $C_i = (c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})^T$. We have

$$\sum_{i=1}^{n} \lambda_{i,a_i}^t c_{i,a} = 0 \tag{3.4}$$

for all $t = 0, \ldots, r-1$ and $a = 0, \ldots, l-1$.

Theorem 3.1. *Codes given by Construction 3.1 attain optimal repair bandwidth for repairing any single failed node.*

Proof. For $u = 0, 1, \ldots, r-1$, let $a(i,u) := (a_n, \ldots, a_{i+1}, u, a_{i-1}, \ldots, a_1)$. We will show that for any $i \in [n]$ and $a = 0, 1, \ldots, l-1$, the coordinates $\{c_{i,a(i,0)}, c_{i,a(i,1)}, \ldots, c_{i,a(i,r-1)}\}$ in C_i are functions of the following set of $n-1$ elements of F:

$$\mu_{j,i}^{(a)} := \sum_{u=0}^{r-1} c_{j,a(i,u)}, \quad j \in [n] \backslash \{i\}. \tag{3.5}$$

In other words, each surviving node only needs to transmit one scalar in F to recover r coordinates in the failed node. Indeed, let us fix some u. Replacing a with $a(i,u)$ in (3.4), we obtain

$$\lambda_{i,u}^t c_{i,a(i,u)} + \sum_{j \neq i} \lambda_{j,a_j}^t c_{j,a(i,u)} = 0. \tag{3.6}$$

Summing (3.6) over $u = 0, 1, \ldots, r-1$ and then writing the result in matrix form, we get

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{i,0} & \lambda_{i,1} & \ldots & \lambda_{i,r-1} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{i,0}^{r-1} & \lambda_{i,1}^{r-1} & \ldots & \lambda_{i,r-1}^{r-1} \end{bmatrix} \begin{bmatrix} c_{i,a(i,0)} \\ c_{i,a(i,1)} \\ \vdots \\ c_{i,a(i,r-1)} \end{bmatrix} = - \begin{bmatrix} \sum_{j \neq i} \mu_{j,i}^{(a)} \\ \sum_{j \neq i} \lambda_{j,a_j} \mu_{j,i}^{(a)} \\ \vdots \\ \sum_{j \neq i} \lambda_{j,a_j}^{r-1} \mu_{j,i}^{(a)} \end{bmatrix}. \tag{3.7}$$

By construction $\lambda_{i,0}, \ldots, \lambda_{i,r-1}$ are distinct, so we can solve this system for $\{c_{i,a(i,0)}, c_{i,a(i,1)}, \ldots, c_{i,a(i,r-1)}\}$ given the set of elements in (3.5).

To explain how this argument supports the overall repair of the node C_i, observe that we can partition the l coordinates of C_i into r^{n-1} disjoint groups of size r each, where the row indices of the coordinates within the same group differ only in the i-th digit of their r-ary expansions. In other words, each group in the partition can be written as $\{c_{i,a(i,0)}, c_{i,a(i,1)}, \ldots, c_{i,a(i,r-1)}\}$ for some $a \in \{0, 1, \ldots, l-1\}$. As shown above, the coordinates in each group can be repaired by downloading one element of the field F from each of the $n-1$ surviving nodes. Therefore, overall the node C_i can be repaired by downloading r^{n-1} elements of F from each surviving node, achieving the optimal repair bandwidth of $(n-1)(l/r)$, cf. (2.1). □

The repair procedure of a single node has low complexity: indeed, according to (3.7), it can be accomplished by operations with $r \times r$ matrices (rather than much larger $l \times l$ matrices).

Theorem 3.2. *The code $\mathcal{C}$ given by Construction 3.1 is MDS.*

Proof. We write out the parity-check equations (2.3) coordinatewise. For all $a = 0, 1, \ldots, l-1$, we have

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{1,a_1} & \lambda_{2,a_2} & \ldots & \lambda_{n,a_n} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{1,a_1}^{r-1} & \lambda_{2,a_2}^{r-1} & \ldots & \lambda_{n,a_n}^{r-1} \end{bmatrix} \begin{bmatrix} c_{1,a} \\ c_{2,a} \\ \vdots \\ c_{n,a} \end{bmatrix} = 0 \tag{3.8}$$

Clearly every r columns of the parity-check matrix in (3.8) have rank r, so any k out of n elements in the set $\{c_{1,a}, c_{2,a}, \ldots, c_{n,a}\}$ can recover the whole set. Since this holds for all $a = 0, 1, \ldots, l-1$, we conclude that any k nodes of a codeword in $\mathcal{C}$ can recover the whole codeword. □

Remark 3.2. (Relation of Construction 1 to Reed-Solomon codes) *The codewords of the code $\mathcal{C}$ given by Construction 3.1 can be written as $l \times n$ matrices, where the columns correspond to the nodes. Eq. (3.8) implies that for each $a \in \{0, 1, \ldots, l-1\}$, the row indexed by $a = (a_n, \ldots, a_1)$ of this matrix is encoded as a Reed-Solomon code with the evaluation points $\lambda_{1,a_1}, \lambda_{2,a_2}, \ldots, \lambda_{n,a_n}$ independently of other rows (this corresponds to the diagonal structure of the parity-check matrices A_i). Therefore, the MDS property of the array code $\mathcal{C}$ is inherited immediately from the MDS property of RS codes. By choosing the evaluation points of each row using the r-ary expansion idea, we also achieve the optimal repair bandwidth.*

3.3.2 Complexity of encoding, decoding, and updates

The code given by Construction 3.1 can be efficiently transformed into systematic form. Without loss of generality we assume that the first k nodes are systematic (information) nodes. By (3.8), for all $a = 0, 1, \ldots, l-1$, we have

$$\begin{aligned} &\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{k+1,a_{k+1}} & \lambda_{k+2,a_{k+2}} & \ldots & \lambda_{k+r,a_{k+r}} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{k+1,a_{k+1}}^{r-1} & \lambda_{k+2,a_{k+2}}^{r-1} & \ldots & \lambda_{k+r,a_{k+r}}^{r-1} \end{bmatrix} \begin{bmatrix} c_{k+1,a} \\ c_{k+2,a} \\ \vdots \\ c_{k+r,a} \end{bmatrix} \\ &= - \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{1,a_1} & \lambda_{2,a_2} & \ldots & \lambda_{k,a_k} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{1,a_1}^{r-1} & \lambda_{2,a_2}^{r-1} & \ldots & \lambda_{k,a_k}^{r-1} \end{bmatrix} \begin{bmatrix} c_{1,a} \\ c_{2,a} \\ \vdots \\ c_{k,a} \end{bmatrix}. \end{aligned} \tag{3.9}$$

Consequently, in the encoding process we do not need to invert an $rl \times rl$ matrix, instead, we only need to invert $r \times r$ matrices l times, gaining a factor of l^2 in complexity. Similarly, in the decoding process, if some r nodes are erased, then in order to recover them, we only need to invert $r \times r$ matrices l times.

Another useful parameter of codes is *update complexity* [4]. On account of the MDS property, in order to update the value of a stored element $c_{i,a}$ in an information node, one needs to update at least one coordinate in every parity node [62]. From (3.9) it is easy to see that for any $i \in [k]$ and $a = 0, \ldots, l-1$, to update $c_{i,a}$, we only need to update $c_{k+1,a}, \ldots, c_{k+r,a}$. Thus Construction 3.1 gives an optimal update code.

3.4 Explicit MDS array codes with the UER d-optimal repair property

The general construction in (2.3)-(3.3) can also be used to construct an $(n, k = n - r, l)$ MDS array code $\mathcal{C}$ with the UER d-optimal repair property, $k \leq d \leq n - 1$.

Construction 3.2. *Let F be a finite field of size $|F| \geq sn$, where $s = d + 1 - k$. Let $\{\lambda_{i,j}\}_{i\in[n],j=0,1,\ldots,s-1}$ be sn distinct elements in F. Consider the code family given by (2.3)-(3.3), where $l = s^n$ and*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_a^T, \ i = 1, \ldots, n.$$

Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F and a_i is the i-th digit from the right in the representation of a in the s-ary form, $a = (a_n, a_{n-1}, \ldots, a_1)$.

Define $a(i, u)$ and $c_{i,a}$ in the same way as in Sect. 3.3.

Theorem 3.3. *The code $\mathcal{C}$ given by Construction 3.2 is an MDS code.*

Proof. Same as the proof of Theorem 3.2. □

By the same arguments as in the previous section, $\mathcal{C}$ also has low-complexity encoding, decoding, and the optimal update property.

Let us show that the code $\mathcal{C}$ has the UER d-optimal repair property. First recall a simple property of GRS codes.

Remark 3.3. *The minimum distance of the code* GRS(n, k, Ω, v) *is $n - k + 1$. Note that the projection of the GRS code on any subset of coordinates $\Omega' \subset \Omega$, $|\Omega'| \geq k$, is itself a GRS code. In particular, its distance equals $|\Omega'| - k + 1$.*

Theorem 3.4. *The code $\mathcal{C}$ given by Construction 3.2 has the UER d-optimal repair property.*

Proof. Without loss of generality, we consider the case of repairing C_1. Let

$$\mu_{j,1}^{(a)} := \sum_{u=0}^{s-1} c_{j,a(1,u)}, \quad j \in \{2, 3, \ldots, n\}. \tag{3.10}$$

Using arguments similar to those that lead to (3.7), we obtain

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{1,0} & \lambda_{1,1} & \ldots & \lambda_{1,s-1} \\ \lambda_{1,0}^2 & \lambda_{1,1}^2 & \ldots & \lambda_{1,s-1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{1,0}^{r-1} & \lambda_{1,1}^{r-1} & \ldots & \lambda_{1,s-1}^{r-1} \end{bmatrix} \begin{bmatrix} c_{1,a(1,0)} \\ c_{1,a(1,1)} \\ \vdots \\ c_{1,a(1,s-1)} \end{bmatrix} = - \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{2,a_2} & \lambda_{3,a_3} & \ldots & \lambda_{n,a_n} \\ \lambda_{2,a_2}^2 & \lambda_{3,a_3}^2 & \ldots & \lambda_{n,a_n}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{2,a_2}^{r-1} & \lambda_{3,a_3}^{r-1} & \ldots & \lambda_{n,a_n}^{r-1} \end{bmatrix} \begin{bmatrix} \mu_{2,1}^{(a)} \\ \mu_{3,1}^{(a)} \\ \vdots \\ \mu_{n,1}^{(a)} \end{bmatrix}. \tag{3.11}$$

Define polynomials $p_0(x) = \prod_{u=0}^{s-1}(x - \lambda_{1,u})$, and $p_i(x) = x^i p_0(x)$ for $i = 0, 1, \ldots, r - s - 1$. We have proved the case of $d = n - 1$ in the previous section, so here we only consider the case when $d < n - 1$, and so $r - s - 1 \geq 0$. Since the degree of $p_i(x)$ is less than r for all $i = 0, 1, \ldots, r - s - 1$, we can write

$$p_i(x) = \sum_{j=0}^{r-1} p_{i,j} x^j.$$

Define the $(r - s) \times r$ matrix

$$P = \begin{bmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,r-1} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,r-1} \\ \vdots & \vdots & \vdots & \vdots \\ p_{r-s-1,0} & p_{r-s-1,1} & \cdots & p_{r-s-1,r-1} \end{bmatrix}.$$

Since

$$\begin{aligned} & P \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{1,0} & \lambda_{1,1} & \ldots & \lambda_{1,s-1} \\ \lambda_{1,0}^2 & \lambda_{1,1}^2 & \ldots & \lambda_{1,s-1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{1,0}^{r-1} & \lambda_{1,1}^{r-1} & \ldots & \lambda_{1,s-1}^{r-1} \end{bmatrix} \\ = & \begin{bmatrix} p_0(\lambda_{1,0}) & p_0(\lambda_{1,1}) & \ldots & p_0(\lambda_{1,s-1}) \\ p_1(\lambda_{1,0}) & p_1(\lambda_{1,1}) & \ldots & p_1(\lambda_{1,s-1}) \\ \vdots & \vdots & \vdots & \vdots \\ p_{r-s-1}(\lambda_{1,0}) & p_{r-s-1}(\lambda_{1,1}) & \ldots & p_{r-s-1}(\lambda_{1,s-1}) \end{bmatrix} = 0, \end{aligned}$$

together with (3.11), we have

$$P \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{2,a_2} & \lambda_{3,a_3} & \ldots & \lambda_{n,a_n} \\ \lambda_{2,a_2}^2 & \lambda_{3,a_3}^2 & \ldots & \lambda_{n,a_n}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{2,a_2}^{r-1} & \lambda_{3,a_3}^{r-1} & \ldots & \lambda_{n,a_n}^{r-1} \end{bmatrix} \begin{bmatrix} \mu_{2,1}^{(a)} \\ \mu_{3,1}^{(a)} \\ \vdots \\ \mu_{n,1}^{(a)} \end{bmatrix} = 0. \tag{3.12}$$

Note that

$$P \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \lambda_{2,a_2} & \lambda_{3,a_3} & \ldots & \lambda_{n,a_n} \\ \lambda_{2,a_2}^2 & \lambda_{3,a_3}^2 & \ldots & \lambda_{n,a_n}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{2,a_2}^{r-1} & \lambda_{3,a_3}^{r-1} & \ldots & \lambda_{n,a_n}^{r-1} \end{bmatrix} = \begin{bmatrix} p_0(\lambda_{2,a_2}) & p_0(\lambda_{3,a_3}) & \ldots & p_0(\lambda_{n,a_n}) \\ p_1(\lambda_{2,a_2}) & p_1(\lambda_{3,a_3}) & \ldots & p_1(\lambda_{n,a_n}) \\ \vdots & \vdots & \vdots & \vdots \\ p_{r-s-1}(\lambda_{2,a_2}) & p_{r-s-1}(\lambda_{3,a_3}) & \ldots & p_{r-s-1}(\lambda_{n,a_n}) \end{bmatrix}$$

$$= \begin{bmatrix} p_0(\lambda_{2,a_2}) & p_0(\lambda_{3,a_3}) & \dots & p_0(\lambda_{n,a_n}) \\ p_0(\lambda_{2,a_2})\lambda_{2,a_2} & p_0(\lambda_{3,a_3})\lambda_{3,a_3} & \dots & p_0(\lambda_{n,a_n})\lambda_{n,a_n} \\ \vdots & \vdots & \vdots & \vdots \\ p_0(\lambda_{2,a_2})\lambda_{2,a_2}^{r-s-1} & p_0(\lambda_{3,a_3})\lambda_{3,a_3}^{r-s-1} & \dots & p_0(\lambda_{n,a_n})\lambda_{n,a_n}^{r-s-1} \end{bmatrix}.$$

Moreover, $p_0(\lambda_{2,a_2}), p_0(\lambda_{3,a_3}), \dots, p_0(\lambda_{n,a_n})$ are all nonzero. Thus $(\mu_{2,1}^{(a)}, \mu_{3,1}^{(a)}, \dots, \mu_{n,1}^{(a)})$ forms a Generalized Reed-Solomon code of length $n-1$ and dimension d. According to Remark 3.3, given a nonnegative integer t such that $d+2t < n$, any $d+2t$ out of $n-1$ elements in $\{\mu_{2,1}^{(a)}, \mu_{3,1}^{(a)}, \dots, \mu_{n,1}^{(a)}\}$ suffice to recover the whole set as long as the number of erroneous elements in the $d+2t$ elements is not greater than t. Moreover, (3.11) implies that $\{c_{1,a(1,0)}, c_{1,a(1,1)}, \dots, c_{1,a(1,s-1)}\}$ can be determined by $\{\mu_{2,1}^{(a)}, \mu_{3,1}^{(a)}, \dots, \mu_{n,1}^{(a)}\}$. Consequently, we can recover C_1 by accessing any $d+2t$ surviving nodes and downloading the total of $(d+2t)l/s$ symbols of F from these nodes as long as the number of erroneous nodes among the helper nodes is not greater than t. This completes the proof. □

3.5 MDS array codes with the UER d-optimal repair property for several values of d simultaneously

In the previous two sections, we constructed MDS array codes with the UER d-optimal repair property for a single value of d. In this section we give a simple extension of the previous constructions to make the code have the UER d-optimal repair property for several values of d simultaneously. Let $n, k, m, d_1, d_2, \dots, d_m$ be any positive integers such that $k \le d_1, \dots, d_m < n$. We will show that by replacing s in Construction 3.2 with the value

$$s = \text{lcm}(d_1 + 1 - k, d_2 + 1 - k, \dots, d_m + 1 - k)$$

we obtain an $(n, k, l = s^n)$ MDS array code $\mathcal{C}$ with the UER d_i-optimal repair property for all $i = 1, \dots, m$ simultaneously.

By Theorem 3.3, $\mathcal{C}$ is an MDS array code. In the next theorem we establish results about the repair properties of the code $\mathcal{C}$.

Theorem 3.5. *The code $\mathcal{C}$ has the UER d_i-optimal repair property for any $i \in [m]$.*

Proof. Let $s_i = d_i + 1 - k$. Similarly to the proof of Theorem 3.4, we only prove the case of repairing C_1. Since $s_i | s$, we can partition the set $\{0, 1, \dots, s-1\}$ into s/s_i subsets $\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_{s/s_i}$, such that $|\mathcal{I}_j| = s_i$ for any $j \in [s/s_i]$, where $[s/s_i] = \{1, 2, \dots, s/s_i\}$. Following the proof of Theorem 3.4, we can show that for any $j \in [s/s_i]$, $a \in \{0, 1, \dots, l-1\}$, any nonnegative integer t such that $d_i + 2t < n$, and any $\mathcal{R} \subseteq \{2, 3, \dots, n\}$ of size $|\mathcal{R}| = d_i + 2t$, we can recover $\{c_{1,a(1,u)} : u \in \mathcal{I}_j\}$ by acquiring the set of values $\{\sum_{u \in \mathcal{I}_j} c_{v,a(1,u)} : v \in \mathcal{R}\}$ as long as the number of erroneous nodes in $\{C_i : i \in \mathcal{R}\}$ is not greater than t. Therefore, we can recover C_1 by accessing any $d_i + 2t$ surviving nodes and downloading the total of $(d_i + 2t)l/s_i$ symbols of F from these nodes as long as the number of erroneous nodes in the helper nodes is not greater than t. This completes the proof. □

Corollary 3.6. *The $(n, k, (n-k)^n)$ MDS array code given by Construction 3.1 has the UER d-optimal repair property if $(d+1-k)|(n-k)$.*

Example 3.1. *A $(k+4, k, 4^{k+4})$ MDS array code given by Construction 3.1 will automatically have the UER $(k+1)$-optimal repair property. A $(k+6, k, 6^{k+6})$ MDS array code given by Construction 3.1 has both the UER $(k+1)$-optimal repair property and the UER $(k+2)$-optimal repair property.*

3.6 Explicit MDS array codes with the UER (h, d)-optimal repair property for all $h \leq r$ and $k \leq d \leq n-h$ simultaneously

Given integers n and r, we construct a family of $(n, k = n-r, l)$ MDS array codes with the UER (h, d)-optimal repair property for all $h \leq r$ and $k \leq d \leq n-h$ simultaneously.

Construction 3.3. *Let F be a finite field of size $|F| \geq sn$, where $s = lcm(1, 2, \ldots, r)$. Let $\{\lambda_{i,j}\}_{i\in[n],j=0,1,\ldots,s-1}$ be sn distinct elements in F. Let $l = s^n$. Consider the code family given by (2.3)-(3.3), where the matrices A_i are given by*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_a^T, \ i = 1, \ldots, n. \tag{3.13}$$

Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F and a_i is defined in Construction 3.2.

Note that the difference between this construction and Construction 3.2 is in the choice of s. Define $a(i, u)$ and $c_{i,a}$ in the same way as in Sect. 3.3.

Clearly, the code $\mathcal{C}$ given by Construction 3.3 is an MDS array code.

Theorem 3.7. *The code $\mathcal{C}$ given by Construction 3.3 has the UER (h, d)-optimal repair property for any $h \leq r$ and $k \leq d \leq n-h$.*

Proof. By Theorem 3.5, $\mathcal{C}$ has the UER d-optimal repair property for any $k \leq d \leq n-1$. Now we show how to optimally repair h erasures. Without loss of generality, suppose that nodes $C_{\mathcal{F}} = \{C_1, C_2, \ldots, C_h\}$ are erased and we access nodes $C_{\mathcal{R}} = \{C_{h+1}, C_{h+2}, \ldots, C_{h+d+2t}\}$ to recover $C_{\mathcal{F}}$. Moreover, suppose that there are at most t erroneous nodes in $C_{\mathcal{R}}$.

To show the claim about the repair property of $\mathcal{C}$, we present a scheme that repairs the codes $C_1, C_2, \ldots, C_h$ one by one. More specifically, we first use $C_{\mathcal{R}}$ to repair C_1, then use $C_{\mathcal{R}} \cup C_1$ to repair C_2, then use $C_{\mathcal{R}} \cup C_1 \cup C_2$ to repair $C_3, \ldots$, and finally use $C_{\mathcal{R}} \cup C_1 \cup C_2 \cup \cdots \cup C_{h-1}$ to repair C_h. Let $s_i = i + d - k$.

When repairing C_i, we partition the set $\{0, 1, \ldots, s-1\}$ into s/s_i subsets $\mathcal{I}_1^{(i)}, \mathcal{I}_2^{(i)}, \ldots, \mathcal{I}_{s/s_i}^{(i)}$, where $\mathcal{I}_j^{(i)} = \{(j-1)s_i, (j-1)s_i + 1, \ldots, (j-1)s_i + s_i - 1\}$ for $j \in$

$[s/s_i]$. By the proof of Theorem 3.4, in order to recover C_i it suffices to know the values $\{\sum_{u\in\mathcal{I}_j^{(i)}} c_{v,a(i,u)} : a_i = 0, j \in [s/s_i], v \in \mathcal{R}\cup[i-1]\}$. Since we have already recovered $C_1,\ldots,C_{i-1}$, to recover C_i we only need to download the set of values

$$\Big\{ \sum_{u\in\mathcal{I}_j^{(i)}} c_{v,a(i,u)} : a_i = 0, j \in [s/s_i], v \in \mathcal{R}\Big\}$$

from $C_{\mathcal{R}}$. Thus, in order to recover $C_{\mathcal{F}}$, it suffices to know the values of elements in the set

$$\Lambda_h = \bigcup_{i=1}^{h} \Big\{ \sum_{u\in\mathcal{I}_j^{(i)}} c_{v,a(i,u)} : a_i = 0, j \in [s/s_i], v \in \mathcal{R}\Big\}.$$

In order to determine the values of these elements, we only need to download a spanning set for Λ_h over F from $C_{\mathcal{R}}$.

Define $\Omega_{i,v} = \{\sum_{u\in\mathcal{I}_j^{(i)}} c_{v,a(i,u)} : a_i = 0, j \in [s/s_i]\}$ for $i \in [h], v \in \mathcal{R}$ and $\Lambda_{1,v} = \Omega_{1,v}, \Lambda_{i,v} = \Lambda_{i-1,v} \cup \Omega_{i,v}$ for $i = 2,3,\ldots,h, v \in \mathcal{R}$. Given $a \in \{0,1,\ldots,l-1\}$ and $i \in [n]$, define the set $\Psi(a,i) = \{w : w \in [i-1], s_w | a_w\}$. For $i = 2,3,\ldots,h$, $q = 0,\ldots,i-1$ define

$$\Gamma_{i,v,q} = \Big\{ \sum_{u\in\mathcal{I}_j^{(i)}} c_{v,a(i,u)} : a_i = 0, |\Psi(a,i)| = q, j \in [s/s_i]\Big\}.$$

Let $B_{1,v} = \Omega_{1,v}, B_{i,v} = B_{i-1,v} \cup \Gamma_{i,v,0}, i = 2,3,\ldots,h$. We use induction on i to show that $\Lambda_{i,v} \subseteq \text{span}(B_{i,v})$ for every $i \in [h]$ and $v \in \mathcal{R}$. Clearly this claim holds for $i = 1$. Now suppose that it holds for $i = m$ and consider the case $i = m+1$. By the induction hypothesis, $\Lambda_{m,v} \subseteq \text{span}(B_{m,v})$, so it suffices to prove that

$$\Lambda_{m,v} \cup \Omega_{m+1,v} \subseteq \text{span}(\Lambda_{m,v} \cup \Gamma_{m+1,v,0}).$$

Note that $\Omega_{m+1,v} = \bigcup_{q=0}^{m} \Gamma_{m+1,v,q}$. Thus we only need to prove that $\Gamma_{m+1,v,q} \subseteq \text{span}(\Lambda_{m,v} \cup \Gamma_{m+1,v,0})$ for all $q = 0,1,\ldots,m$. We prove this claim by induction on q. This claim trivially holds for $q = 0$. Now suppose that it holds for some $q \geq 1$ and consider the case $q+1$. Given any a satisfying that $a_{m+1} = 0$ and $\Psi(a,m+1), |\Psi(a,m+1)| = q+1$, suppose that $w \in \Psi(a,m+1)$, namely, $s_w | a_w$, then $|\Psi(a(w,a_w+u),m+1)| = q$ for all $u \in [s_w - 1]$. By the induction hypothesis,

$$\sum_{u_2\in\mathcal{I}_j^{(m+1)}} c_{v,a(w,m+1,a_w+u_1,u_2)} \in \text{span}(\Lambda_{m,v} \cup \Gamma_{m+1,v,0})$$

for all $u_1 \in [s_w - 1]$ and $j \in [s/s_{m+1}]$, where $a(i_1,i_2,u_1,u_2)$ is obtained from a by replacing a_{i_1} with u_1 and a_{i_2} with u_2. Therefore,

$$\sum_{u_1=1}^{s_w-1} \sum_{u_2\in\mathcal{I}_j^{(m+1)}} c_{v,a(w,m+1,a_w+u_1,u_2)} \in \text{span}\,(\Lambda_{m,v} \cup \Gamma_{m+1,v,0}). \tag{3.14}$$

Note that

$$\sum_{u_1=0}^{s_w-1} c_{v,a(w,m+1,a_w+u_1,u_2)} \in \Omega_{w,v} \subseteq \Lambda_{m,v}$$

for any $u_2 \in \{0, 1, \ldots, s-1\}$. As a result,

$$\sum_{u_1=0}^{s_w-1} \sum_{u_2 \in \mathcal{I}_j^{(m+1)}} c_{v,a(w,m+1,a_w+u_1,u_2)} \in \text{span}(\Lambda_{m,v}). \tag{3.15}$$

Subtracting (3.14) from (3.15), we obtain

$$\sum_{u \in \mathcal{I}_j^{(m+1)}} c_{v,a(m+1,u)} \in \text{span}(\Lambda_{m,v} \cup \Gamma_{m+1,v,0})$$

for any $j \in [s/s_{m+1}]$. This establishes the induction step of the second induction and proves that $\Omega_{m+1,v} \subseteq \text{span}(\Lambda_{m,v} \cup \Gamma_{m+1,v,0})$. Therefore, $\Lambda_{m+1,v} \subseteq \text{span}(B_{m+1,v})$ for any $v \in \mathcal{R}$, and this completes the proof of the first induction.

Since $\Lambda_h = \bigcup_{v \in \mathcal{R}} \Lambda_{h,v}$ and $\Lambda_{h,v} \subseteq \text{span}(B_{h,v})$ for every $v \in \mathcal{R}$, to recover $C_{\mathcal{F}}$ we only need to download $\bigcup_{v \in \mathcal{R}} B_{h,v}$ from $C_{\mathcal{R}}$.

Finally, we find the size of the set $B_{h,v}$ for some fixed $v \in \mathcal{R}$. Since $B_{i,v} = B_{i-1,v} \cup \Gamma_{i,v,0}$, we have $|B_{i,v}| \leq |B_{i-1,v}| + |\Gamma_{i,v,0}|$. We will prove that $|B_{i,v}| \leq \frac{il}{s_i}$ by induction on i. By definition $|B_{1,v}| = |\Omega_{1,v}| = \frac{l}{s_1}$. Suppose that the claim holds for $i = m$ and consider the case $i = m+1$. It is easy to see that

$$\begin{aligned}
|\Gamma_{m+1,v,0}| &= \frac{s_1 - 1}{s_1} \frac{s_2 - 1}{s_2} \cdots \frac{s_m - 1}{s_m} \frac{l}{s_{m+1}} \\
&= \frac{d-k}{d+1-k} \frac{d+1-k}{d+2-k} \cdots \frac{d+m-1-k}{d+m-k} \frac{l}{d+m+1-k} \\
&= \frac{d-k}{d+m-k} \frac{l}{d+m+1-k}.
\end{aligned}$$

By the induction hypothesis,

$$\begin{aligned}
|B_{m+1,v}| &\leq |B_{m,v}| + |\Gamma_{m+1,v,0}| \\
&\leq \frac{ml}{d+m-k} + \frac{d-k}{d+m-k} \frac{l}{d+m+1-k} \\
&= \frac{(m+1)l}{d+m+1-k} = \frac{(m+1)l}{s_{m+1}}.
\end{aligned}$$

This establishes the induction step and proves that $|B_{h,v}| \leq \frac{hl}{d+h-k}$ for any $v \in \mathcal{R}$. We obtain

$$\Big| \bigcup_{v \in \mathcal{R}} B_{h,v} \Big| \leq \frac{hl|\mathcal{R}|}{d+h-k} = \frac{h(d+2t)l}{d+h-k}.$$

The proof is complete. □

Since we set $A_1, A_2, \ldots, A_n$ to be diagonal matrices in Construction 3.3, the encoding and repair processes involve only operations with $r \times r$ matrices over F, and the code $\mathcal{C}$ given by Construction 3.3 also has the optimal update property. Moreover, by the proof of Theorem 3.4 and Theorem 3.7, we can see that the repair process only requires operations with matrices of size no larger than $n \times n$.

3.7 Optimal-repair MDS array codes with optimal access property over small fields

In this section we construct an explicit family of MDS array codes with optimal access property. As above, we rely on the general construction (2.3)-(3.3) to construct an $(n, k = n - r, l = r^{n-1})$ array code. However in this section we take A_n to be the identity matrix and take $A_1, \ldots, A_{n-1}$ to be permutation matrices rather than diagonal matrices. This choice is beneficial in two ways: first, we are able to reduce the field size from rn in earlier construction to any field F of size $|F| \geq n + 1$, while also obtaining the optimal access property.

Construction 3.4. *Let F be a finite field of size $|F| \geq n + 1$ and let γ be its primitive element. Let $l = r^{n-1}$. Consider the code family given by (2.3)-(3.3), where the matrices $A_1, A_2, \ldots, A_n$ are given by*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_{a(i,a_i \oplus 1)}^T, \ i = 1, 2, ..., n-1,$$
$$A_n = I,$$

where $\oplus$ denotes addition modulo r, $\lambda_{i,0} = \gamma^i$ for all $i \in [n-1]$ and $\lambda_{i,u} = 1$ for all $i \in [n-1]$ and all $u \in \{1, 2, \ldots, r-1\}$. Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F, a_i is the i-th digit from the right in the representation of $a = (a_{n-1}, a_{n-2}, \ldots, a_1)$ in the r-ary form, and $a(i, u)$ is defined in the same way as in Sect. 3.3.

Remark 3.4. *Since $\prod_{u=0}^{r-1} \lambda_{i,u} = \gamma^i$, we have*

$$\prod_{u=0}^{r-1} \lambda_{i,u} \neq \prod_{u=0}^{r-1} \lambda_{j,u} \text{ for any } i, j \in [n-1], i \neq j.$$
$$\prod_{u=0}^{r-1} \lambda_{i,u} \neq 1 \text{ for any } i \in [n-1]. \tag{3.16}$$

It will be clear from the proofs below that the values of $\{\lambda_{i,u} : i \in [n], u = 0, 1, \ldots, r-1\}$ in Construction 3.4 can be assigned arbitrarily as long as (3.16) is satisfied.

Clearly, for $t = 0, 1, \ldots, r-1$ and $i \in [n-1]$, we have

$$A_i^t = \sum_{a=0}^{l-1} \beta_{i,a_i,t} e_a e_{a(i,a_i \oplus t)}^T,$$

where $\beta_{i,u,0} = 1$ and $\beta_{i,u,t} = \prod_{v=u}^{u\oplus(t-1)} \lambda_{i,v}$ for $t = 1, \ldots, r-1$ and $u = 0, 1, \ldots, r-1$.

Theorem 3.8. *The code $\mathcal{C}$ given by Construction 3.4 has the optimal access property.*

Proof. Let us write out the parity-check equations (2.3) coordinatewise:

$$c_{n,a} + \sum_{i=1}^{n-1} \beta_{i,a_i,t} c_{i,a(i,a_i\oplus t)} = 0 \text{ for all } t = 0, 1, \ldots, r-1 \text{ and } a = 0, \ldots, l-1, \tag{3.17}$$

where $c_{i,a}$ is defined in Sect. 3.3. First suppose we want to repair C_i for some $i \in [n-1]$. We will show that we only need to access the values in the set $\{c_{j,a} : a_i = 0\}$ from C_j for every $j \neq i$. Indeed, by (3.17), we have

$$\beta_{i,a_i,t} c_{i,a(i,a_i\oplus t)} = -c_{n,a} - \sum_{j\neq i,n} \beta_{j,a_j,t} c_{j,a(j,a_j\oplus t)}. \tag{3.18}$$

From (3.18) we see that the values $\{c_{i,a} : a_i = t\}$ can be determined by $\{c_{j,a} : j \neq i, a_i = 0\}$. Since (3.18) holds for every $t = 0, 1, \ldots, r-1$, we see that for $i \in [n-1]$, C_i can be determined by the values $\{c_{j,a} : j \neq i, a_i = 0\}$.

Now consider the case when the failed node is C_n. By (3.17), we know that the values in the set $\{c_{n,a} : a_1 \oplus a_2 \oplus \cdots \oplus a_{n-1} = r \ominus t\}$ can be determined by $\{c_{j,a} : j \neq n, a_1 \oplus a_2 \oplus \cdots \oplus a_{n-1} = 0\}$, where $\ominus$ denotes subtraction modulo r. Since (3.17) holds for every $t = 0, 1, \ldots, r-1$, we conclude that C_n can be determined by $\{c_{j,a} : j \neq n, a_1 \oplus a_2 \oplus \cdots \oplus a_{n-1} = 0\}$. This completes the proof. □

Our next task is to establish the MDS property of $\mathcal{C}$. The code $\mathcal{C}$ is MDS if and only if every $r \times r$ block submatrix of

$$\begin{bmatrix} I & I & \ldots & I \\ A_1 & A_2 & \ldots & A_n \\ \vdots & \vdots & \vdots & \vdots \\ A_1^{r-1} & A_2^{r-1} & \ldots & A_n^{r-1} \end{bmatrix}$$

is invertible. A criterion for this is given in the following lemma.

Lemma 3.9 (Block Vandermonde matrix)**.** *Let $B_1, \ldots, B_r$ be $l \times l$ matrices such that $B_i B_j = B_j B_i$ for all $i, j \in [r]$. The matrix*

$$M_r = \begin{bmatrix} I & I & \ldots & I \\ B_1 & B_2 & \ldots & B_r \\ \vdots & \vdots & \vdots & \vdots \\ B_1^{r-1} & B_2^{r-1} & \ldots & B_r^{r-1} \end{bmatrix}$$

is invertible if and only if $B_i - B_j$ is invertible for all $i \neq j$.

Proof. Suppose that $B_i - B_j$ is invertible for any $i, j \in [r], i \neq j$. Clearly the claim holds for $r = 2$. Now suppose that it holds for $r = s$. Consider the matrix

$$M_{s+1} = \begin{bmatrix} I & I & \dots & I \\ B_1 & B_2 & \dots & B_{s+1} \\ \vdots & \vdots & \vdots & \vdots \\ B_1^s & B_2^s & \dots & B_{s+1}^s \end{bmatrix}.$$

For $i = s, s-1, \dots, 1$, multiply the i-th "row" on the left by B_1 and subtract from the $(i+1)$-th row. Note that these operations do not change the rank of M_{s+1} since they leave its row space unchanged. Next, subtract the first column from all the other columns (this clears the first row without changing the column space and hence the rank of M_{s+1}) to obtain

$$M'_{s+1} = \begin{bmatrix} I & 0 & \dots & 0 \\ 0 & B_2 - B_1 & \dots & B_{s+1} - B_1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & (B_2 - B_1)B_2^{s-1} & \dots & (B_{s+1} - B_1)B_{s+1}^{s-1} \end{bmatrix}$$
$$= \begin{bmatrix} I & 0 & \dots & 0 \\ 0 & B_2 - B_1 & \dots & B_{s+1} - B_1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & B_2^{s-1}(B_2 - B_1) & \dots & B_{s+1}^{s-1}(B_{s+1} - B_1) \end{bmatrix}$$

(here we relied on the commuting condition $B_iB_j = B_jB_i$). Since the matrices $B_i - B_1, i = 2, \dots, s+1$ are invertible, we can multiply the i-th column on the right by $(B_i - B_1)^{-1}$ without changing the rank. We conclude that the matrix

$$M''_{s+1} = \begin{bmatrix} I & 0 & \dots & 0 \\ 0 & I & \dots & I \\ \vdots & \vdots & \vdots & \vdots \\ 0 & B_2^{s-1} & \dots & B_{s+1}^{s-1} \end{bmatrix}$$

has the same rank as M_{s+1}. By the induction hypothesis M''_{s+1} is invertible, and so is M_{s+1}. This completes the induction step.

Conversely, suppose that M_r is invertible. For $r = 2$ this implies that the matrix $B_1 - B_2$ is invertible. Now assume that the claim holds for $r = s$ and consider the case $r = s + 1$. Since M_{s+1} is invertible, M'_{s+1} is also invertible. Assume that $B_i - B_1$ is singular for some $i \in \{2, 3, \dots, s+1\}$, then

$$\begin{bmatrix} 0 \\ B_i - B_1 \\ \vdots \\ B_i^{s-1}(B_i - B_1) \end{bmatrix} = \begin{bmatrix} 0 \\ I \\ \vdots \\ B_i^{s-1} \end{bmatrix} (B_i - B_1)$$

contains linearly dependent columns, and thus M'_{s+1} is singular, contradiction. Thus $B_i - B_1$ is invertible for any $i \neq 1$. Consequently M''_{s+1} is invertible. By the induction assumption we conclude that $B_i - B_j$ is invertible for any $i, j \in [s+1], i \neq j$. □

Theorem 3.10. *The code $\mathcal{C}$ given by Construction 3.4 is an MDS array code.*

Proof. On account of Lemma 3.9, the claim will follow if we prove that for any $i \neq j$, $A_iA_j = A_jA_i$ and that the matrices $A_i - A_j$ are invertible. The matrix $A_n = I$, so we need to verify that for any $i, j \in [n-1], i \neq j$,

$$A_iA_j = A_jA_i = \sum_{a=0}^{l-1} \lambda_{i,a_i}\lambda_{j,a_j} e_a e^T_{a(i,j,a_i\oplus 1,a_j\oplus 1)},$$

where $a(i,j,u,v)$ is obtained from a by replacing a_i with u and a_j with v. This establishes the commuting part.

Now suppose that $A_ix = A_jx$ for some $i, j \in [n-1], i \neq j$ and some vector $x \in F^l$. Let $x = \sum_{a=0}^{l-1} x_a e_a$, where $x_a \in F$. Then

$$A_ix = \sum_{a=0}^{l-1} \lambda_{i,a_i} x_{a(i,a_i\oplus 1)} e_a,$$
$$A_jx = \sum_{a=0}^{l-1} \lambda_{j,a_j} x_{a(j,a_j\oplus 1)} e_a.$$

Therefore,

$$\lambda_{i,a_i} x_{a(i,a_i\oplus 1)} = \lambda_{j,a_j} x_{a(j,a_j\oplus 1)} \tag{3.19}$$

for every $a = 0, 1, \ldots, l-1$. Since $\lambda_{i,u} \neq 0$ for all $i \in [n-1]$ and all $u = 0, 1, \ldots, r-1$, we can write (3.19) as

$$x_a = \frac{\lambda_{j,a_j}}{\lambda_{i,a_i\ominus 1}} x_{a(i,j,a_i\ominus 1,a_j\oplus 1)}. \tag{3.20}$$

Repeating this step, we obtain

$$\begin{aligned} x_a &= \frac{\lambda_{j,a_j}}{\lambda_{i,a_i\ominus 1}} x_{a(i,j,a_i\ominus 1,a_j\oplus 1)} \\ &= \frac{\lambda_{j,a_j}}{\lambda_{i,a_i\ominus 1}} \frac{\lambda_{j,a_j\oplus 1}}{\lambda_{i,a_i\ominus 2}} x_{a(i,j,a_i\ominus 2,a_j\oplus 2)} \\ &= \cdots = \frac{\lambda_{j,a_j}}{\lambda_{i,a_i\ominus 1}} \frac{\lambda_{j,a_j\oplus 1}}{\lambda_{i,a_i\ominus 2}} \cdots \frac{\lambda_{j,a_j\oplus (r-1)}}{\lambda_{i,a_i\ominus r}} x_{a(i,j,a_i\ominus r,a_j\oplus r)} \\ &= \frac{\prod_{u=0}^{r-1} \lambda_{j,u}}{\prod_{u=0}^{r-1} \lambda_{i,u}} x_a \end{aligned}$$

for every $a = 0, 1, \ldots, l-1$. By (3.16), $x_a = 0$ for all $a = 0, \ldots, l-1$. Thus $(A_i - A_j)x = 0$ implies that $x = 0$. We conclude that the matrices $A_i - A_j$ are invertible for all $i, j \in [n-1], i \neq j$.

Now suppose that $A_i x = A_n x = x$ for some $i \in [n-1]$ and some vector $x \in F^l$. Then we have

$$\lambda_{i,a_i} x_{a(i,a_i \oplus 1)} = x_a. \tag{3.21}$$

Thus

$$\begin{aligned} x_a &= \lambda_{i,a_i} x_{a(i,a_i \oplus 1)} = \lambda_{i,a_i} \lambda_{i,a_i \oplus 1} x_{a(i,a_i \oplus 2)} \\ &= \cdots = \lambda_{i,a_i} \lambda_{i,a_i \oplus 1} \cdots \lambda_{i,a_i \oplus (r-1)} x_{a(i,a_i \oplus r)} \\ &= \Big(\prod_{u=0}^{r-1} \lambda_{i,u} \Big) x_a. \end{aligned}$$

By (3.16), $x_a = 0$ for all $a = 0, \ldots, l-1$. So $x = 0$ and $\ker(A_i - A_n) = 0$, or, in other words the matrix $A_i - A_n$ is invertible for all $i \in [n-1]$. This completes the proof. □

3.7.1 Complexity of encoding and decoding

The encoding and decoding procedures solve the same problem, namely, determining the values of r nodes from the known values of k nodes. Without loss of generality, suppose that we want to determine $C_{k+1}, \ldots, C_{k+r}$ from $C_1, C_2, \ldots, C_k$. Note that (3.17) contains rl equations and we have rl unknown elements here. Since the code is MDS, the unknown values are uniquely determined.

Now let us show that instead of inverting an $rl \times rl$ matrix, we only need to invert matrices of size $r^{r+1} \times r^{r+1}$. Observe that, given any $b_1, b_2, \ldots, b_k \in \{0, 1, \ldots, r-1\}$, the r^{r+1} unknown elements $\{c_{i,a} : i = k+1, k+2, \ldots, k+r, a_1 = b_1, a_2 = b_2, \ldots, a_k = b_k\}$ appear in exactly r^{r+1} equations in (3.17), and these r^{r+1} equations only contain these r^{r+1} unknown elements. For this reason, we can find these r^{r+1} unknown elements by inverting an $r^{r+1} \times r^{r+1}$ matrix.

3.8 Explicit MDS array codes with the UER d-optimal access property

Construction 3.5. *Let F be a finite field of size $|F| \geq n+1$ and let γ be a primitive element in F. Let $s = d+k-1$ and $l = s^n$. Consider the code given by (2.3)-(3.3), where the matrices $A_1, A_2, \ldots, A_n$ are given by*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_{a(i,a_i \oplus 1)}^T, \ i = 1, \ldots, n,$$

where $\oplus$ denotes addition modulo s, $\lambda_{i,0} = \gamma^i$ for all $i \in [n]$ and $\lambda_{i,u} = 1$ for all $i \in [n]$ and all $u \in [s-1]$. Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F, a_i is the i-th digit from the right in the representation of $a = (a_n, a_{n-1}, \ldots, a_1)$ in the s-ary form, and $a(i,u)$ is defined in the same way as in Sect. 3.3.

Theorem 3.11. *The code $\mathcal{C}$ given by Construction 3.5 is an MDS array code.*

Proof. Paralleling the proof of Theorem 3.10, we can show that for any $i \neq j$, $A_iA_j = A_jA_i$ and that the matrix $A_i - A_j$ is invertible. Thus $\mathcal{C}$ is an MDS array code. □

Theorem 3.12. *The code $\mathcal{C}$ given by Construction 3.5 has the UER d-optimal access property.*

Proof. Suppose we want to repair C_i. By an argument similar to the proof of Theorem 3.8, we only need to know the values $\{c_{j,a} : a_i = 0\}$ from C_j for every $j \neq i$. Define a function $g : \{0, 1, \ldots, l/s-1\} \to \{0, 1, \ldots, l-1\}$ as $g(a) = (a_{n-1}, a_{n-2}, \ldots, a_i, 0, a_{i-1}, a_{i-2}, \ldots, a_1)$, where a is an element in $\{0, 1, \ldots, l/s - 1\}$ with the s-ary expansion $(0, a_{n-1}, a_{n-2}, \ldots, a_1)$. Define the column vector $C_j^{(i)} \in F^{l/s}$ as $C_j^{(i)} = (c_{j,g(0)}, c_{j,g(1)}, \ldots, c_{j,g(l/s-1)})^T$ for all $j \neq i$. In order to prove the theorem, we only need to prove that $(C_1^{(i)}, C_2^{(i)}, \ldots, C_{i-1}^{(i)}, C_{i+1}^{(i)}, C_{i+2}^{(i)}, \ldots, C_n^{(i)})$ forms an $(n-1, d, s^{n-1})$ MDS array code.

Notice that $A_j^s = \sum_{a=0}^{l-1}(\prod_{u=0}^{s-1} \lambda_{j,u}) e_a e_{a(j,a_j \oplus s)}^T = \gamma^j I$ for all $j \in [n]$. Note also that

$$\sum_{j=1}^{n} A_j^m C_j = 0, \quad \sum_{j=1}^{n} A_j^{m+s} C_j = 0$$

for all $m = 0, 1, \ldots, r-s-1$. Multiplying the first equation by γ^i and then subtracting it from the second one, we obtain

$$\sum_{j \neq i} (\gamma^j - \gamma^i) A_j^m C_j = 0, \quad m = 0, 1, \ldots, r-s-1. \tag{3.22}$$

Let $e_0^{(l/s)}, e_1^{(l/s)}, \ldots, e_{l/s-1}^{(l/s)}$ be the standard basis vectors of $F^{l/s}$ over F. Define $l/s \times l/s$ matrix

$$B_j = \sum_{a=0}^{l/s-1} \lambda_{j,a_j} e_a^{(l/s)} (e_{a(j,a_j \oplus 1)}^{(l/s)})^T \text{ for } j \in [i-1],$$

$$B_j = \sum_{a=0}^{l/s-1} \lambda_{j+1,a_j} e_a^{(l/s)} (e_{a(j,a_j \oplus 1)}^{(l/s)})^T \text{ for } i \leq j < n.$$

It is easy to see that (3.22) implies

$$\sum_{j=1}^{i-1} (\gamma^j - \gamma^i) B_j^m C_j^{(i)} + \sum_{j=i}^{n-1} (\gamma^{j+1} - \gamma^i) B_j^m C_{j+1}^{(i)} = 0$$
$$\text{for all } m = 0, 1, \ldots, r-s-1.$$

As in the proof of Theorem 3.10, we can show that for any $j_1, j_2 \in [n-1], j_1 \neq j_2$, $B_{j_1}B_{j_2} = B_{j_2}B_{j_1}$ and that the matrix $B_{j_1} - B_{j_2}$ is invertible. Moreover, $r - s = n - 1 - d$. Thus

$$\begin{bmatrix} I & I & \ldots & I \\ B_1 & B_2 & \ldots & B_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ B_1^{r-s-1} & B_2^{r-s-1} & \ldots & B_{n-1}^{r-s-1} \end{bmatrix}$$

is a parity-check matrix of an $(n-1, d, s^{n-1})$ MDS array code. Multiplying each block column with a nonzero constant does not change the MDS property. As a result, the set of vectors $(C_1^{(i)}, C_2^{(i)}, \ldots, C_{i-1}^{(i)}, C_{i+1}^{(i)}, C_{i+2}^{(i)}, \ldots, C_n^{(i)})$ forms an $(n-1, d, s^{n-1})$ MDS array code. Therefore, if we access any $d+2t$ out of $n-1$ vectors in the set $(C_1^{(i)}, C_2^{(i)}, \ldots, C_{i-1}^{(i)}, C_{i+1}^{(i)}, C_{i+2}^{(i)}, \ldots, C_n^{(i)})$, we will be able to recover the whole set and further recover C_i as long as the number of erroneous nodes among the helper nodes is not greater than t. This completes the proof. □

3.9 An MDS array code family with the UER d-optimal access property for several values of d simultaneously

In this section we present a simple extension of the code family in Construction 3.5 which gives MDS array codes with the UER d-optimal access property for several values of d simultaneously. More specifically, given any positive integers $n, k, m, d_1, d_2, \ldots, d_m$ such that $k \leq d_1, \ldots, d_m < n$, we will show that by replacing s in Construction 3.5 with the value

$$s = \text{lcm}(d_1 + 1 - k, d_2 + 1 - k, \ldots, d_m + 1 - k)$$

we obtain an $(n, k, l = s^n)$ MDS array code $\mathcal{C}$ with the UER d_i-optimal access property for all $i = 1, \ldots, m$ simultaneously.

On account of Theorem 3.10, we already know that $\mathcal{C}$ is an MDS array code. It remains to show that it has the UER d_i-optimal access property for any $i \in [m]$.

Theorem 3.13. *The code $\mathcal{C}$ has the UER d_i-optimal access property for any $i \in [m]$.*

Proof. Given any $i \in [m]$, we show that $\mathcal{C}$ has the UER d_i-optimal access property. Let $s_i = d_i + 1 - k$.

Without loss of generality, we only consider the case of repairing C_n. By an argument similar to the proof of Theorem 3.8, we only need to know the values $\{c_{j,a} : a_n = 0, s_i, 2s_i, \ldots, (s/s_i - 1)s_i\}$ from C_j for every $j \in [n-1]$. Define a function

$$\begin{aligned} g : \{0, 1, \ldots, l/s_i - 1\} &\to \{0, 1, \ldots, l-1\} \\ a &\mapsto (s_i a_n, a_{n-1}, a_{n-2}, \ldots, a_1), \end{aligned}$$

where a is an element in $\{0, 1, \ldots, l/s_i - 1\}$ with s-ary expansion $(a_n, a_{n-1}, a_{n-2}, \ldots, a_1)$. Define the column vector $C_j^{(n)} \in F^{l/s_i}$ as $C_j^{(n)} = (c_{j,g(0)}, c_{j,g(1)}, \ldots, c_{j,g(l/s_i-1)})^T$ for all $j \in [n-1]$. As mentioned above, $\{C_j^{(n)} : j \in [n-1]\}$ contains the information we need to recover C_n. Let us prove that $(C_1^{(n)}, C_2^{(n)}, \ldots, C_{n-1}^{(n)})$ forms an $(n-1, d, l/s_i)$ MDS array code.

Observe that

$$\sum_{j=1}^{n} A_j^m C_j = 0, \quad \sum_{j=1}^{n} A_j^{m+s_i} C_j = 0$$

for all $m = 0, 1, \ldots, r - s_i - 1$. Multiplying the first equation on the left by $A_n^{s_i}$ and then subtracting it from the second one, we obtain

$$\sum_{j=1}^{n-1}(A_j^{s_i} - A_n^{s_i})A_j^m C_j = 0, \quad m = 0, 1, \ldots, r - s_i - 1. \tag{3.23}$$

Let $e_0^{(l/s_i)}, e_1^{(l/s_i)}, \ldots, e_{l/s_i-1}^{(l/s_i)}$ be the standard basis vectors of F^{l/s_i} over F. Define $l/s_i \times l/s_i$ matrices

$$B_j = \sum_{a=0}^{l/s_i-1} \lambda_{j,a_j} e_a^{(l/s_i)}(e_{a(j,a_j\oplus 1)}^{(l/s_i)})^T \text{ for } j \in [n-1],$$
$$B_n = \sum_{a=0}^{l/s_i-1} \Big(\prod_{q=s_i a_n}^{s_i a_n + s_i - 1} \lambda_{n,q} \Big) e_a^{(l/s_i)}(e_{a(n,(s_i a_n \oplus s_i)/s_i)}^{(l/s_i)})^T.$$

It is easy to see that (3.23) implies the equality

$$\sum_{j=1}^{n-1}(B_j^{s_i} - B_n)B_j^m C_j^{(n)} = 0, \quad m = 0, 1, \ldots, r - s_i - 1.$$

Now suppose that $B_j^{s_i} x = B_n x$ for some $j \in [n-1]$ and some vector $x \in F^{l/s_i}$. Let $x = \sum_{a=0}^{l/s_i-1} x_a e_a^{(l/s_i)}$, where $x_a \in F$. Then

$$B_j^{s_i} x = \sum_{a=0}^{l/s_i-1} \Big(\prod_{q=a_j}^{a_j \oplus (s_i-1)} \lambda_{j,q} \Big) x_{a(j,a_j\oplus s_i)} e_a^{(l/s_i)},$$
$$B_n x = \sum_{a=0}^{l/s_i-1} \Big(\prod_{q=s_i a_n}^{s_i a_n + s_i - 1} \lambda_{n,q} \Big) x_{a(n,(s_i a_n \oplus s_i)/s_i)} e_a^{(l/s_i)}.$$

Therefore,

$$\Big(\prod_{q=a_j}^{a_j \oplus (s_i-1)} \lambda_{j,q} \Big) x_{a(j,a_j\oplus s_i)} = \Big(\prod_{q=s_i a_n}^{s_i a_n + s_i - 1} \lambda_{n,q} \Big) x_{a(n,(s_i a_n \oplus s_i)/s_i)} \tag{3.24}$$

for every $a = 0, 1, \ldots, l/s_i - 1$. Let us rewrite (3.24) as

$$x_a = \frac{\prod_{q=s_i a_n}^{s_i a_n + s_i - 1} \lambda_{n,q}}{\prod_{q=a_j \ominus s_i}^{a_j \ominus 1} \lambda_{j,q}} x_{a(j,n,a_j \ominus s_i, (s_i a_n \oplus s_i)/s_i)}.$$

Repeating this step, we obtain

$$x_a = \frac{\prod_{q=s_i a_n}^{s_i a_n \oplus (s-1)} \lambda_{n,q}}{\prod_{q=a_j \ominus s}^{a_j \ominus 1} \lambda_{j,q}} x_{a(j,n,a_j \ominus s, (s_i a_n \oplus s)/s_i)}$$

$$= \frac{\prod_{u=0}^{s-1} \lambda_{n,u}}{\prod_{u=0}^{s-1} \lambda_{j,u}} x_a$$
$$= \frac{\gamma^n}{\gamma^j} x_a$$

for every $a = 0, 1, \ldots, l-1$. Since $\gamma^j \neq \gamma^n$ for any $j \in [n-1]$, we have $x_a = 0$ for all $a = 0, \ldots, l-1$. Thus $(B_j^{s_i} - B_n)x = 0$ implies that $x = 0$. We conclude that the matrix $B_j^{s_i} - B_n$ is invertible for all $j \in [n-1]$.

Following the proof of Theorem 3.10, we can show that

$$\begin{bmatrix} I & I & \ldots & I \\ B_1 & B_2 & \ldots & B_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ B_1^{r-s_i-1} & B_2^{r-s_i-1} & \ldots & B_{n-1}^{r-s_i-1} \end{bmatrix}$$

is the parity-check matrix of an $(n-1, d_i, l/s_i)$ MDS array code. Multiplying each block column with an invertible matrix does not change the MDS property. As a result, $(C_1^{(n)}, C_2^{(n)}, \ldots, C_{n-1}^{(n)})$ forms an $(n-1, d_i, l/s_i)$ MDS array code. Therefore, if we access any $d_i + 2t$ out of $n-1$ vectors in the set $(C_1^{(n)}, C_2^{(n)}, \ldots, C_{n-1}^{(n)})$, we will be able to recover the whole set and further recover C_n as long as the number of erroneous nodes among the helper nodes is not greater than t. This completes the proof. □

3.10 Explicit MDS array codes with the UER (h, d)-optimal access property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously

Given integers n and r, we construct a family of $(n, k = n - r, l)$ MDS array codes with the UER (h, d)-optimal access property for all $h \leq r$ and $k \leq d \leq n - h$ simultaneously.

Construction 3.6. *Let F be a finite field of size $|F| \geq n + 1$ and let γ be its primitive element. Let $s = lcm(1, 2, \ldots, r)$ and $l = s^n$. Consider the code family given by (2.3)-(3.3), where the matrices A_i are given by*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_{a(i,a_i \oplus 1)}^T, \ i = 1, \ldots, n,$$

where $\oplus$ denotes addition modulo s, $\lambda_{i,0} = \gamma^i$ for all $i \in [n]$ and $\lambda_{i,u} = 1$ for all $i \in [n]$ and all $u \in [s-1]$. Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F, a_i is the i-th digit from the right in the representation of $a = (a_n, a_{n-1}, \ldots, a_1)$ in the s-ary form, and $a(i, u)$ is defined in the same way as in Sect. 3.3.

Note that the difference between this construction and Construction 3.5 is in the choice of s.

Clearly, the code $\mathcal{C}$ given by Construction 3.6 is an MDS array code.

Theorem 3.14. *The code $\mathcal{C}$ given by Construction 3.6 has the UER (h,d)-optimal access property for all $h \leq r$ and $k \leq d \leq n-h$ simultaneously.*

Proof. By Theorem 3.13, $\mathcal{C}$ has the UER d-optimal access property for any $k \leq d \leq n-1$. Now we show how to optimally repair h erasures. Without loss of generality, suppose that nodes $C_{\mathcal{F}} = \{C_1, C_2, \ldots, C_h\}$ are erased and we access nodes $C_{\mathcal{R}} = \{C_{h+1}, C_{h+2}, \ldots, C_{h+d+2t}\}$ to recover $C_{\mathcal{F}}$. Moreover, suppose that there are at most t erroneous nodes in $C_{\mathcal{R}}$.

As before, we repair $C_1, C_2, \ldots, C_h$ one by one. More specifically, we first use $C_{\mathcal{R}}$ to repair C_1, then $C_{\mathcal{R}} \cup C_1$ to repair C_2, then $C_{\mathcal{R}} \cup C_1 \cup C_2$ to repair $C_3, \ldots$, and finally we use $C_{\mathcal{R}} \cup C_1 \cup C_2 \cup \cdots \cup C_{h-1}$ to repair C_h. Let $s_i = i + d - k$. When repairing C_i, according to the proof of Theorem 3.13, we only need to know the values $\{c_{v,a} : a_i = 0, s_i, 2s_i, \ldots, (s/s_i - 1)s_i, v \in \mathcal{R} \bigcup [i-1]\}$. Since we have already recovered $C_1, \ldots, C_{i-1}$, we need to access only the values $\{c_{v,a} : a_i = 0, s_i, 2s_i, \ldots, (s/s_i - 1)s_i, v \in \mathcal{R}\}$ from $C_{\mathcal{R}}$ to recover C_i. Thus in order to recover $C_{\mathcal{F}}$, we need to access the set of elements $\Lambda_h = \bigcup_{i=1}^{h} \{c_{v,a} : a_i = 0, s_i, 2s_i, \ldots, (s/s_i - 1)s_i, v \in \mathcal{R}\}$.

Consider the set

$$\Omega_{j,v} = \{c_{v,a} : a_j = 0, s_j, 2s_j, \ldots, (s/s_j - 1)s_j\}$$

for $j \in [h]$ and $v \in \mathcal{R}$. Let $\Lambda_{1,v} = \Omega_{1,v}, \Lambda_{j+1,v} = \Lambda_{j,v} \cup \Omega_{j+1,v}, j = 1, \ldots, h-1$. We prove by induction on j that $|\Lambda_{j,v}| = \frac{jl}{j+d-k}$. Clearly this is true for $j = 1$. Suppose that the claim is true for $j = m$ and consider the case $j = m+1$. By definition, we have $|\Lambda_{m+1,v}| = |\Lambda_{m,v}| + |\Omega_{m+1,v}| - |\Lambda_{m,v} \cap \Omega_{m+1,v}|$. By the induction hypothesis, $|\Lambda_{m,v}| = \frac{ml}{m+d-k}$. Therefore,

$$|\Lambda_{m,v} \bigcap \Omega_{m+1,v}| = \frac{ml}{m+d-k} \frac{1}{m+1+d-k}.$$

Thus

$$\begin{aligned} |\Lambda_{m+1,v}| &= \frac{ml}{m+d-k} + \frac{l}{m+1+d-k} - \frac{ml}{m+d-k} \frac{1}{m+1+d-k} \\ &= \frac{(m+1)l}{m+1+d-k}. \end{aligned}$$

This concludes the induction step and proves that $|\Lambda_{h,v}| = \frac{hl}{d+h-k}$ for any $v \in \mathcal{R}$. Thus $|\Lambda_h| = |\mathcal{R}||\Lambda_{h,v}| = \frac{h(d+2t)l}{d+h-k}$. The proof is complete. □

3.11 Generalized Reed-Solomon Array codes and d-optimal repair property

In this section we construct a family of MDS array codes with the d-optimal repair property that requires a smaller underlying field size compared to Construction 3.2 and a smaller l compared to both Construction 3.2 and Construction 3.5. The construction forms an extension of Construction 3.4. As a first step, we introduce a new class of MDS array codes.

3.11.1 Generalized Reed-Solomon Array Codes

Definition 3.2. *Let $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ be a set of $l \times l$ matrices over F such that $A_iA_j = A_jA_i$ for all $i, j \in [n]$ and the matrices $A_i - A_j$ are invertible for all $i \neq j$. Let $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$ be a set of $l \times l$ invertible matrices with entries in F such that $A_iV_j = V_jA_i$ for any $i, j \in [n]$. A* Generalized Reed-Solomon array code $\text{GRSA}(n, k, \mathcal{A}, \mathcal{V})$ *is defined as the (n, k, l) array code with the generator matrix*

$$G = \begin{bmatrix} V_1 & V_2 & \ldots & V_n \\ A_1V_1 & A_2V_2 & \ldots & A_nV_n \\ A_1^2V_1 & A_2^2V_2 & \ldots & A_n^2V_n \\ \vdots & \vdots & \vdots & \vdots \\ A_1^{k-1}V_1 & A_2^{k-1}V_2 & \ldots & A_n^{k-1}V_n \end{bmatrix}.$$

More specifically,

$$\begin{aligned} \text{GRSA}(n, k, \mathcal{A}, \mathcal{V}) = \{(C_1, C_2, \ldots, C_n) : [C_1^T C_2^T \ldots C_n^T] \\ = [M_1^T M_2^T \ldots M_k^T]G \text{ for some } M_1, \ldots, M_k \in F^l\}. \end{aligned} \tag{3.25}$$

If $V_1 = V_2 = \cdots = V_n = I$, then we call this code a Reed-Solomon array code and denote it as $\text{RSA}(n, k, \mathcal{A})$.

Lemma 3.9 implies that GRSA codes have the MDS property. We need a description of its dual code, which is analogous to the scalar case (Theorem 10.4 in [35, p.304]).

Theorem 3.15. *Given a Generalized Reed-Solomon array code* $\text{GRSA}(n, k = n-r, \mathcal{A}, \mathcal{V})$ *with $\mathcal{A}$ and $\mathcal{V}$ satisfying the conditions in Def. 3.2, there is a set $\mathcal{W} = \{W_1, W_2, \ldots, W_n\}$ of $l \times l$ invertible matrices such that $A_iW_j = W_jA_i$ for any $i, j \in [n]$, and*

$$\text{GRSA}(n, k, \mathcal{A}, \mathcal{V}) = \{(C_1, C_2, \ldots, C_n) : H[C_1^T C_2^T \ldots C_n^T]^T = 0\}, \tag{3.26}$$

where

$$H = \begin{bmatrix} W_1 & W_2 & \ldots & W_n \\ A_1W_1 & A_2W_2 & \ldots & A_nW_n \\ A_1^2W_1 & A_2^2W_2 & \ldots & A_n^2W_n \\ \vdots & \vdots & \vdots & \vdots \\ A_1^{r-1}W_1 & A_2^{r-1}W_2 & \ldots & A_n^{r-1}W_n \end{bmatrix}.$$

This theorem follows from the following lemma.

Lemma 3.16. *Let $A_1, A_2, \ldots, A_n$ be $l \times l$ matrices with entries in F such that $A_iA_j = A_jA_i$ for all $i, j \in [n]$ and the matrices $A_i - A_j$ are invertible for all $i \neq j$. The following equation holds:*

$$\begin{bmatrix} I & I & \ldots & I \\ A_1 & A_2 & \ldots & A_n \\ A_1^2 & A_2^2 & \ldots & A_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ A_1^{n-2} & A_2^{n-2} & \ldots & A_n^{n-2} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = 0, \tag{3.27}$$

where $B_i = (\prod_{j \neq i}(A_j - A_i))^{-1}, i = 1, \ldots, n$

Proof. Note that the theory of determinants as well as Cramer's rule extend with no extra effort over arbitrary commutative rings with identity [13, Sect. 11.4]. Let R be a commutative ring containing $I, A_1, A_2, \ldots, A_n$. Given a square block matrix partitioned into $l \times l$ square submatrices in R, we view it as a square matrix with entries in R and define the determinant accordingly. Clearly, for any $i \in [n]$ and any nonnegative integer t, $A_i^t \in R$. We use Cramer's rule to solve the following equation

$$\begin{bmatrix} I & I & \ldots & I \\ A_1 & A_2 & \ldots & A_{n-1} \\ A_1^2 & A_2^2 & \ldots & A_{n-1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ A_1^{n-2} & A_2^{n-2} & \ldots & A_{n-1}^{n-2} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_{n-1} \end{bmatrix} = - \begin{bmatrix} I \\ A_n \\ A_n^2 \\ \vdots \\ A_n^{n-2} \end{bmatrix}$$

The expression for the Vandermonde determinant also holds in commutative rings, and we can easily find that $X_i = (\prod_{j \neq n}(A_j - A_n))(\prod_{j \neq i}(A_j - A_i))^{-1}$. Moving the right-hand side of the equation above to the left and then multiplying on the right by B_n, we obtain (3.27). □

Proof of Theorem 3.15. Set $W_i = V_i^{-1}(\prod_{j \neq i}(A_j - A_i))^{-1}$. Notice the fact that if an $l \times l$ matrix A and an $l \times l$ invertible matrix B satisfy $AB = BA$, then $AB^{-1} = B^{-1}BAB^{-1} = B^{-1}ABB^{-1} = B^{-1}A$. Thus $W_i A_j = A_j W_i$ for any $i, j \in [n]$.

Denote the codes defined in (3.25) and (3.26) as $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively. By (3.27), $HG^T = 0$. Thus $\mathcal{C}_1 \subseteq \mathcal{C}_2$. Since G and H both have full rank, $\mathcal{C}_1$ and $\mathcal{C}_2$ have the same dimension kl as a vector space over F. Consequently $\mathcal{C}_1 = \mathcal{C}_2$. This completes the proof. □

3.11.2 A family of MDS array codes with the d-optimal repair property

In this section we use the results about GRSA codes to construct a family of $(n, k = n - r, l)$ MDS array codes with the d-optimal repair property.

Construction 3.7. *Let F be a finite field of size $|F| \geq n + 1$ and let γ be a primitive element in F. Let $s = d + 1 - k$ and $l = s^{n-1}$. Consider the code family given by (2.3)-(3.3), where the matrices $A_1, A_2, \ldots, A_n$ are given by*

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_{a(i,a_i \oplus 1)}^T, \ i = 1, 2, ..., n-1,$$

$$A_n = I,$$

where $\oplus$ denotes addition modulo s, $\lambda_{i,0} = \gamma^i$ for all $i \in [n-1]$ and $\lambda_{i,u} = 1$ for all $i \in [n-1]$ and all $u \in \{1, 2, \ldots, s-1\}$. Here $\{e_a : a = 0, 1, \ldots, l-1\}$ is the standard basis of F^l over F, a_i is the i-th digit from the right in the representation of $a = (a_{n-1}, a_{n-2}, \ldots, a_1)$ in the s-ary form, and $a(i, u)$ is defined in the same way as in Sect. 3.3.

Theorem 3.17. *The code $\mathcal{C}$ given by Construction 3.7 is an MDS array code.*

Proof. The proof parallels the proof of Theorem 3.10: we show that for any $i \neq j$, $A_iA_j = A_jA_i$ and that the matrix $A_i - A_j$ is invertible. Thus, $\mathcal{C}$ is an MDS array code. □

Theorem 3.18. *The code $\mathcal{C}$ given by Construction 3.7 has the d-optimal repair property.*

Proof. By Theorem 3.15, $\mathcal{C} = \text{GRSA}(n, k, \mathcal{A}, \mathcal{V})$ with $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ and $\mathcal{V} = \{V_1, \ldots, V_n\}$, where $V_i = (\prod_{j \neq i}(A_j - A_i))^{-1}$. Suppose that we want to use $C_{i_2}, C_{i_3}, \ldots, C_{i_{d+1}}$ to repair C_{i_1}. Let $\Delta = \{i_1, i_2, \ldots, i_{d+1}\}$. Clearly $(C_{i_1}, C_{i_2}, \ldots, C_{i_{d+1}})$ is a Generalized Reed-Solomon Array code $\text{GRSA}(d+1, k, \mathcal{A}_\Delta, \mathcal{V}_\Delta)$, where $\mathcal{A}_\Delta = \{A_{i_1}, A_{i_2}, \ldots, A_{i_{d+1}}\}$ and $\mathcal{V}_\Delta = \{V_{i_1}, V_{i_2}, \ldots, V_{i_{d+1}}\}$. By Theorem 3.15, there is a set of $l \times l$ invertible matrices $\mathcal{W}_\Delta = \{W_1, W_2, \ldots, W_{d+1}\}$ such that

$$\begin{bmatrix} I & I & \ldots & I \\ A_{i_1} & A_{i_2} & \ldots & A_{i_{d+1}} \\ A_{i_1}^2 & A_{i_2}^2 & \ldots & A_{i_{d+1}}^2 \\ \vdots & \vdots & \vdots & \vdots \\ A_{i_1}^{s-1} & A_{i_2}^{s-1} & \ldots & A_{i_{d+1}}^{s-1} \end{bmatrix} \begin{bmatrix} W_1 C_{i_1} \\ W_2 C_{i_2} \\ \vdots \\ W_{d+1} C_{i_{d+1}} \end{bmatrix} = 0, \tag{3.28}$$

Using the same method as in the proof of Theorem 3.8, we can show that $W_1 C_{i_1}$ can be recovered by downloading a vector in $F^{l/s}$ from each of the nodes $C_{i_2}, C_{i_3}, \ldots, C_{i_{d+1}}$. Since W_1 is invertible, C_{i_1} can be recovered by the same set of vectors. This shows that $\mathcal{C}$ has the d-optimal repair property. □

3.11.3 Extension to d-optimal repair property for several values of d simultaneously

Now we give a simple extension of the previous construction to make the code have d-optimal repair property for several values of d simultaneously. More specifically, give any positive integers $n, k, m, d_1, d_2, \ldots, d_m$ such that $k \leq d_1, \ldots, d_m < n$, we will show that by replacing s in Construction 3.7 with the value

$$s = \text{lcm}(d_1 + 1 - k, d_2 + 1 - k, \ldots, d_m + 1 - k),$$

we will obtain an $(n, k, l = s^{n-1})$ MDS array code $\mathcal{C}$ with d_i-optimal repair property for all $i = 1, \ldots, m$ simultaneously.

By the proof of Theorem 3.10, we know that $\mathcal{C}$ is an MDS array code. Now we prove that $\mathcal{C}$ has d_i-optimal repair property for any $i \in [m]$.

Theorem 3.19. *The code $\mathcal{C}$ has d_i-optimal repair property for any $i \in [m]$.*

Proof. Given any $i \in [m]$, we show that $\mathcal{C}$ has d_i-optimal repair property. Let $s_i = d_i + 1 - k$. Without loss of generality, we only prove the case when we use $C_2, C_3, \ldots, C_{d_i+1}$ to repair C_1. (The proof below needs some slight modifications for the case of repairing

C_n, we omit this special case here.) Following exactly the same steps in the proof of Theorem 3.18, we obtain

$$\begin{bmatrix} I & I & \dots & I \\ A_1 & A_2 & \dots & A_{d_i+1} \\ A_1^2 & A_2^2 & \dots & A_{d_i+1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ A_1^{s_i-1} & A_2^{s_i-1} & \dots & A_{d_i+1}^{s_i-1} \end{bmatrix} \begin{bmatrix} W_1C_1 \\ W_2C_2 \\ \vdots \\ W_{d_i+1}C_{d_i+1} \end{bmatrix} = 0, \tag{3.29}$$

where $W_1, W_2, \dots, W_{d_i+1}$ are some $l \times l$ invertible matrices. Let $C_i' = W_iC_i$. Let $c_{i,a}'$ denote the a-th coordinate of the column vector C_i' for all $a = 0, \dots, l-1$. We can write out the parity-check equations (3.29) coordinatewise:

$$\begin{gathered} \sum_{i=1}^{d_i+1} \beta_{i,a_i,t} c'_{i,a(i,a_i\oplus t)} = 0 \\ \text{for all } t = 0,1,\dots,s_i-1 \text{ and } a = 0,\dots,l-1, \end{gathered} \tag{3.30}$$

where $\oplus$ denotes addition modulo s, $\beta_{i,u,0} = 1$ and $\beta_{i,u,t} = \prod_{v=u}^{u\oplus(t-1)} \lambda_{i,v}$ for $t = 1, \dots, s_i-1$. Clearly (3.30) indicates that for any $t = 0, 1, \dots, s-1$, the coordinates $\{c'_{1,a} : a_i = t, t\oplus 1, t\oplus 2 \dots, t\oplus(s_i-1)\}$ can be determined by $\{c'_{j,a} : j = 2, 3, \dots, d_i+1, a_i = t\}$. Thus C_1' can be determined by $\{c_{j,a} : j = 2, 3, \dots, d_i+1, a_i = 0, s_i, 2s_i, 3s_i, \dots, s-s_i\}$. Since $s_i|s$, we conclude that C_1' and thus C_1 itself can be recovered by downloading a vector in F^{l/s_i} from each of the nodes $C_2, C_3, \dots, C_{d_i+1}$. This completes the proof. □

Corollary 3.20. *The $(n, k, (n-k)^{n-1})$ MDS array code given by Construction 3.4 has d-optimal repair property if $(d+1-k)|(n-k)$.*

3.12 Concluding remarks

In this chapter we resolved the previously open problem of constructing explicit high-rate MSR codes. The intuition behind our approach to constructing explicit MSR codes is as follows. Most earlier constructions relied on the systematic generator representation of the codes. It is well known that the MDS property of the code leads to the requirement that every square submatrix of the systematic generator matrix be invertible. This is a rather stringent condition which the previous papers handled by resorting to some form of random coding arguments. In our construction we rely on a (non-systematic) parity-check representation of the codes. In this form, the MDS property only requires that every $r \times r$ submatrix is invertible, which is a much easier condition to fulfill. Moreover, since we use the block Vandermonde structure (see (2.3) and (3.3)), this condition can be easily satisfied by setting $A_i - A_j$ to be invertible and $A_iA_j = A_jA_i$ for all $i \neq j$ (see Lemma 3.9). This simple observation leads us to the explicit constructions of MSR codes.

Chapter 4: Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization

In this chapter we consider codes that, in addition to the minimum possible repair bandwidth, also have the optimal access property. As a reminder, this means that the data accessed at the helper nodes is the same as the data downloaded for the purposes of repair. We use the same notation as before, so MDS codes with (h, d)-optimal access perform repair of h nodes by addressing d helper nodes. Optimal access is a more restrictive condition, and the penalty for attaining it is the increased value of minimum possible sub-packetization l. In particular, according to [62], a necessary condition for $(1, n-1)$-optimal access is given by the bound $l \geq r^{(k-1)/r}$.

In the previous chapter, for any n and r, we presented an explicit construction of optimal-access MDS codes with sub-packetization $l = r^{n-1}$. In this chapter, we take up the question of reducing the sub-packetization value l to make it approach the aforementioned lower bound. We construct an explicit family of optimal-access codes with $l = r^{\lceil n/r \rceil}$, which differs from the optimal value by at most a factor of r^2. These codes can be constructed over any finite field F as long as $|F| \geq r\lceil n/r \rceil$, and afford low-complexity encoding and decoding procedures.

We also define a version of the repair problem that bridges the context of regenerating codes and codes with locality constraints (LRC codes), which we call *group repair with optimal access*. In this variation, we assume that the set of $n = sm$ nodes is partitioned into m repair groups of size s, and require that the amount of accessed data for repair is the smallest possible whenever the $d = s + k - 1$ helper nodes include all the other $s - 1$ nodes from the same group as the failed node. For this problem, we construct a family of codes with the group optimal access property. These codes can be constructed over any field F of size $|F| \geq n$, and also afford low-complexity encoding and decoding procedures.

The results of this chapter were published in [78].

4.1 Introduction

4.1.1 Optimal-access codes

The subclass of optimal-access codes is of particular interest among optimal-repair codes. Existence and constructions of optimal-access codes were studied in several recent works. We mention the results of [67] which established existence of optimal-access MDS array codes with $l = r^k$ and [47, 49] which proved existence of such codes with $l = r^{\lceil n/r \rceil}$,

although both results require a large-size finite field F. As for explicit constructions, until recently they were known only for $r = 2, 3$. Namely, [67] constructed codes with $l = r^k$ over the field $\mathbb{F}_3$ if $r = 2$ and $\mathbb{F}_4$ if $r = 3$ (independent of k). The paper [44] constructed systematic optimal-access MDS array codes[1] with $l = r^{k/r}$, where k is a multiple of r. The size of the underlying field for [44] is at least $k/2 + 1$ for $r = 2$ and $2k + 1$ for $r = 3$. In Chapter 3, we proposed a family of optimal-access MDS array codes with sub-packetization $l = r^{n-1}$, and this is the only explicit family of optimal-access MDS array codes for $r > 3$ known in the literature.

In this work we present an explicit construction of optimal-access MDS array codes for any r and n with sub-packetization $l = r^{\lceil n/r \rceil}$, which differs from the lower bound by a factor of at most r^2. These codes can be constructed over any finite field F as long as $|F| \geq r\lceil n/r \rceil$, and the encoding and decoding procedures of these codes have low complexity.

Remark 4.1. *The repair problem has been studied for a relatively short time, so the related terminology is still somewhat unsettled. For instance, [47, 49] refer to codes with $l = r^{\lceil n/r \rceil}$ as codes with polynomial sub-packetization. This implicitly assumes the asymptotic regime of $k = Rn$, i.e., of codes with a fixed rate R bounded away from 1. At the same time, arguably the case of $r = o(n)$, for instance constant r, is more important for the repair problem because the encoded data in storage are likely to include only a small number of parity checks. In this regime the above value of l is an exponential function of the block length. To cover all the possible cases, we prefer not to use the terms polynomial or exponential to describe the growth rate of the parameter l.*

Remark 4.2. *As we already mentioned in Section 2.3, after the release of this chapter on arXiv (as [78]), Sasidharan et al. [51] and later Li et al. [33] independently gave explicit constructions of optimal-access MDS codes whose parameters (sub-packetization and field size) are the same as the codes presented in this chapter. Moreover, constructions in both papers are very similar to our construction.*

4.1.2 Repair groups and node regeneration: Group optimal access property

In this chapter, we also introduce a coding problem for distributed storage that bridges codes with locality, in particular, LRCs (e.g., [36, 57, 59]) and regenerating codes. To motivate it, consider an architecture of distributed storage under which $n = sm$ storage nodes are partitioned into m local groups (we assume throughout that $s \leq r$). Nodes in the same group are logically better connected (for instance, they are geographically close to each other and thus have stable links between them), while the connectivity between nodes from different groups fluctuates smoothly over time (for instance, relying on a slowly fading channel). When a node fails, the system seeks to perform repair by accessing the minimum possible amount of data on a set of helper nodes. Efficient repair

[1] [44] also considered relaxing systematic optimal-access to systematic optimal-repair, and gave an explicit construction of codes for $r = 3$ and k a multiple of 4 with sub-packetization $l = r^{k/(r+1)}$ over the field of size linear in k.

also suggests that the helper nodes are chosen from a subset of nodes most easily reachable from the failed node. Since the failed node can always connect to all the nodes in the same group as itself, an MDS array code with the $(s, d = s + k - 1)$-group optimal access property can minimize the disk I/O and network traffic during the repair of any single failed node for such systems.

This motivates the following definition.

Definition 4.1. *Let $\mathcal{C}$ be an $(n = sm, k, l)$ MDS array code whose nodes are partitioned into m groups of size s each. We say that $\mathcal{C}$ has the $(s, d = s + k - 1)$-*group optimal access property *if $N(\mathcal{C}, \{i\}, \mathcal{R}_i) = dl/s$ for any i and any set of helper nodes $\mathcal{R}_i$ of size d that contains all the $s - 1$ nodes from the same group as i.*

By definition, repair of the failed node in the group repair mode can be performed by accessing the volume of data that attains the lower bound (2.1), justifying the optimality qualifier.

In this chapter, we construct an explicit family of $(n = sm, k = n - r, l = s^m)$ MDS array codes with the $(s, d = s + k - 1)$-group optimal access property for any s, m and r such that $r \geq s$. These codes can be constructed over any finite field F as long as $|F| \geq n$, and are equipped with low-complexity encoding and decoding. Our construction is flexible in the sense that it allows any number m of local groups and any number $s \leq r$ of storage nodes in a local group.

Remark 4.3. *We note that coding designs that address data regeneration based on different conditions at different helper nodes, based on access conditions or transient unavailability (degraded reads or hard errors) have been considered in a number of earlier works. For instance, in [77] we constructed codes that support repair of one or more failed nodes by accessing any set of d helper nodes in the same encoding block. A different line of research that establishes conditions under which helper node selection improves the storage/bandwidth tradeoff was recently developed in [1]. Yet another link between regenerating codes and LRC codes is the "local regeneration" problem [27, 32], where the local repair of the code is also required to have small bandwidth.*

4.1.3 Organization of the chapter

The code constructions are presented in the next section. Section 4.3 contains a proof of the MDS property of the constructed codes, and Section 4.4 gives a proof of the group optimal access property.

4.2 Code construction

The code constructions in this chapter are still defined by designing matrices $A_{t,i}, 0 \leq t \leq r - 1, 1 \leq i \leq n$ in (2.3). For an $l \times l$ matrix A, let $A(a, b)$ be the entry in the a-th row and b-th column, $0 \leq a, b \leq l - 1$. Below we use the convention that $0^0 = 1$.

Construction 4.1. *Let s, r and m be positive integers such that $s \leq r \leq sm$, let $n = sm, l = s^m$. Let F be a finite field of size $|F| \geq n$, let $\{\lambda_i\}_{i \in [n]}$ be n distinct elements in*

F, and let $\gamma \in F\backslash\{0,1\}$. Given an integer $a, 0 \le a \le l-1$, we write its s-ary expansion as $a = (a_m, a_{m-1}, \ldots, a_1)$.

For $v \in [m]$ and $0 \le u \le s-1$, define $a(v,u) := (a_m, a_{m-1}, \ldots, a_{v+1}, u, a_{v-1}, a_{v-2}, \ldots, a_1)$. For $v \in [m], u = 0, 1, \ldots, s-1$, and $t = 0, 1, \ldots, r-1$, define an $l \times l$ matrix $A_{t,(v-1)s+u+1}$ as follows: for $0 \le a, b \le l-1$, let

$$A_{t,(v-1)s+u+1}(a,b) = \begin{cases} \lambda^t_{(v-1)s+u+1} & \text{if } a_v < u, b = a, \\ \gamma\lambda^t_{(v-1)s+u+1} & \text{if } a_v > u, b = a, \\ \lambda^t_{(v-1)s+w+1} & \text{if } a_v = u, b = a(v,w) \\ & \text{for some } w \in \{0,1,\ldots,s-1\}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

We construct an $(n, k = n-r, l)$ array code defined by (2.3), where the matrices $A_{t,i}$ are defined in (4.1).

We will show that the code $\mathcal{C}$ defined by Construction 4.1 has the MDS property. In the case of $s = r$ it also has the optimal access property, while if $s < r$ it has the group optimal access property.

In Section 4.3 we give an example of the above matrices for $s = r = 3$ and $m = 2$ and show that the obtained codes have the MDS property.

In Construction 4.1 we assumed that the code length is a product of two numbers, s and m. While this assumption leads to a simple uniform formulation of the code construction, it can be easily lifted at the expense of a more detailed notation. Namely, the following construction extends the case of $s = r$ in Construction 4.1 to cover all possible code length n and has essentially the same properties as the codes defined above.

Construction 4.2. *Let $n = rm + r'$ and $l = r^{m+1}$, where $r > 0, m \ge 0$ are integers and $1 \le r' \le r-1$. Let F be a finite field of size $|F| \ge r(m+1)$, let $\{\lambda_i, i = 1, \ldots, r(m+1)\}$ be distinct elements of F, and let $\gamma \in F\backslash\{0,1\}$. Given an integer a between 0 and $l-1$, we write its r-ary expansion as $a = (a_{m+1}, a_m, \ldots, a_1)$. For $v \in \{1, \ldots, m+1\}$ and $0 \le u \le r-1$, define $a(v,u) := (a_{m+1}, a_m, \ldots, a_{v+1}, u, a_{v-1}, a_{v-2} \ldots, a_1)$.*

For $v \in [m], u = 0, 1, \ldots, r-1$, and $t = 0, 1, \ldots, r-1$, define an $l \times l$ matrix $A_{t,(v-1)r+u+1}$ as follows: for $0 \le a, b \le l-1$, let

$$A_{t,(v-1)r+u+1}(a,b) = \begin{cases} \lambda^t_{(v-1)r+u+1} & \text{if } a_v < u, b = a, \\ \gamma\lambda^t_{(v-1)r+u+1} & \text{if } a_v > u, b = a, \\ \lambda^t_{(v-1)r+w+1} & \text{if } a_v = u, b = a(v,w) \\ & \text{for some } w \in \{0,1,\ldots,r-1\}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

For $u = 0, 1, \ldots, r'-1$, and $t = 0, 1, \ldots, r-1$, define an $l \times l$ matrix $A_{t,mr+u+1}$ as

follows: for $0 \leq a, b \leq l-1$, *let*

$$A_{t,mr+u+1}(a,b) = \begin{cases} \lambda_{mr+u+1}^t & \text{if } a_{m+1} < u, b = a, \\ \gamma\lambda_{mr+u+1}^t & \text{if } a_{m+1} > u, b = a, \\ \lambda_{mr+w+1}^t & \text{if } a_{m+1} = u, b = a(m+1,w) \\ & \text{for some } w \in \{0,1,\ldots,r-1\}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

We construct an $(n = rm + r', k = n - r, l = r^{m+1})$ *array code defined by* (2.3)*, where the matrices* $A_{t,i}, 0 \leq t \leq r-1, 1 \leq i \leq n$ *are defined in* (4.2)-(4.3)*.*

4.3 The MDS property

In this section we show that the code family given by Construction 4.1 has the MDS property. We start with an example that shows the working of the definition (2.3)-(4.1) as well as provides intuition for the proof of the MDS property given below in this section. While the notation in the proof makes it difficult to glean an intuitive picture, this example serves to visualize the ideas behind the construction and the proof.

4.3.1 Example

Take $s = r = 3$ and $m = 2$ in Construction 4.1, so $n = 6$ and $l = 9$. Let us write out the 9×9 matrices $A_{t,i}, i = 1, \ldots, 6$. The code presented below can be realized over any field of size $|F| \geq n = 6$, so the smallest field is $\mathbb{F}_7$.

$$A_{t,1} = \begin{bmatrix} \lambda_1^t & \lambda_2^t & \lambda_3^t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \gamma\lambda_1^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma\lambda_1^t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_1^t & \lambda_2^t & \lambda_3^t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma\lambda_1^t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^t & \lambda_2^t & \lambda_3^t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^t \end{bmatrix},$$

$$A_{t,2} = \begin{bmatrix} \lambda_2^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \lambda_1^t & \lambda_2^t & \lambda_3^t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma\lambda_2^t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2^t & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_1^t & \lambda_2^t & \lambda_3^t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2^t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2^t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^t & \lambda_2^t & \lambda_3^t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2^t \end{bmatrix},$$

$$A_{t,3} = \begin{bmatrix} \lambda_3^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \lambda_3^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \lambda_1^t & \lambda_2^t & \lambda_3^t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_3^t & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_3^t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_1^t & \lambda_2^t & \lambda_3^t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_3^t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_3^t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^t & \lambda_2^t & \lambda_3^t \end{bmatrix},$$

$$A_{t,4} = \begin{bmatrix} \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t & 0 & 0 \\ 0 & \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t & 0 \\ 0 & 0 & \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t \\ 0 & 0 & 0 & \gamma\lambda_4^t & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma\lambda_4^t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma\lambda_4^t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_4^t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_4^t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_4^t \end{bmatrix},$$

$$A_{t,5} = \begin{bmatrix} \lambda_5^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \lambda_5^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_5^t & 0 & 0 & 0 & 0 & 0 & 0 \\ \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t & 0 & 0 \\ 0 & \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t & 0 \\ 0 & 0 & \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t \\ 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^t \end{bmatrix},$$

$$A_{t,6} = \begin{bmatrix} \lambda_6^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \lambda_6^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_6^t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_6^t & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_6^t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda_6^t & 0 & 0 & 0 \\ \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t & 0 & 0 \\ 0 & \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t & 0 \\ 0 & 0 & \lambda_4^t & 0 & 0 & \lambda_5^t & 0 & 0 & \lambda_6^t \end{bmatrix}.$$

The MDS property states that any 3×3 block submatrix of the 3×6 block matrix formed of the matrices $A_{t,i}$ is invertible (here the blocks are $l \times l$ matrices). Below we show this for the matrix

$$B = \begin{bmatrix} A_{0,1} & A_{0,2} & A_{0,5} \\ A_{1,1} & A_{1,2} & A_{1,5} \\ A_{2,1} & A_{2,2} & A_{2,5} \end{bmatrix}$$

Let X is a column vector in F^{27} with coordinates $X = (x_0, x_1, \ldots, x_{26})^T$. Our claim will follow if we prove that $BX = 0$ implies that $X = 0$.

We proceed as follows. For convenience of presentation, let us permute the rows of B to obtain a matrix $D = PB$, where the permutation matrix $(P_{ij})_{0 \le i,j \le 26}$ is given by

$$P_{ij} = 1 \text{ iff } i = (j - j \bmod 9)/9 + 3(j \bmod 9). \tag{4.4}$$

It is clear that P has exactly one 1 in each row, so it is indeed a permutation on $\{0, 1, \ldots, 26\}$ (note that multiplication by a full-rank matrix does not change the rank). We will prove that the matrix D has a trivial null space, i.e., $DX = 0$ implies that $X = 0$. Writing out the condition $DX = 0$ explicitly, we obtain a system of equations given in (4.5).

Since the coefficients λ_i are distinct for different i, the highlighted rows in (4.5) imply that $x_2 = x_8 = x_{11} = x_{17} = x_{20} = x_{26} = 0$. Eliminating these variables from (4.5), we obtain a system of equations given by (4.6).

Looking at the first three rows in (4.6), and treating $x_1 + x_9$ as a new variable, we conclude that $x_0 = x_1 + x_9 = x_{18} = 0$. Similarly, the second group of three rows implies that $\gamma x_1 + x_9 = x_{10} = x_{19} = 0$. Taking these results together and noting that $\gamma \neq 1$, we see that $x_0 = x_1 = x_{18} = x_9 = x_{10} = x_{19} = 0$.

$$
\left[\begin{array}{cccccccccccccccccccccccccc}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & \lambda_4 & 0 & 0 & \lambda_5 & 0 & 0 & \lambda_6 & 0 & 0 \\
0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & \lambda_4^2 & 0 & 0 & \lambda_5^2 & 0 & 0 & \lambda_6^2 & 0 & 0 \\
0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & \lambda_4 & 0 & 0 & \lambda_5 & 0 & 0 & \lambda_6 & 0 \\
0 & 0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & \lambda_4^2 & 0 & 0 & \lambda_5^2 & 0 & 0 & \lambda_6^2 & 0 \\
0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2 & 0 & 0 & 0 & 0 & 0 & \lambda_4 & 0 & 0 & \lambda_5 & 0 & 0 & \lambda_6 \\
0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2^2 & 0 & 0 & 0 & 0 & 0 & \lambda_4^2 & 0 & 0 & \lambda_5^2 & 0 & 0 & \lambda_6^2 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^2
\end{array}\right]
\left[\begin{array}{c} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \\ x_{16} \\ x_{17} \\ x_{18} \\ x_{19} \\ x_{20} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{25} \\ x_{26} \end{array}\right] = 0. \tag{4.5}
$$

$$
\left[\begin{array}{ccccccccccccccccccccc}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_1 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_1^2 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 & 0 & 0 & 0 \\
0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5^2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & \lambda_4 & 0 & \lambda_5 & 0 & 0 & \lambda_6 & 0 \\
0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & 0 & \lambda_2^2 & 0 & 0 & 0 & 0 & \lambda_4^2 & 0 & \lambda_5^2 & 0 & 0 & \lambda_6^2 & 0 \\
0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & \lambda_4 & 0 & \lambda_5 & 0 & 0 & \lambda_6 \\
0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & 0 & 0 & 0 & \lambda_4^2 & 0 & \lambda_5^2 & 0 & 0 & \lambda_6^2 \\
0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 \\
0 & 0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_5^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 \\
0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5 & 0 \\
0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & \lambda_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_5^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & \gamma \\
0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & \lambda_2 & 0 & 0 & 0 & 0 & \gamma\lambda_5 \\
0 & 0 & 0 & 0 & 0 & 0 & \gamma\lambda_1^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1^2 & \lambda_2^2 & 0 & 0 & 0 & 0 & \gamma\lambda_5^2
\end{array}\right]
\left[\begin{array}{c} x_0 \\ x_1 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_9 \\ x_{10} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \\ x_{16} \\ x_{18} \\ x_{19} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{25} \end{array}\right] = 0. \tag{4.6}
$$

A similar argument used for the last 9 rows in (4.6) shows that $x_5 = x_{14} = x_{23} = x_6 = x_7 + x_{15} = x_{24} = \gamma x_7 + x_{15} = x_{16} = x_{25} = 0$, and so $x_5 = x_{14} = x_{23} = x_6 = x_7 = x_{24} = x_{15} = x_{16} = x_{25} = 0$. Writing out the remaining equations, we obtain the following set of equations:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ \lambda_1 & \lambda_2 & \lambda_2 & 0 & \lambda_5 & 0 \\ \lambda_1^2 & \lambda_2^2 & \lambda_2^2 & 0 & \lambda_5^2 & 0 \\ 0 & \gamma & 1 & 1 & 0 & 1 \\ 0 & \gamma\lambda_1 & \lambda_1 & \lambda_2 & 0 & \lambda_5 \\ 0 & \gamma\lambda_1^2 & \lambda_1^2 & \lambda_2^2 & 0 & \lambda_5^2 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \\ x_{12} \\ x_{13} \\ x_{21} \\ x_{22} \end{bmatrix} = 0.$$

From this set of equations we obtain that $x_4 + x_{12} = x_3 = x_{21} = \gamma x_4 + x_{12} = x_{13} = x_{22} = 0$. Thus, $x_3 = x_4 = x_{21} = x_{12} = x_{13} = x_{22} = 0$. Overall these arguments prove that $X = 0$, and so B is invertible. □

4.3.2 A proof of the MDS property

Let us fix s, r, and m, so $n = sm$ and $l = s^m$. The code $\mathcal{C}$ given by Construction 4.1 is an MDS array code if for any $1 \le i_1 < i_2 < \cdots < i_r \le n$, the matrix

$$B_{s,r,m}[i_1, i_2, \ldots, i_r] := \begin{bmatrix} A_{0,i_1} & A_{0,i_2} & \ldots & A_{0,i_r} \\ A_{1,i_1} & A_{1,i_2} & \ldots & A_{1,i_r} \\ \vdots & \vdots & \vdots & \vdots \\ A_{r-1,i_1} & A_{r-1,i_2} & \ldots & A_{r-1,i_r} \end{bmatrix}$$

is invertible. Below we suppress the parameters s, r, m, and $i_1, i_2, \ldots, i_r$ from the notation and write B to refer to this matrix. In other words, given a vector $X \in F^{rl}$ we need to prove that

$$BX = 0 \tag{4.7}$$

implies that $X = 0$, where X is a vector in F^{rl} with coordinates $X = (x_0, x_1, \ldots, x_{rl-1})^T$. The proof essentially follows the example in Sect. 4.3.1. We begin with a preview which also serves to introduce some notation.

In order to transform B into a matrix of the form (4.5), let us define a permutation matrix $(P_{ij})_{0 \le i,j < rl}$ by setting

$$P_{ij} = 1 \text{ iff } i = (j - j \bmod l)/l + r(j \bmod l); \tag{4.8}$$

compare with our example in (4.4).

Define a matrix $D = PB$. We shall prove that

$$DX = 0 \tag{4.9}$$

implies that $X = 0$. (The full notation for D should be $D_{s,r,m}[i_1, i_2, \ldots, i_r]$, but we again suppress the parameters.) For $a = 0, 1, \ldots, l-1$, define $D^{(a)}$ to be the $r \times rl$ submatrix

of D consisting of rows $ar, ar+1, \ldots, ar+r-1$. Define a column vector

$$L_i = (1, \lambda_i, \ldots, \lambda_i^{r-1})^T, \quad i \in [n]. \tag{4.10}$$

On account of (4.1), for every $a \in \{0, 1, \ldots, l-1\}$, all the nonzero columns of $D^{(a)}$ belong to the set

$$\{L_1, L_2, \ldots, L_n, \gamma L_1, \gamma L_2, \ldots, \gamma L_n\}.$$

We proceed by defining several subsets of the set of column indices of $D^{(a)}$ for every $a \in \{0, 1, \ldots, l-1\}$:

- Let $\mathcal{U}^{(a)} \subset [n]$ be a subset such that $i \in \mathcal{U}^{(a)}$ if and only if there is a nonzero column in $D^{(a)}$ equal to either L_i or γL_i;
- Let $\mathcal{J}^{(a)} \subset \{0, 1, \ldots, rl\}$ be the set of indices of the nonzero columns in $D^{(a)}$;
- For $i \in [n]$, define the set $\mathcal{J}^{(a)}(i) \subset \mathcal{J}^{(a)}$ as
  $\mathcal{J}^{(a)}(i) = \{j \in \mathcal{J}^{(a)} : \text{ the } j\text{th column of } D^{(a)} \text{ is either } L_i \text{ or } \gamma L_i\}$.

In our example above, let $D^{(0)}$ be the first $r = 3$ rows of the matrix $D = D_{3,3,2}[1, 2, 5]$ in (4.5). Then the only nonzero columns in $D^{(0)}$ are of type L_1, L_2, L_3, or L_5, and so $\mathcal{U}^{(0)} = \{1, 2, 3, 5\}$. The indices of the nonzero columns are given by $\mathcal{J}^{(0)} = \{0, 1, 2, 9, 18\}$, and $\mathcal{J}^{(0)}(1) = \{0\}, \mathcal{J}^{(0)}(2) = \{1, 9\}, \mathcal{J}^{(0)}(3) = \{2\}, \mathcal{J}^{(0)}(5) = \{18\}$.

Clearly the sets $\mathcal{J}^{(a)}(i)$ form a partition of the set $\mathcal{J}^{(a)}$, so

$$\mathcal{J}^{(a)} = \bigcup_{i \in \mathcal{U}^{(a)}} \mathcal{J}^{(a)}(i).$$

According to (4.1), for every $i \in [n]$, every diagonal entry of $A_{t,i}$ is either λ_i^t or $\gamma\lambda_i^t$ (see also the example in Sect. 4.3.1 where we explicitly write out the matrices $A_{t,1}, \ldots, A_{t,6}$). Therefore, for every $i \in [n]$, every row of $A_{t,i}$ contains at least one of the elements λ_i^t and $\gamma\lambda_i^t$. As an immediate consequence, for every $i \in \{i_1, \ldots, i_r\}$, the set of nonzero columns in the strip of r rows $D^{(a)}, a = 0, 1, \ldots, l-1$ contains at least one column out of the pair $(L_i, \gamma L_i)$. This implies that $\{i_1, i_2, \ldots, i_r\} \subseteq \mathcal{U}^{(a)}$ for all $a \in \{0, 1, \ldots, l-1\}$. Our strategy of proving that (4.9) is satisfied only for $X = 0$ will be to find a set of indices

$$\mathcal{S} = \{a : a \in \{0, 1, \ldots, l-1\}, |\mathcal{U}^{(a)}| = r\}$$

(as before $\mathcal{S} = \mathcal{S}_{s,r,m}[i_1, i_2, \ldots, i_r]$). For every $a \in \mathcal{S}$, all the nonzero columns of $D^{(a)}$ belong to the set

$$\{L_{i_1}, L_{i_2}, \ldots, L_{i_r}, \gamma L_{i_1}, \gamma L_{i_2}, \ldots, \gamma L_{i_r}\}.$$

Since the columns $L_{i_1}, L_{i_2}, \ldots, L_{i_r}$ form a Vandermonde matrix, we conclude that the corresponding variables or their linear combinations are 0, and therefore we can eliminate some of the variables in (4.9). Referring to our example, this set is exactly the set of highlighted rows in the matrix in (4.5), and thus in this case the set of strip labels equals $\mathcal{S} = \{2, 8\}$.

Suppose that such a set $\mathcal{S}$ is found. Then the equations

$$D^{(a)}X = 0, \quad a \in \mathcal{S},$$

will imply that

$$x_j = 0 \text{ for all } j \in \bigcup_{a \in \mathcal{S}} \mathcal{J}^{(a)}. \tag{4.11}$$

Using (4.11), we can eliminate some of the variables in (4.9) and obtain the system

$$\widetilde{D}\widetilde{X} = 0$$

(in the example this corresponds to obtaining (4.6) from (4.5)).

Let $\tilde{l} = l - |\mathcal{S}|$ be the remaining count of variables x_i, so that $\widetilde{D}$ is an $r\tilde{l} \times r\tilde{l}$ matrix and $\widetilde{X} \in F^{r\tilde{l}}$. We iterate the above steps and define subsets $\widetilde{\mathcal{D}}^{(a)}, \widetilde{\mathcal{U}}^{(a)}$ for all $a \in \{0, 1, \ldots, \tilde{l} - 1\}$. In the example the matrix $\widetilde{D}$ is given in (4.6), and the new set $\widetilde{\mathcal{S}}$ of the groups of r equations is given by $\widetilde{\mathcal{S}} = \{0, 1, 4, 5, 6\}$. Restricting our attention to the equations in these groups, we eliminate another subset of variables by proving that they are necessarily equal to zero, and continue this procedure until finally all of the variables have been shown to be zero.

A rigorous proof uses induction and is given below.

Theorem 4.1. *The code $\mathcal{C}$ given by Construction 4.1 is an $(n = sm, k = n - r, l = s^m)$ MDS array code.*

To prove this theorem we need to show that for every choice of the indices $1 \leq i_1 < i_2 < \cdots < i_r \leq n$, the only X that satisfies (4.9) is the all-zero vector. This will follow from the next two lemmas.

Lemma 4.2. *For any $1 \leq i_1 < i_2 < \cdots < i_r \leq n$,*

$$\min_{a \in \{0,1,\ldots,l-1\}} |\mathcal{U}^{(a)}[i_1, i_2, \ldots, i_r]| = r.$$

Proof. As mentioned above, $\{i_1, i_2, \ldots, i_r\} \subseteq \mathcal{U}^{(a)}$ for any $a \in \{0, 1, \ldots, l-1\}$, so $\min_a |\mathcal{U}^{(a)}| \geq r$ (we again simplify the notation by dropping the indices $i_1, \ldots, i_r$). We claim that there always exists an index $a \in \{0, 1, \ldots, l-1\}$ such that $\mathcal{U}^{(a)} = \{i_1, i_2, \ldots, i_r\}$. To see this, define the (possibly empty) set

$$\mathcal{G} = \{v : v \in [m], \{(v-1)s + 1, (v-1)s + 2, \ldots, vs\} \subseteq \{i_1, i_2, \ldots, i_r\}\}. \tag{4.12}$$

Now choose $a \in \{0, 1, \ldots, l-1\}$ such that

$$\text{if } v \in [m] \backslash \mathcal{G} \text{ then } (v-1)s + a_v + 1 \notin \{i_1, i_2, \ldots, i_r\}. \tag{4.13}$$

To see that we can always find such an a, notice that by (4.12), for every $v \in [m] \backslash \mathcal{G}$, there exists a number $y_v \in \{1, 2, \ldots, s\}$ such that $(v-1)s + y_v \notin \{i_1, i_2, \ldots, i_r\}$. In order to satisfy (4.13), it suffices to set the v-th digit of a to be $y_v - 1$, i.e., to set $a_v = y_v - 1$.

Next we prove that $\mathcal{U}^{(a)} = \{i_1, i_2, \ldots, i_r\}$, which is equivalent to the following statement:

Claim: For any $i \in \{i_1, i_2, \ldots, i_r\}$, all the nonzero entries of the a-th row of $A_{t,i}$ belong to the set

$$\{\lambda_{i_1}^t, \lambda_{i_2}^t, \ldots, \lambda_{i_r}^t, \gamma\lambda_{i_1}^t, \gamma\lambda_{i_2}^t, \ldots, \gamma\lambda_{i_r}^t\}.$$

Let us write $i = (v-1)s + u + 1$, where $v \in [m]$ and $u \in \{0, 1, \ldots, s-1\}$. We consider the following two cases:

Case 1: $v \in [m] \backslash \mathcal{G}$

In this case (4.13) states that $(v-1)s + a_v + 1 \notin \{i_1, i_2, \ldots, i_r\}$, and therefore $(v-1)s + a_v + 1 \neq i$ (keep in mind that $i \in \{i_1, i_2, \ldots, i_r\}$). As a result, $a_v \neq u$. According to (4.1), if $a_v < u$, then the a-th row of $A_{t,i}$ contains a single nonzero entry λ_i^t; if $a_v > u$, then the a-th row of $A_{t,i}$ contains a single nonzero entry $\gamma\lambda_i^t$. Both λ_i^t and $\gamma\lambda_i^t$ belong to the set $\{\lambda_{i_1}^t, \lambda_{i_2}^t, \ldots, \lambda_{i_r}^t, \gamma\lambda_{i_1}^t, \gamma\lambda_{i_2}^t, \ldots, \gamma\lambda_{i_r}^t\}$. This establishes the claim for this case.

Case 2: $v \in \mathcal{G}$

If $a_v \neq u$, then the claim holds by the argument in Case 1, so let $a_v = u$. The a-th row of $A_{t,i}$ contains s nonzero entries $\lambda_{(v-1)s+1}^t, \lambda_{(v-1)s+2}^t, \ldots, \lambda_{vs}^t$. By (4.12), they all belong to $\{\lambda_{i_1}^t, \lambda_{i_2}^t, \ldots, \lambda_{i_r}^t\}$. This establishes the claim for this case and completes the proof of the lemma.

□

Lemma 4.3. *For every $a \in \{0, 1, \ldots, l-1\}$, $x_j = 0$ for all $j \in \mathcal{J}^{(a)}$.*

Proof. We will argue by induction on the cardinality of the set $\mathcal{U}^{(a)}$. By Lemma 4.2, to establish the induction basis we need to prove that the lemma holds for all a such that $|\mathcal{U}^{(a)}| = r$. Let a be one of the values that have this property. We will prove that for every $t \in [r]$, $x_j = 0$ for all $j \in \mathcal{J}^{(a)}(i_t)$.

Let us write the index $i_t, t \in [r]$ in the form

$$i_t = (v_t - 1)s + u_t + 1,$$

where $v_t \in [m]$ and $0 \leq u_t \leq s-1$. Let us further partition $[r]$ into three disjoint subsets $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3$ as follows:

$$\mathcal{K}_1 = \{t : t \in [r], a_{v_t} = u_t\} \cup \{t : t \in [r], \nexists\, p \in [r] \text{ s.t. } v_p = v_t \text{ and } u_p = a_{v_t}\},$$
$$\mathcal{K}_2 = \{t : t \in [r], a_{v_t} > u_t\} \cap \{t : t \in [r], \exists\, p \in [r] \text{ s.t. } v_p = v_t \text{ and } u_p = a_{v_t}\},$$
$$\mathcal{K}_3 = \{t : t \in [r], a_{v_t} < u_t\} \cap \{t : t \in [r], \exists\, p \in [r] \text{ s.t. } v_p = v_t \text{ and } u_p = a_{v_t}\}.$$

We will prove our claim separately for each of these subsets, starting with $\mathcal{K}_1$. By definition (4.1), all the nonzero entries in the matrix $A_{h,(v-1)s+u+1}$ belong to the set $\{\lambda_{(v-1)s+1}^h, \lambda_{(v-1)s+2}^h, \ldots, \lambda_{vs}^h, \gamma\lambda_{(v-1)s+1}^h, \gamma\lambda_{(v-1)s+2}^h, \ldots, \gamma\lambda_{vs}^h\}$, where $h = 0, 1, \ldots, r-1$. Moreover, if $a_v \neq u$, then the a-th row of the matrix $A_{h,(v-1)s+u+1}$ contains only a single nonzero entry, either $\lambda_{(v-1)s+u+1}^h$ or $\gamma\lambda_{(v-1)s+u+1}^h$. On the other hand, if $a_v = u$, then the a-th row of the matrix $A_{h,(v-1)s+u+1}$ contains s nonzero entries. In particular, if $a_v = u$, then the only appearance of the element $\lambda_{(v-1)s+u+1}^h$ in the a-th row of all the matrices

$A_{h,i}, i \in [n]$ is in position (a,a) of $A_{h,(v-1)s+u+1}$, i.e., on the diagonal of $A_{h,(v-1)s+u+1}$ (and $\gamma\lambda^h_{(v-1)s+u+1}$ does not appear in the a-th row of any one of the matrices $A_{h,i}, i \in [n]$). As an immediate consequence, if $a_v = u$, then the column $L_{(v-1)s+u+1}$ appears at most once in the matrix $D^{(a)}$ (it appears when $(v-1)s+u+1 \in \{i_1, \ldots, i_r\}$), and $\gamma L_{(v-1)s+u+1}$ does not appear in $D^{(a)}$. Therefore, for each $t \in \mathcal{K}_1$, we have $\mathcal{J}^{(a)}(i_t) = \{(t-1)l + a\}$. The vectors $L_{i_j}, j = 1, \ldots, r$ are linearly independent, which implies that $x_{(t-1)l+a} = 0, t \in \mathcal{K}_1$.

Moving to the case $t \in \mathcal{K}_2$, observe that $\mathcal{J}^{(a)}(i_t) = \{(t-1)l+a, (p-1)l+a(v_t,u_t)\}$ which implies that

$$\gamma x_{(t-1)l+a} + x_{(p-1)l+a(v_t,u_t)} = 0. \tag{4.14}$$

Let us consider the submatrix $D^{(a(v_t,u_t))}$. Its nonzero columns are described as follows:

$$\mathcal{U}^{(a(v_t,u_t))} = \{i_1, i_2, \ldots, i_r\}, \quad \mathcal{J}^{(a(v_t,u_t))}(i_p) = \{(t-1)l + a, (p-1)l + a(v_t,u_t)\}.$$

From this we see that

$$x_{(t-1)l+a} + x_{(p-1)l+a(v_t,u_t)} = 0. \tag{4.15}$$

Since $\gamma \neq 1$, conditions (4.14) and (4.15) imply that $x_{(t-1)l+a} = x_{(p-1)l+a(v_t,u_t)} = 0$, exhausting the case of $t \in \mathcal{K}_2$.

The last remaining case $t \in \mathcal{K}_3$ is very similar to $\mathcal{K}_2$, and we again obtain that $x_j = 0$ for all $j \in \mathcal{J}^{(a)}(i_t)$. This concludes the proof of the induction basis.

Now suppose that the statement of the lemma holds true for all a such that $|\mathcal{U}^{(a)}| \leq w-1$ for some $w \geq r$ and let us prove it for all a such that $|\mathcal{U}^{(a)}| = w$. We again aim to show that for every $i \in \mathcal{U}^{(a)}$, $x_j = 0$ for all $j \in \mathcal{J}^{(a)}(i)$.

For $i \in \mathcal{U}^{(a)} \backslash \{i_1, i_2, \ldots, i_r\}$, there exists a unique $t \in [r]$ such that $a_{v_t} = u_t$ and $1 \leq i - (v_t - 1)s \leq s$, and $\mathcal{J}^{(a)}(i) = \{(t-1)l + a(v_t, \alpha)\}$, where $\alpha = i - (v_t-1)s - 1$. Consider the matrix $D^{(a(v_t,\alpha))}$. By our choice of i there is no $p \in [r]$ such that $v_p = v_t$ and $u_p = \alpha$. Therefore, $\mathcal{U}^{(a(v_t,\alpha))} \subset \mathcal{U}^{(a)}$ and $i \notin \mathcal{U}^{(a(v_t,\alpha))}$. This implies that $|\mathcal{U}^{(a(v_t,\alpha))}| \leq w-1$, so the induction hypothesis applies, and $x_j = 0$ for all $j \in \mathcal{J}^{(a(v_t,\alpha))}$. Furthermore, $(t-1)l + a(v_t,\alpha) \in \mathcal{J}^{(a(v_t,\alpha))}$, and thus $x_{(t-1)l+a(v_t,\alpha)} = 0$. Rephrasing this, we have shown that for every $i \in \mathcal{U}^{(a)} \backslash \{i_1, i_2, \ldots, i_r\}$, $x_j = 0$ for all $j \in \mathcal{J}^{(a)}(i)$.

We are left to consider the variables $\cup_{t\in[r]}\{x_j : j \in \mathcal{J}^{(a)}(i_t)\}$. To show that they must be 0 to satisfy $DX = 0$, we note that the left-hand side of $D^{(a)}X = 0$ reduces to a linear combination of the linearly independent columns $L_{i_j}, j = 1, \ldots, r$. Therefore, the coefficients of this linear combination are all 0. This implies that for every $t \in [r]$, $x_j = 0$ for all $j \in \mathcal{J}^{(a)}(i_t)$. This claim is proved in exactly the same way as the induction basis above, so we omit the proof. This completes the induction step. □

Corollary 4.4. *The code $\mathcal{C}'$ given by Construction 4.2 is an $(n = rm + r', k = n - r, l = r^{m+1})$ MDS array code.*

Proof. Consider the MDS code $\mathcal{C}$ of length $n = r(m+1)$ given by Construction 4.1. The code $\mathcal{C}'$ is obtained from $\mathcal{C}$ by discarding $r - r'$ coordinates, and so is also MDS. □

We finish this section by a brief remark on the complexity of node repair (decoding) and encoding of the constructed codes. From the coding-theoretic perspective, the

repair problem is erasure correction, and is very similar to the encoding problem (the encoding is correction of $r = n - k$ erasures from k known nodes). From the example in Section 4.3.1 and the proof of Theorem 4.1, we immediately see that the encoding and decoding procedures of codes given by either of Constructions 4.1 or 4.2 rely on inversion of $r \times r$ matrices over F, and thus have low complexity.

4.4 The optimal access property

Consider the code $\mathcal{C}$ given by Construction 4.1. Recall that we write the i-th node as $C_i = (c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})^T$. For every $i \in [n]$, define the following set of coordinates of the i-th node:

$$\mathcal{C}_i^{(v,u)} = \{c_{i,a} : a \in \{0, 1, \ldots, l-1\}, a_v = u\}. \tag{4.16}$$

where a_v is the v-th digit of a.

Theorem 4.5. *Consider the code $\mathcal{C}$ given by Construction 4.1. For any $v \in [m]$ and $u \in \{0, 1, \ldots, s-1\}$, the node $C_{(v-1)s+u+1}$ can be recovered from the elements in the set*

$$\mathcal{C}^{(v,u)} = \bigcup_{\substack{i \in [n] \\ i \neq (v-1)s+u+1}} \mathcal{C}_i^{(v,u)}.$$

Proof. Fix a value of t. Let us write out the a-th row of the equation $\sum_{i=1}^{n} A_{t,i} C_i = 0$ (cf. (2.3)). We first notice that since $n = sm$, the equation $\sum_{i=1}^{n} A_{t,i} C_i = 0$ is equivalent to

$$\sum_{q=1}^{m} \sum_{w=0}^{s-1} A_{t,(q-1)s+w+1} C_{(q-1)s+w+1} = 0.$$

By definition (4.1), if $a_q > w$, then the a-th row of $A_{t,(q-1)s+w+1}$ contains a single nonzero entry $\gamma \lambda_{(q-1)s+w+1}^t$ located in the a-th column; if $a_q < w$, then the a-th row of $A_{t,(q-1)s+w+1}$ contains a single nonzero entry $\lambda_{(q-1)s+w+1}^t$ located in the a-th column; if $a_q = w$, then the a-th row of $A_{t,(q-1)s+w+1}$ contains s nonzero entries located in columns $a(q,0)$ to $a(q,s-1)$. Thus, the a-th row of the equation $\sum_{i=1}^{n} A_{t,i} C_i = 0$ can be written as follows:

$$\sum_{q \in [m]} \Bigg(\sum_{w=0}^{a_q - 1} \gamma \lambda_{(q-1)s+w+1}^t c_{(q-1)s+w+1,a} + \sum_{w=0}^{s-1} \lambda_{(q-1)s+w+1}^t c_{(q-1)s+a_q+1,a(q,w)} + \sum_{w=a_q+1}^{s-1} \lambda_{(q-1)s+w+1}^t c_{(q-1)s+w+1,a} \Bigg) = 0, \tag{4.17}$$

where the first sum in the parentheses corresponds to the case $a_q > w$; the second sum corresponds to the case $a_q = w$; and the third sum corresponds to the case $a_q < w$. Since our aim is to repair the node $C_{(v-1)s+u+1}$, let us break the sum on $q \in [m]$ in (4.17) into

two parts: $q \neq v$ and $q = v$. We obtain the following equation:

$$\sum_{q\neq v, q\in[m]} \Big(\sum_{w=0}^{a_q-1} \gamma \lambda_{(q-1)s+w+1}^{t} c_{(q-1)s+w+1,a} + \sum_{w=0}^{s-1} \lambda_{(q-1)s+w+1}^{t} c_{(q-1)s+a_q+1,a(q,w)} + \sum_{w=a_q+1}^{s-1} \lambda_{(q-1)s+w+1}^{t} c_{(q-1)s+w+1,a} \Big)$$
$$+ \Big(\sum_{w=0}^{a_v-1} \gamma \lambda_{(v-1)s+w+1}^{t} c_{(v-1)s+w+1,a} + \underline{\sum_{w=0}^{s-1} \lambda_{(v-1)s+w+1}^{t} c_{(v-1)s+a_v+1,a(v,w)}} + \sum_{w=a_v+1}^{s-1} \lambda_{(v-1)s+w+1}^{t} c_{(v-1)s+w+1,a} \Big) = 0. \tag{4.18}$$

For all $t = 0, 1, \ldots, r-1$ and all a satisfying $a_v = u$, all the terms in (4.18) apart from the underlined term can be found from the elements in the set $\mathcal{C}^{(v,u)}$. Indeed,

$$\begin{bmatrix} \sum_{w=0}^{s-1} c_{(v-1)s+u+1,a(v,w)} \\ \sum_{w=0}^{s-1} \lambda_{(v-1)s+w+1} c_{(v-1)s+u+1,a(v,w)} \\ \vdots \\ \sum_{w=0}^{s-1} \lambda_{(v-1)s+w+1}^{r-1} c_{(v-1)s+u+1,a(v,w)} \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \lambda_{(v-1)s+1} & \lambda_{(v-1)s+2} & \cdots & \lambda_{(v-1)s+s} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{(v-1)s+1}^{r-1} & \lambda_{(v-1)s+2}^{r-1} & \cdots & \lambda_{(v-1)s+s}^{r-1} \end{bmatrix} \begin{bmatrix} c_{(v-1)s+u+1,a(v,0)} \\ c_{(v-1)s+u+1,a(v,1)} \\ \vdots \\ c_{(v-1)s+u+1,a(v,s-1)} \end{bmatrix}.$$

Since $r \geq s$, the coordinates in the set $\{c_{(v-1)s+u+1,a(v,w)} : w = 0, 1, \ldots, s-1\}$ can be found from the set $\{\sum_{w=0}^{s-1} \lambda_{(v-1)s+w+1}^{t} c_{(v-1)s+u+1,a(v,w)} : t = 0, 1, \ldots, r-1\}$ for all $a = 0, 1, \ldots, l-1$. In particular, the values in the set

$$\{c_{(v-1)s+u+1,a} : a = 0, 1, \ldots, l-1\} = \{c_{(v-1)s+u+1,a(v,w)} : a_v = u, w = 0, 1, \ldots, s-1\}$$

can be found from the values

$$\Big\{ \sum_{w=0}^{s-1} \lambda_{(v-1)s+w+1}^{t} c_{(v-1)s+u+1,a(v,w)} : a_v = u, t = 0, 1, \ldots, r-1 \Big\}.$$

As mentioned above, the values $\{\sum_{w=0}^{s-1} \lambda_{(v-1)s+w+1}^{t} c_{(v-1)s+u+1,a(v,w)} : a_v = u, t = 0, 1, \ldots, r-1\}$ are uniquely determined by the elements in the set $\mathcal{C}^{(v,u)}$. We conclude that the entire node $C_{(v-1)s+u+1}$ can be determined by the elements in $\mathcal{C}^{(v,u)}$. □

Theorem 4.6. *Let $s = r$, then the code $\mathcal{C}$ given by Construction 4.1 has the optimal access property.*

Proof. Since $\mathcal{C}_i^{(v,u)}$ contains exactly a $(1/r)$th fraction of the coordinates of C_i, by Theorem 4.5 we only need to access a $1/r$ proportion of the data stored in each of the surviving nodes in order to repair a single node failure. □

Theorem 4.7. *The code $\mathcal{C}'$ given by Construction 4.2 has the optimal access property.*

Proof. Using the same approach as in Theorem 4.5, we can show that for all $v \in [m]$ and $u \in \{0, 1, \ldots, r-1\}$, the node $C_{(v-1)r+u+1}$ can be determined by the elements in the set $\{c_{i,a} : i \neq (v-1)r + u + 1, a_v = u\}$, and that for all $u \in \{0, 1, \ldots, r' - 1\}$, the node C_{mr+u+1} can be determined by the elements in the set $\{c_{i,a} : i \neq mr + u + 1, a_{m+1} = u\}$. □

Note that upon setting $s = r$ in Construction 4.1, we obtain optimal-access MDS array codes with code length divisible by r, and Construction 4.2 gives optimal-access MDS array codes with code length not divisible by r. Thus we can construct optimal-access $(n, n-r, r^{\lceil n/r \rceil})$ MDS array codes for any n and r.

4.4.1 Group optimal access

In the second part of this section we examine the group optimal access property of the codes considered above and prove the following result.

Theorem 4.8. *The code $\mathcal{C}$ given by Construction 4.1 has the $(s, s+k-1)$-group optimal access property.*

This theorem will follow from Theorem 4.5 and a lemma proved below in this section.

Let us define the following subset of indices

$$\mathcal{N}^{(v)} = [n] \backslash \{(v-1)s + 1, (v-1)s + 2, \ldots, vs\}, \quad v \in [m].$$

Lemma 4.9. *Consider the code $\mathcal{C}$ given by Construction 4.1. For every $v \in [m]$ and $u \in \{0, 1, \ldots, s-1\}$, and every $\mathcal{M} \subseteq \mathcal{N}^{(v)}$ with cardinality $|\mathcal{M}| = k$, the values of elements in the set $\cup_{i \in \mathcal{N}^{(v)}} \mathcal{C}_i^{(v,u)}$ (cf. (4.16)) are determined by the elements in the set $\cup_{i \in \mathcal{M}} \mathcal{C}_i^{(v,u)}$.*

Before proving Lemma 4.9, let us explain how this lemma together with Theorem 4.5 implies Theorem 4.8. The group optimal access property simply means that for every $v \in [m]$ and every $u \in \{0, 1, \ldots, s-1\}$, the node $C_{(v-1)s+u+1}$ can be repaired by connecting to $\{C_{(v-1)s+u'+1} : u' \in \{0, 1, \ldots, s-1\} \setminus \{u\}\}$ together with any other k helper nodes in the set $\{C_i : i \in \mathcal{N}^{(v)}\}$, and accessing exactly $(1/s)$th fraction of coordinates of each helper node. In Theorem 4.5, we have shown that the node $C_{(v-1)s+u+1}$ can be repaired if we know the values of all the elements in the set $\bigcup_{i \neq (v-1)s+u+1} \mathcal{C}_i^{(v,u)}$ from all the surviving nodes. By definition each set $\mathcal{C}_i^{(v,u)}$ contains exactly $(1/s)$th fraction of the coordinates of C_i. This is where we need Lemma 4.9 which states that the values of the elements in the set $\cup_{i \in \mathcal{N}^{(v)}} \mathcal{C}_i^{(v,u)}$ (cf. (4.16)) can be calculated from the elements in the set $\cup_{i \in \mathcal{M}} \mathcal{C}_i^{(v,u)}$ for any $\mathcal{M} \subseteq \mathcal{N}^{(v)}$ such that $|\mathcal{M}| = k$. This establishes that the code $\mathcal{C}$ given by Construction 4.1 has the $(s, s+k-1)$-group optimal access property.

Proof. (of Lemma 4.9) The case $s = r$ is trivially true, so we assume that $s < r$. Let us write (2.3) in matrix form:

$$\begin{bmatrix} A_{0,1} & A_{0,2} & \dots & A_{0,n} \\ A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ \vdots & \vdots & \vdots & \vdots \\ A_{r-1,1} & A_{r-1,2} & \dots & A_{r-1,n} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix} = 0. \tag{4.19}$$

Let us permute the equations in this system using the permutation matrix defined in (4.8) and denote the resulting matrix of coefficients by H. As before, let $H^{(a)}, a = 0, 1, \dots, l-1$ be a submatrix of H formed of rows $ar, ar+1, \dots, (a+1)r-1$. Let us write out the equation $H^{(a)}[C_1, C_2, \dots, C_n]^T = 0$ as follows:

$$\begin{aligned} \sum_{q\in[m]\setminus v} \Big(& \sum_{w=0}^{a_q-1} \gamma c_{(q-1)s+w+1,a} L_{(q-1)s+w+1} \\ & + \sum_{w=0}^{s-1} c_{(q-1)s+a_q+1,a(q,w)} L_{(q-1)s+w+1} + \sum_{w=a_q+1}^{s-1} c_{(q-1)s+w+1,a} L_{(q-1)s+w+1} \Big) \\ = -\Big(& \sum_{w=0}^{a_v-1} \gamma c_{(v-1)s+w+1,a} L_{(v-1)s+w+1} + \sum_{w=0}^{s-1} c_{(v-1)s+a_v+1,a(v,w)} L_{(v-1)s+w+1} \\ & + \sum_{w=a_v+1}^{s-1} c_{(v-1)s+w+1,a} L_{(v-1)s+w+1} \Big), \end{aligned} \tag{4.20}$$

where the vectors $L_i, i \in [n]$ are defined in (4.10) (this equation amounts to taking columns $(v-1)s+1, (v-1)s+2, \dots, vs$ to the right-hand side of (4.19)).

Define polynomials $g_0^{(v)}(x) = \prod_{w=1}^{s}(x - \lambda_{(v-1)s+w})$, and $g_j^{(v)}(x) = x^j g_0^{(v)}(x)$ for $j = 0, 1, \dots, r-s-1$. Since the degree of $g_j^{(v)}(x)$ is less than r for all $j = 0, 1, \dots, r-s-1$, we can write

$$g_j^{(v)}(x) = \sum_{t=0}^{r-1} g_{j,t}^{(v)} x^t.$$

Define the $(r-s) \times r$ matrix

$$G^{(v)} = \begin{bmatrix} g_{0,0}^{(v)} & g_{0,1}^{(v)} & \dots & g_{0,r-1}^{(v)} \\ g_{1,0}^{(v)} & g_{1,1}^{(v)} & \dots & g_{1,r-1}^{(v)} \\ \vdots & \vdots & \vdots & \vdots \\ g_{r-s-1,0}^{(v)} & g_{r-s-1,1}^{(v)} & \dots & g_{r-s-1,r-1}^{(v)} \end{bmatrix}.$$

We have

$$G^{(v)} L_i = \begin{bmatrix} g_{0,0}^{(v)} & g_{0,1}^{(v)} & \dots & g_{0,r-1}^{(v)} \\ g_{1,0}^{(v)} & g_{1,1}^{(v)} & \dots & g_{1,r-1}^{(v)} \\ \vdots & \vdots & \vdots & \vdots \\ g_{r-s-1,0}^{(v)} & g_{r-s-1,1}^{(v)} & \dots & g_{r-s-1,r-1}^{(v)} \end{bmatrix} \begin{bmatrix} 1 \\ \lambda_i \\ \vdots \\ \lambda_i^{r-1} \end{bmatrix} = \hat{L}_i^{(v)}, \tag{4.21}$$

where $\hat{L}_i^{(v)}, i \in [n]$ is given by

$$\hat{L}_i^{(v)} = \begin{bmatrix} g_0^{(v)}(\lambda_i) \\ g_1^{(v)}(\lambda_i) \\ \vdots \\ g_{r-s-1}^{(v)}(\lambda_i) \end{bmatrix} = g_0^{(v)}(\lambda_i) \begin{bmatrix} 1 \\ \lambda_i \\ \vdots \\ \lambda_i^{r-s-1} \end{bmatrix}. \tag{4.22}$$

By definition, $g_0^{(v)}(\lambda_i) = 0$ for all $(v-1)s+1 \leq i \leq vs$, and so $G^{(v)}L_i = 0$ for all $(v-1)s+1 \leq i \leq vs$. Observe that every term on the right-hand-side of (4.20) contains one column vector from the set $\{L_i : (v-1)s+1 \leq i \leq vs\}$. As a result, multiplying equation (4.20) by $G^{(v)}$ on the left, we obtain

$$\sum_{q \neq v, q \in [m]} \Big(\sum_{w=0}^{a_q - 1} \gamma c_{(q-1)s+w+1,a} G^{(v)} L_{(q-1)s+w+1} + \sum_{w=0}^{s-1} c_{(q-1)s+a_q+1,a(q,w)} G^{(v)} L_{(q-1)s+w+1} + \sum_{w=a_q+1}^{s-1} c_{(q-1)s+w+1,a} G^{(v)} L_{(q-1)s+w+1} \Big) = 0.$$

Using (4.21), this equation can be written as

$$\sum_{q \neq v, q \in [m]} \Big(\sum_{w=0}^{a_q - 1} \gamma c_{(q-1)s+w+1,a} \hat{L}^{(v)}_{(q-1)s+w+1} + \sum_{w=0}^{s-1} c_{(q-1)s+a_q+1,a(q,w)} \hat{L}^{(v)}_{(q-1)s+w+1} + \sum_{w=a_q+1}^{s-1} c_{(q-1)s+w+1,a} \hat{L}^{(v)}_{(q-1)s+w+1} \Big) = 0. \tag{4.23}$$

In order to prove the theorem, we only need to prove that given any $v \in [m]$ and $u \in \{0, 1, \ldots, s-1\}$, and any $i_1 < i_2 < \cdots < i_{r-s}$ such that $\{i_1, i_2, \ldots, i_{r-s}\} \subseteq \mathcal{N}^{(v)}$, the values of elements in the set $\cup_{t=1}^{r-s} \mathcal{C}_{i_t}^{(v,u)}$ can be determined by the elements in the set $\cup_{i \in \mathcal{M}} \mathcal{C}_i^{(v,u)}$, where $\mathcal{M} = \mathcal{N}^{(v)} \backslash \{i_1, i_2, \ldots, i_{r-s}\}$. We will prove that we can find the elements in the set $\cup_{t=1}^{r-s} \mathcal{C}_{i_t}^{(v,u)}$ from $\cup_{i \in \mathcal{M}} \mathcal{C}_i^{(v,u)}$ using equation (4.23).

For a given $a = 0, 1, \ldots, l-1$, denote by $E^{(a)}$ the set of equations in (4.23). Each set $E^{(a)}$ contains $r-s$ scalar equations. Observe that if $a_v = u$, then $E^{(a)}$ contains only elements in $\cup_{i \in \mathcal{N}^{(v)}} \mathcal{C}_i^{(v,u)}$. Moreover, every element in the set $\cup_{i \in \mathcal{N}^{(v)}} \mathcal{C}_i^{(v,u)}$ appears at least once in the equations $\{E^{(a)} : a_v = u\}$. Therefore, equations (4.23) contain $(r-s)l/s$ scalar equations and $(r-s)l/s$ unknown elements, namely, the elements in the set $\cup_{t=1}^{r-s} \mathcal{C}_{i_t}^{(v,u)}$.

Let us set all the elements in the set $\cup_{i \in \mathcal{M}} \mathcal{C}_i^{(v,u)}$ to 0 in $E^{(a)}$ and denote by $E_{\mathcal{M}}^{(a)}$ the obtained set of equations. (In other words, $E_{\mathcal{M}}^{(a)}$ are the equations obtained by eliminating all the terms which contain elements in the set $\cup_{i \in \mathcal{M}} \mathcal{C}_i^{(v,u)}$ in (4.23).) In order to prove the lemma, it suffices to show that the equations $\{E_{\mathcal{M}}^{(a)} : a_v = u\}$ imply that all the elements in the set $\cup_{t=1}^{r-s} \mathcal{C}_{i_t}^{(v,u)}$ are 0.

Recall that equation (4.9) can be viewed as the set of equations $\{D^{(a)}[i_1, i_2, \ldots, i_r]X = 0 : a \in \{0, 1, \ldots, l-1\}\}$. Note that, once we form a vector X of the elements in the set $\cup_{t=1}^{r-s}\mathcal{C}_{i_t}^{(v,u)}$, equations $\{E_{\mathcal{M}}^{(a)} : a_v = u\}$ have almost the same form as $\{D^{(a)}[i_1, i_2, \ldots, i_r]X = 0 : a \in \{0, 1, \ldots, l-1\}\}$. The only difference is that in the equations $\{E_{\mathcal{M}}^{(a)} : a_v = u\}$, the columns $\{\hat{L}_i^{(v)} : i \in \mathcal{N}^{(v)}\}$ take place of the vectors $\{L_i : i \in [n]\}$, and $r - s$ takes place of r.

Examining closely the proof of Theorem 4.1, we note that the property that any r columns in the set $\{L_i, i \in [n]\}$ are linearly independent, suffices to show that $X = 0$. Since $p_0^{(v)}(\lambda_i) \neq 0$ for any $i \in \mathcal{N}^{(v)}$, by Definition (4.22) any $r - s$ vectors in the set of vectors $\{\hat{L}_i^{(v)} : i \in \mathcal{N}^{(v)}\}$ are also linearly independent. Thus, using exactly the same arguments as in the proof of Theorem 4.1, we can show that the equations $\{E_{\mathcal{M}}^{(a)} : a_v = u\}$ imply that all the elements in the set $\cup_{t=1}^{r-s}\mathcal{C}_{i_t}^{(v,u)}$ are 0. This completes the proof of the theorem. □

4.5 Concluding remarks

In the previous chapter we constructed optimal-access MDS codes using the block Vandermonde structure; see the discussion in Section 3.12. Although it enables us to satisfy the conditions imposed by the MDS property in a straightforward way, this additional structure limits our choices of the matrices in (2.3), and the penalty is a relatively large sub-packetization value $l = r^{n-1}$.

For this reason, in the construction presented above we do not use the block Vandermonde structure. Instead, we divide the rl parity equations into l groups of size r, and each such group forms the parity equations of a GRS code. These l groups are designed in such a way that whenever r nodes fail, one can recover them by successive decoding of l GRS codes. This approach enables one to satisfy the conditions of the MDS property and at the same time leads to a nearly optimal sub-packetization value.

A natural extension of the presented construction which we have not been able to obtain so far, is to allow for general values of d between k and $n - 1$. A recent work [52] presented explicit MDS codes with the $(1, d)$-optimal access property and small sub-packetization value for a particular choice of $d < n - 1$. At the same time, for general values of d, the best known explicit construction is still the one presented in Chapter 3 of this thesis (see also [77]).

Chapter 5: Optimal repair of Reed-Solomon codes: Achieving the cut-set bound

In this chapter we study the repair problem of RS codes. Previous works such as [26] and follow-up papers [10, 11] that addressed this problem stopped short of constructing optimal-repair RS codes, so this has remained an open problem. It is this problem that we address and resolve in this chapter.

More specifically, given any n, k and d such that $k \leq d \leq n-1$, we explicitly construct an (n, k) RS code with sub-packetization $l = \exp((1 + o(1))n \log n)$ that has the $(1, d)$-optimal repair property. We also prove an almost matching lower bound on l, showing that the super-exponential scaling is both necessary and sufficient for achieving the cut-set bound (2.1) using linear repair schemes. More precisely, we prove that for scalar MDS codes (including the RS codes) with optimal repair property, the sub-packetization l must satisfy $l \geq \exp((1 + o(1))k \log k)$.

We also consider the problem of lowering the value of l at the expense of allowing ourselves slightly greater repair bandwidth than the cut-set bound (2.1). Along these lines, given any n and r, we construct an $(n, k = n - r)$ RS code with sub-packetization $l = r^n$ that asymptotically achieves (2.1)for the repair of any single failed node from all the other $n - 1$ surviving nodes. The exponent of the sub-packetization value of this construction is $n \log r$. We note that in the regime of fixed r and growing n, the exponent of the sub-packetization value of this construction is much smaller than that of the aforementioned RS code construction.

The results presented in Sections 5.1-5.4 of this chapter were published in [64], and the results in Section 5.5 were published in [75].

5.1 Introduction

5.1.1 Repair schemes for scalar linear MDS codes

While there has been much research into constructions and properties of MSR codes specifically designed for the repair task, it is also of interest to study the repair bandwidth of general families of MDS codes, for instance, RS codes. In [56], Shanmugam et al. proposed a framework for studying the repair bandwidth of a scalar linear (n, k) MDS code $\mathcal{C}$ over some finite field E (called symbol field below). The idea of [56] is to "vectorize" the code construction by considering $\mathcal{C}$ as an array code over some subfield F of E. This approach provides a bridge between RS codes and MDS array codes, wherein the extension degree $l := [E : F]$ can be viewed as the value of sub-packetization. The

code $\mathcal{C}$ is viewed as an (n, k) MDS array code with sub-packetization l, and the repair bandwidth is defined exactly in the same way as above. The cut-set bound (2.1) and the definition of MSR codes also apply to this setup.

In this chapter we study repair of RS codes, focusing on linear repair schemes, i.e., we assume that the repair operations are linear over the field F. Guruswami and Wootters [26] gave a characterization for linear repair schemes of scalar linear MDS codes based on the framework in [56]. We will use this characterization to prove one of our main results, namely, a lower bound on the sub-packetization, so we recall it in the theorem below. In this theorem E is the degree-l extension of the field F. Viewing E as an l-dimensional vector space over F, we use the notation $\dim_F(a_1, a_2, \ldots, a_t)$ to refer to the dimension of the subspace spanned by the set $\{a_1, a_2, \ldots, a_t\} \subset E$ over F.

We will need a result from [26] which we state in the form that is suited to our needs.

Theorem 5.1 ([26]). *Let $\mathcal{C} \subseteq E^n$ be a scalar linear MDS code of length n. Let F be a subfield of E such that $[E : F] = l$. For a given $i \in \{1, \ldots, n\}$ the following statements are equivalent.*

(1) There is a linear repair scheme of the node c_i over F such that the repair bandwidth $N(\mathcal{C}, i, [n] \setminus \{i\}) \leq b$.

(2) There is a subset of codewords $\mathcal{P}_i \subseteq \mathcal{C}^{\perp}$ with size $|\mathcal{P}_i| = l$ such that

$$\dim_F(\{x_i : x \in \mathcal{P}_i\}) = l,$$

and

$$b \geq \sum_{j \in [n] \setminus \{i\}} \dim_F(\{x_j : x \in \mathcal{P}_i\})$$

In order to prove that (2) implies (1) in this theorem, [26] proposed a linear repair scheme which makes use of trace functionals. Since this chapter focuses on the repair problem of RS codes, we recap a special case of this linear repair scheme that is designed for RS codes.

5.1.2 The linear repair scheme of [26] for RS codes

Suppose we want to repair a code $\mathcal{C} = \mathrm{RS}_E(n, k, \Lambda)$ over the base field $F \subseteq E$. More precisely, if a single codeword symbol is erased, we will recover this symbol by download sub-symbols of the base field F from the surviving nodes. Let $\mathrm{tr}(\beta) = \mathrm{tr}_{E/F}(\beta) := \beta + \beta^q + \beta^{q^2} + \cdots + \beta^{q^{l-1}}$ be the trace function. In order to make the repair scheme F-linear, [26] uses F-linear transforms $L_\gamma : E \to F$ given by the *trace functionals* $L_\gamma(\beta) = \mathrm{tr}(\gamma\beta)$.

Let $\{\zeta_1, \ldots, \zeta_l\}$ be a basis for E over F, and let $\{\mu_1, \ldots, \mu_l\}$ be its dual (trace-orthogonal) basis, then for all $\beta \in E$

$$\beta = \sum_{i=1}^{l} (\mathrm{tr}(\zeta_i \beta)\mu_i).$$

Therefore, we can make the following observation: *If* $\{\zeta_1, \ldots, \zeta_l\}$ *is a basis for* E *over* F*, then* $\{\mathrm{tr}(\zeta_i \beta)\}_{i=1}^{l}$ *uniquely determines* β.

Let $\mathcal{C}^{\perp}$ be the dual code of $\mathcal{C} = \mathrm{RS}_E(n, k, \Lambda)$. Suppose that the codeword symbol c_i in a codeword $c = (c_1, \ldots, c_n) \in \mathcal{C}$ is erased. We can find l codewords $\{c_j^{\perp} = (c_{j,1}^{\perp}, \ldots, c_{j,n}^{\perp})\}_{j=1}^{l}$ in $\mathcal{C}^{\perp}$ such that $\{c_{1,i}^{\perp}, \ldots, c_{l,i}^{\perp}\}$ is a basis of E over F. By the observation above, knowing the values of $\{\mathrm{tr}(c_{j,i}^{\perp} c_i)\}_{j=1}^{l}$ suffices to recover the erased symbol c_i. Since the trace is an F-linear transformation, we have

$$\mathrm{tr}(c_{j,i}^{\perp} c_i) = -\sum_{t \neq i} \mathrm{tr}(c_{j,t}^{\perp} c_t) \text{ for all } j \in [l].$$

Thus knowing the values of $\{\{\mathrm{tr}(c_{j,t}^{\perp} c_t)\}_{j \in [l]}\}_{t \in [n], t \neq i}$ suffices to recover c_i. Let B_t be a maximal linearly independent subset of the set $\{c_{j,t}^{\perp}\}_{j \in [l]}$ over F. Again due to the F-linearity of the trace function, $\{\mathrm{tr}(c_{j,t}^{\perp} c_t)\}_{j \in [l]}$ can be calculated from $\{\mathrm{tr}(\beta c_t)\}_{\beta \in B_t}$. Consequently, c_i can be recovered from $\{\{\mathrm{tr}(\beta c_t)\}_{\beta \in B_t}\}_{t \in [n], t \neq i}$. The total number of sub-symbols in F we need to download from the surviving nodes to recover c_i is $\sum_{t \in [n], t \neq i} \dim_F(\{c_{j,t}^{\perp}\}_{j \in [l]})$.

We conclude that to efficiently recover c_i, we need to find l codewords in $\mathcal{C}^{\perp}$ that minimize the quantity $\sum_{t \in [n], t \neq i} \dim_F(\{c_{j,t}^{\perp}\}_{j \in [l]})$ under the condition that $\{c_{1,i}^{\perp}, \ldots, c_{l,i}^{\perp}\}$ is a basis for E over F.

As already remarked, $\mathcal{C}^{\perp} = \mathrm{GRS}_E(n, n-k, \Lambda, v)$ for some nonzero coefficients $v = (v_1, \ldots, v_n) \in E^n$. Choosing a codeword from $\mathcal{C}^{\perp} = \mathrm{GRS}_E(n, n-k, \Lambda, v)$ is equivalent to choosing a polynomial with degree less than $n - k$. Suppose $\Lambda = \{\alpha_1, \ldots, \alpha_n\}$. Since $v_1, \ldots, v_n$ are nonzero constants, our task of efficiently repairing c_i is reduced to finding l polynomials $\{f_j\}_{j \in [l]}$ of degree less than $n - k$ such that the quantity

$$\sum_{t \in [n], t \neq i} \dim_F(\{f_j(\alpha_t)\}_{j \in [l]}) \tag{5.1}$$

is minimized under the condition that $\{f_1(\alpha_i), \ldots, f_l(\alpha_i)\}$ is a basis for E over F.

5.1.3 Related results in the literature

In addition to this general linear repair scheme for RS codes, the authors of [26] also presented a specific repair scheme for a family of RS codes and further proved that (in some cases) the repair bandwidth of RS codes using this scheme is the smallest possible among all linear repair schemes and all scalar linear MDS codes with the same parameters. At the same time, the approach of [26] has some limitations. Namely, their repair scheme applies only for small sub-packetization $l = \log_{n/r} n$, and the optimality claim only holds for this specific sub-packetization value. At the same time, in order to achieve the cut-set bound, l needs to be exponentially large in n for a fixed value of r [22], so the repair bandwidth of this scheme is rather far from the bound. Subsequently, Ye and Barg [75] used the general linear repair scheme in [26] to construct an explicit family of RS codes with asymptotically optimal repair bandwidth: the ratio between the actual repair bandwidth of the codes and the cut-set bound approaches 1 as the code length n goes to infinity.

In [26], there is one more restriction on the parameters of the RS codes, namely they achieve the smallest possible repair bandwidth only if the number of parities is of

the form $r = q^s, (l-s)|l$. In [11], Dau and Milenkovic generalized the scheme in [26] and extended their results to all values of $s = 1, \ldots, l-1$. The repair bandwidth attained in [11] is $(n-1)(l-s)$ symbols of F for $r \geq q^s$, and is the smallest possible whenever r is a power of q. In [10], Dau et al. extended the results of [26] to repair of multiple erasures.

To summarize the earlier work, constructions of RS codes (or any scalar MDS codes) that meet the cut-set bound have as yet been unknown, so the existence question of such codes has been an open problem. In this chapter, we resolve this problem in the affirmative, presenting such a construction. We also prove a lower bound on the sub-packetization of scalar linear MDS codes that attain the cut-set bound with a linear repair scheme, showing that there is a penalty for the scalar case compared to MDS array codes.

5.1.4 Our results

(1) **Explicit constructions of RS codes achieving the cut-set bound:** Given any n, k and $d, k \leq d \leq n-1$, we construct an (n, k) RS code over the field $E = \mathbb{F}_{q^l}$ that achieves the cut-set bound (2.1) when repairing *any* single failed node from *any* d helper nodes. As above, we view RS codes over E as vector codes over the subfield $F = \mathbb{F}_q$. The main novelty in our construction is the choice of the evaluation points for the code in such a way that the degrees of the evaluation points over F are distinct primes. As a result, the symbol field is an extension field of F with degree no smaller than the product of these distinct primes. For the actual repair we rely on the linear scheme proposed in [26] (this is essentially the only possible linear repair approach).

The value of sub-packetization l of our construction equals s times the product of the first n distinct primes in an arithmetic progression,

$$l = s\left(\prod_{\substack{i=1 \\ p_i \equiv 1 \bmod s}}^{n} p_i\right),$$

where $s := d + 1 - k$. This product is a well-studied function in number theory, related to a classical arithmetic function $\psi(n, s, a)$ (which is essentially the sum of logarithms of the primes). The prime number theorem in arithmetic progressions (for instance, [31, p.121]) yields asymptotic estimates for l. In particular, for fixed s and large n, we have $l = e^{(1+o(1))n \log n}$.

In contrast, for the case $d = n-1$ (i.e., $s = r = n-k$), there exist MSR array codes that attain sub-packetization $l = r^{\lceil n/(r+1) \rceil}$ [68], which is the smallest known value among MSR codes[1]. So although this distinct prime structure allows us to achieve the cut-set bound, it makes us pay a penalty on the sub-packetization.

[1] The construction of [68] achieves the cut-set bound only for repair of systematic nodes, and gives $l = r^{\lceil k/(r+1) \rceil}$. Using the approach of [77], it is possible to modify the construction of [68] and to obtain an MSR code with $l = r^{\lceil n/(r+1) \rceil}$.

(2) **A lower bound on the sub-packetization of scalar MDS codes achieving the cut-set bound:** Surprisingly, we also show that the distinct prime structure discussed above is necessary for any scalar linear MDS code (not just the RS codes) to achieve the cut-set bound under linear repair. Namely, given d such that $k+1 \leq d \leq n-1$, we prove that for any (n, k) scalar linear MSR code with $(1, d)$-optimal repair property, the sub-packetization l is bounded below by $l \geq \prod_{i=1}^{k-1} p_i$, where p_i is the i-th smallest prime. By the Prime Number Theorem [31], we obtain the lower asymptotic bound on l of the form $l \geq e^{(1+o(1))k \log k}$.

(3) **Main result:** In summary, we obtain the following results for the smallest possible sub-packetization of scalar linear MDS codes, including the RS codes, whose repair bandwidth achieves the cut-set bound.

Theorem 5.2. *Let $\mathcal{C}$ be an $(n, k = n - r)$ scalar linear MDS code over the field $E = \mathbb{F}_{q^l}$, and let d be an integer satisfying $k + 1 \leq d \leq n - 1$. Suppose that for any single failed node of $\mathcal{C}$ and any d helper nodes there is a linear repair scheme over $\mathbb{F}_q$ that uses the bandwidth $dl/(d + 1 - k)$ symbols of $\mathbb{F}_q$, i.e., it achieves the cut-set bound (2.1). For a fixed $s = d + 1 - k$ and $n, k \to \infty$ the following bounds on the smallest possible sub-packetization hold true:*

$$e^{(1+o(1))k \log k} \leq l \leq e^{(1+o(1))n \log n}. \tag{5.2}$$

For large s, we have $l \leq s \prod_{i: p_i \equiv 1 \bmod s}^{n} p_i$, where the product goes over the first n distinct primes in the arithmetic progression.

Remark 5.1. *The bound on l can be made more explicit even for large s, and the answer depends on whether we accept the Generalized Riemann Hypothesis (if yes, we can still claim the bound $l \leq \exp((1 + o(1))n \log n)$).*

(4) **Discussion: Array codes and scalar codes** The lower bound in (5.2) is much larger than the sub-packetization of many known MSR array code constructions. To make the comparison between the repair parameters of scalar codes and array codes clearer, we summarize the tradeoff between the repair bandwidth and the sub-packetization of some known MDS code constructions in Table 5.1. We only list the papers considering the repair of a single node from all the remaining $n-1$ helper nodes. Moreover, in the table we limit ourselves to explicit code constructions, and do not list multiple existence results that appeared in recent years.

While there is an overlap between the table in Section 2.3 and Table 5.1 below, the latter table is compiled with the comparison of scalar and vector regenerating codes in mind, and provides a better illustration of our current discussion.

As discussed earlier, the constructions of [11, 26] have optimal repair bandwidth among all the RS codes with the same sub-packetization value as in these papers[2]. At

[2]Expressing the sub-packetization of the construction in [11] via n and r is difficult. The precise form of the result in [11] is as follows: for every $s < l$ and $r \geq q^s$, the authors construct repair schemes of RS codes of length $n = q^l$ with repair bandwidth $(n - 1)(l - s)$. Moreover, if $r = q^s$, then the schemes proposed in [11] achieve the smallest possible repair bandwidth for codes with these parameters.

Table 5.1: Tradeoff between repair bandwidth and sub-packetization ($d = n - 1$)

Code construction	Repair bandwidth	sub-packetization	achieving cut-set bound
Array codes			
$(n, k = n - r, l)$ MSR array codes for $2k \leq (n + 1)$, [42]	$\frac{(n-1)l}{r}$	$l = r$	Yes
(n, k, l) MSR array codes (a modification of [68])	$\frac{(n-1)l}{r}$	$l = r^{\lceil n/(r+1) \rceil}$	Yes
(n, k, l) MSR array codes [78]	$\frac{(n-1)l}{r}$	$l = r^{\lceil n/r \rceil}$	Yes
(n, k, l) MDS array codes with design parameter $t \geq 1$ [24]	$(1 + \frac{1}{t})\frac{(n-1)l}{r}$	$l = r^t$	No
Scalar codes			
(n, k) RS code [75] (Section 5.5 of this chapter)	$< \frac{(n+1)l}{r}$	$l = r^n$	No
(n, k) RS code [26]	$n - 1$	$l = \log_{n/r} n$	No
(n, k) RS code [11]	$(n - 1)l(1 - \log_n r)$	$\log_q n$	No
(n, k) RS code (Section 5.3 of this chapter)	$\frac{(n-1)l}{r}$	$l \approx n^n$	Yes

the same time, these values are too small for the constructions of [11, 26] to achieve the cut-set bound. From the first three rows of the table one can clearly see that the achievable sub-packetization values for MSR array codes are much smaller than the lower bound for scalar linear MSR codes derived in this chapter. This is to be expected since for array codes we only require the code to be linear over the "repair field," i.e., F, and not the symbol field E as in the case of scalar codes.

5.1.5 Organization of the chapter

In Section 5.2, we present a simple construction of RS codes that achieve the cut-set bound for some of the nodes. This construction is inferior to the more involved construction of Section 5.3, but simple to follow, and already contains some of the main ideas of the later part, so we include it as a warm-up for the later results. In Section 5.3, we present our main construction of RS codes that achieve the cut-set bound for the repair of any single node, proving the upper estimate in (5.2). In Section 5.4, we prove the lower bound on the sub-packetization of scalar linear MSR codes, finishing the proof of (5.2). In Section 5.5, we construct an $(n, k = n - r)$ RS code with sub-packetization $l = r^n$ that asymptotically achieves (2.1)for the repair of any single failed node from all the other $n - 1$ surviving nodes.

5.2 A simple construction

In this section we present a simple construction of RS codes that achieve the cut-set bound for the repair of certain nodes. We note that any (n, k) MDS code trivially allows repair that achieves the cut-set bound for $d = k$. We say that a node in an MDS code has a *nontrivial optimal repair scheme* if for a given $d > k$ it is possible to repair this node from any d helper nodes with repair bandwidth achieving the cut-set bound. The code family presented in this section is different from standard MSR codes in the sense that although the repair bandwidth of our construction achieves the cut-set bound, the number of helper nodes depends on the node being repaired.

Denote by $\pi(t)$ the number of primes less than or equal to t. Let F be a finite field and let E be the extension of F of degree t. The trace function $\mathrm{tr}_{E/F} : E \to F$ is defined by

$$\mathrm{tr}_{E/F}(x) := x + x^{|F|} + x^{|F|^2} + \cdots + x^{|F|^{t-1}}.$$

In the next theorem we construct a special subfamily of RS codes. Our construction enables nontrivial repair of $\pi(r)$ nodes, which without loss of generality we take to be nodes $1, 2, \ldots, \pi(r)$. Let $d_i, i = 1, 2, \ldots, \pi(r)$ be the number of helper nodes used to repair the i-th node. We will take $d_i = p_i + k - 1$, where p_i is the i-th smallest prime number. The repair scheme presented below supports repair of node i by connecting to any d_i helper nodes and downloading a $\frac{1}{p_i}$-th proportion of information stored at each of these nodes. Since $p_i = d_i - k + 1$, this justifies the claim of achieving the cut-set bound for repair of a single node.

Theorem 5.3. *Let $n \geq k$ be two positive integers, and let $r = n - k$. There exists an (n, k) RS code over a field E such that $\pi(r)$ of its coordinates admit nontrivial optimal repair schemes.*

Proof. Let $m := \pi(r)$ and let $q \geq n - m$ be a prime power. Let E be the $\left(\prod_{i=1}^{m} p_i\right)$-th degree extension of the finite field $\mathbb{F}_q$.

Let $\alpha_i, i = 1, \ldots, m$ be an element of order p_i over $\mathbb{F}_q$, so that $\mathbb{F}_{q^{p_i}} = \mathbb{F}_q(\alpha_i)$, where $\mathbb{F}_q(\alpha_i)$ denotes the field obtained by adjoining α_i to $\mathbb{F}_q$. It is clear that $E = \mathbb{F}_q(\alpha_1, \ldots, \alpha_m)$. Define m subfields F_i of E by setting

$$F_i = \mathbb{F}_q(\alpha_j : j \neq i),$$

so that $E = F_i(\alpha_i)$ and $[E : F_i] = p_i$, $i = 1, \ldots, m$. Let $\alpha_{m+1}, \ldots, \alpha_n \in \mathbb{F}_q$ be arbitrary $n - m$ distinct elements of the field, and let $\Omega = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$.

Let $\mathcal{C} = \mathrm{RS}_E(n, k, \Omega)$ be the RS code of dimension k with evaluation points Ω and let $\mathcal{C}^\perp$ be its dual code. We claim that for $i = 1, 2, \ldots, m$, the i-th coordinate (node) of $\mathcal{C}$ can be optimally repaired from any d_i helper nodes, where

$$d_i = p_i + k - 1.$$

Let $i \in \{1, 2, \ldots, m\}$ and let us show how to repair the ith node. Choose a subset of helper nodes $\mathcal{R}_i \subseteq [n] \backslash \{i\}, |\mathcal{R}_i| = d_i$, and note that since $p_i \leq r$, we have $d_i \leq n - 1$. Let $h(x)$ be the annihilator polynomial of the set $\{\alpha_j : j \in [n] \setminus (\mathcal{R}_i \cup \{i\})\}$, i.e.,

$$h(x) = \prod_{j \in [n] \backslash (\mathcal{R}_i \cup \{i\})} (x - \alpha_j). \tag{5.3}$$

Since $\deg(h(x)) = n - k - p_i$, we have $\deg(x^s h(x)) < r$ for all $s = 0, 1, \ldots, p_i - 1$. As a result, for all $s = 0, \ldots, p_i - 1$, the vector

$$(v_1 \alpha_1^s h(\alpha_1), \ldots, v_n \alpha_n^s h(\alpha_n)) \in \mathcal{C}^\perp, \tag{5.4}$$

cf. (2.2). Let $c = (c_1, \ldots, c_n) \in \mathcal{C}$ be a codeword. By (5.4) we have

$$\sum_{j=1}^{n} v_j h(\alpha_j) \alpha_j^s c_j = 0, \quad s = 0, \ldots, p_i - 1.$$

Let $\mathrm{tr}_i := \mathrm{tr}_{E/F_i}$ denote the trace from E to F_i. We have

$$\sum_{j=1}^{n} \mathrm{tr}_i(v_j h(\alpha_j) \alpha_j^s c_j) = 0, \quad s = 0, \ldots, p_i - 1.$$

Equivalently, we can write

$$\begin{aligned} \mathrm{tr}_i(v_i h(\alpha_i) \alpha_i^s c_i) &= -\sum_{j \neq i} \mathrm{tr}_i(v_j h(\alpha_j) \alpha_j^s c_j) \\ &= -\sum_{j \in \mathcal{R}_i} \mathrm{tr}_i(v_j h(\alpha_j) \alpha_j^s c_j) \\ &= -\sum_{j \in \mathcal{R}_i} \alpha_j^s \, \mathrm{tr}_i(v_j h(\alpha_j) c_j), \quad s = 0, \ldots, p_i - 1, \end{aligned} \tag{5.5}$$

where the second equality follows from (5.3) and the third follows because $\alpha_j \in F_i$ for all $j \neq i$ and tr_i is an F_i-linear map.

The information used to recover the value c_i (to repair the ith node) is comprised of the following d_i elements of F_i :

$$\mathrm{tr}_i(v_j h(\alpha_j) c_j), \quad j \in \mathcal{R}_i.$$

Let us show that these elements indeed suffice. First, by (5.5), given these elements, we can calculate the values of $\mathrm{tr}_i(v_i h(\alpha_i)\alpha_i^s c_i)$ for all $s = 0, \ldots, p_i - 1$. The mapping

$$\begin{aligned} &E \to F_i^{p_i} \\ &\gamma \mapsto \left(\mathrm{tr}_i\left(v_i h(\alpha_i)\gamma\right), \mathrm{tr}_i\left(v_i h(\alpha_i)\alpha_i\gamma\right), \ldots, \mathrm{tr}_i\left(v_i h(\alpha_i)\alpha_i^{p_i-1}\gamma\right)\right). \end{aligned}$$

is in fact a bijection, which can be realized as follows. Since the set $\{1, \alpha_i, \ldots, \alpha_i^{p_i-1}\}$ forms a basis of E over F_i and $v_i h(\alpha_i) \neq 0$, the set $\{v_i h(\alpha_i), v_i h(\alpha_i)\alpha_i, \ldots, v_i h(\alpha_i)\alpha_i^{p_i-1}\}$ also forms a basis. Let $\{\theta_0, \theta_1, \ldots, \theta_{p_i-1}\}$ be the dual basis of $\{v_i h(\alpha_i), v_i h(\alpha_i)\alpha_i, \ldots, v_i h(\alpha_i)\alpha_i^{p_i-1}\}$, i.e.,

$$\mathrm{tr}_i(v_i h(\alpha_i)\alpha_i^s \theta_j) = \begin{cases} 0, & \text{if } s \neq j \\ 1, & \text{if } s = j \end{cases} \quad \text{for all } s, j \in \{0, 1, \ldots, p_i - 1\}.$$

The value c_i can now be found as follows:

$$c_i = \sum_{s=0}^{p_i-1} \mathrm{tr}_i(v_i h(\alpha_i)\alpha_i^s c_i)\theta_s.$$

(this is the essence of the repair scheme proposed in [26]).

The presented arguments constitute a linear repair scheme of the node $c_i, i = 1, \ldots m$ over F_i. The information downloaded from each of the helper nodes consists of one element of F_i, or, in other words, the $(1/p_i)$th proportion of the contents of each node. This shows that node i admits nontrivial optimal repair. The proof is thereby complete. □

Example 5.1. *Take $q = 5, k = 3, r = 5$. We have $\pi(r) = 3$ and $p_1 = 2, p_2 = 3, p_3 = 5$. Let us construct an $(8, 3)$ RS code over the field $E = \mathbb{F}_{5^{30}}$, where the first 3 nodes admit nontrivial optimal repair schemes. Let α be a primitive element of E. Choose the set $\Omega = \{\alpha_1, \ldots, \alpha_8\}$ as follows:*

$$\alpha_1 = \alpha^{\frac{5^{30}-1}{5^2-1}}, \alpha_2 = \alpha^{\frac{5^{30}-1}{5^3-1}}, \alpha_3 = \alpha^{\frac{5^{30}-1}{5^5-1}}, \alpha_4 = 0, \alpha_5 = 1, \alpha_6 = 2, \alpha_7 = 3, \alpha_8 = 4.$$

The number of helper nodes for the first 3 nodes is $(d_1, d_2, d_3) = (4, 5, 7)$. It is easy to verify that for any subset $A \subseteq \{1, 2, 3\}$

$$\mathbb{F}_5(\alpha_i : i \in A) = \mathbb{F}_{m_A}, \text{ where } m_A = 5^{(\prod_{i \in A} p_i)}.$$

The code $\mathcal{C}$ constructed in the above proof is given by $\mathcal{C} = \mathrm{RS}_E(8, 3, \Omega)$. Let us address the task of repairing c_3 from all the remaining 7 helper nodes with repair bandwidth achieving the cut-set bound. Let $\mathcal{C}^\perp = \mathrm{GRS}_E(8, 5, \Omega, v)$, where $v = (v_1, \ldots, v_8) \in$

$(E^*)^8$. *We download the value* $\mathrm{tr}_{E/\mathbb{F}_{5^6}}(v_j c_j)$ *from each helper node* $c_j, j \neq 3$. *Since* $[E : \mathbb{F}_{5^6}] = p_3$, *this amounts to downloading exactly a* $1/p_3 = (1/5)$*-th fraction of the information stored at each helper node, which achieves the cut-set bound. The value of* c_3 *can be found from the downloaded information using the following* 5 *equations:*

$$\mathrm{tr}_{E/\mathbb{F}_{5^6}}(\alpha_3^s v_3 c_3) = -\sum_{j\neq 3} \mathrm{tr}_{E/\mathbb{F}_{5^6}}(\alpha_j^s v_j c_j) = -\sum_{j\neq 3} \alpha_j^s \,\mathrm{tr}_{E/\mathbb{F}_{5^6}}(v_j c_j), \quad s = 0, \ldots, 4.$$

Indeed, the downloaded symbols suffice to recover the vector $(\mathrm{tr}_{E/\mathbb{F}_{5^6}}(\alpha_3^s v_3 c_3), s = 0, \ldots, 4)$, *and therefore also suffice to repair the symbol* c_3.

5.3 A family of RS codes achieving the cut-set bound

In this section we develop the ideas discussed above and construct RS codes achieving the cut-set bound with nontrivial optimal repair of all nodes. More precisely, given any positive integers $k < d \leq n-1$, we explicitly construct an (n, k) RS code $\mathcal{C}$ achieving the cut-set bound for the repair of *any* single node from *any* d helper nodes. In other words, $\mathcal{C}$ is an (n, k) MSR code with $(1, d)$-optimal repair property.

Let $\mathbb{F}_p$ be a finite field of prime order (for simplicity we can take $p = 2$). Denote $s := d - k + 1$ and let $p_1, \ldots, p_n$ be n distinct primes such that

$$p_i \equiv 1 \bmod s \quad \text{for all } i = 1, 2, \ldots, n. \tag{5.6}$$

According to Dirichlet's theorem, there are infinitely many such primes. For $i = 1, \ldots, n$, let α_i be an element of degree p_i over $\mathbb{F}_p$, i.e., $[\mathbb{F}_p(\alpha_i) : \mathbb{F}_p] = p_i$, and define

$$\mathbb{F} := \mathbb{F}_p(\alpha_1, \ldots, \alpha_n). \tag{5.7}$$

Note that for any subset of indices $A \subseteq [n]$, the field $\mathbb{F}_p(\{\alpha_i : i \in A\})$ is an extension of $\mathbb{F}_p$ of degree $\prod_{i\in A} p_i$, and in particular, $\mathbb{F}$ has degree $\prod_{i=1}^n p_i$ over $\mathbb{F}_p$. Next, we define n distinct subfields F_i of the field $\mathbb{F}$ and one extension field $\mathbb{K}$ of $\mathbb{F}$.

1. For $i = 1, \ldots, n$, define $F_i = \mathbb{F}_p(\{\alpha_j : j \neq i\})$. Note that $\mathbb{F} = F_i(\alpha_i)$ and $[\mathbb{F} : F_i] = p_i$.

2. The field $\mathbb{K}$ is defined to be the degree-s extension of the field $\mathbb{F}$, i.e. there exists an element $\beta \in \mathbb{K}$ of degree s over $\mathbb{F}$ such that $\mathbb{K} = \mathbb{F}(\beta)$. We also have $[\mathbb{K} : F_i] = sp_i$ for all i.

We are ready to construct a family of RS codes that can be optimally repaired for each node. The set $\alpha_1, \ldots, \alpha_n$ serves as the set of evaluation points of the code.

The following theorem is the main result of this section.

Theorem 5.4. *Let* k, n, d *be any positive integers such that* $k < d < n$. *Let* $\Omega = \{\alpha_1, \ldots, \alpha_n\}$, *where* $\alpha_i, i = 1, \ldots, n$ *is an element of degree* p_i *over* $\mathbb{F}_p$ *and* p_i *is the* i*th smallest prime that satisfies* (5.6). *The code* $\mathcal{C} := \mathrm{RS}_{\mathbb{K}}(n, k, \Omega)$ *achieves the cut-set bound for the repair of any single node from any* d *helper nodes. In other words,* $\mathcal{C}$ *is an* (n, k) *MSR code with* $(1, d)$*-optimal repair property.*

Remark 5.2. *The code constructions in this chapter rely on the condition of the form $\alpha_i \notin \mathbb{F}_q(\alpha_j, j \neq i), i = 1, \ldots, n$ (in this section we also require that the extension degree $[\mathbb{F} : F_i] \equiv 1$ mod $s, i = 1, \ldots, n$). The most efficient way to accomplish this in terms of the value of sub-packetization l is to take the extension degrees to be the smallest (distinct) primes, and this is the underlying idea behind the code constructions presented in this chapter.*

Proof. Our repair scheme of the i-th node is performed over the field F_i. More specifically, for every $i \in [n]$, we explicitly construct a vector space S_i over the field F_i such that

$$\dim_{F_i} S_i = p_i, \quad S_i + S_i\alpha_i + \cdots + S_i\alpha_i^{s-1} = \mathbb{K}, \tag{5.8}$$

where $S_i\alpha := \{\gamma\alpha : \gamma \in S_i\}$, and the operation $+$ is the Minkowski sum of sets, $T_1 + T_2 := \{\gamma_1 + \gamma_2 : \gamma_1 \in T_1, \gamma_2 \in T_2\}$. Note that the sum in (5.8) is in fact a direct sum since the dimension of each summand is p_i, and $[\mathbb{K} : F_i] = sp_i$. We will describe a construction of S_i and prove that S_i satisfies (5.8) in Lemma 5.5 later in this section. For now let us assume that we have such vector spaces $S_i, i = 1, 2, \ldots, n$ and continue the proof of the theorem.

Suppose that we want to repair the i-th node from a subset $\mathcal{R} \subseteq [n]\backslash\{i\}$ of $|\mathcal{R}| = d$ helper nodes. Let $h(x)$ be the annihilator polynomial of the set $\{\alpha_j : j \in [n] \setminus (\mathcal{R} \cup \{i\})\}$, i.e.,

$$h(x) = \prod_{j \in [n]\backslash(\mathcal{R}\cup\{i\})} (x - \alpha_j). \tag{5.9}$$

By (2.2) the dual code of $\mathcal{C}$ is $\mathcal{C}^\perp = \mathrm{GRS}_{\mathbb{K}}(n, n-k, \Omega, v)$ where the coefficients $v = (v_1, \ldots, v_n) \in (\mathbb{K}^*)^n$ are nonzero. Clearly, $\deg(x^t h(x)) \leq s - 1 + n - (d+1) < n - k$ for all $t = 0, 1, \ldots, s-1$, so for any such t we have

$$(v_1\alpha_1^t h(\alpha_1), \ldots, v_n\alpha_n^t h(\alpha_n)) \in \mathcal{C}^\perp. \tag{5.10}$$

These s dual codewords will be used to recover the i-th coordinate. Let $c = (c_1, \ldots, c_n) \in \mathcal{C}$ be a codeword, and let us construct a repair scheme for the coordinate (node) c_i using the values $\{c_j : j \in \mathcal{R}\}$. Rewrite (5.10) as follows:

$$\sum_{j=1}^{n} v_j\alpha_j^t h(\alpha_j)c_j = 0 \text{ for all } t = 0, \ldots, s-1. \tag{5.11}$$

Let $e_1, \ldots, e_{p_i}$ be an arbitrary basis of the subspace S_i over the field F_i. From (5.11) we obtain the following system of sp_i equations:

$$\sum_{j=1}^{n} e_m v_j\alpha_j^t h(\alpha_j)c_j = 0, \quad t = 0, \ldots, s-1; m = 1, \ldots, p_i.$$

Let $\mathrm{tr}_i := \mathrm{tr}_{\mathbb{K}/F_i}$ be the trace map to the subfield F_i. From the last set of equations we have

$$\sum_{j=1}^{n} \mathrm{tr}_i(e_m v_j \alpha_j^t h(\alpha_j) c_j) = 0 \text{ for all } t = 0, \ldots, s-1 \text{ and all } m = 1, \ldots, p_i, \tag{5.12}$$

Arguing as in (5.5), let us write (5.12) in the following form: for all $t = 0, \ldots, s-1$ and all $m = 1, \ldots, p_i$,

$$\begin{aligned} \mathrm{tr}_i(e_m \alpha_i^t v_i h(\alpha_i) c_i) &= -\sum_{j \neq i} \mathrm{tr}_i(e_m v_j \alpha_j^t h(\alpha_j) c_j) \\ &= -\sum_{j \in \mathcal{R}} \mathrm{tr}_i(e_m v_j \alpha_j^t h(\alpha_j) c_j) \\ &= -\sum_{j \in \mathcal{R}} \alpha_j^t h(\alpha_j) \, \mathrm{tr}_i(e_m v_j c_j), \end{aligned} \tag{5.13}$$

where the second equality follows from (5.9) and the third follows from the fact that the trace mapping tr_i is F_i-linear, and that $\alpha_j \in F_i$ for all $j \neq i$.

As before, to recover c_i, we download the following p_i symbols of F_i from each helper node $c_j, j \in \mathcal{R}$:

$$\mathrm{tr}_i(e_m v_j c_j) \text{ for } m = 1, \ldots, p_i. \tag{5.14}$$

These field elements suffice to recover the node c_i. Indeed, according to (5.13), we can calculate the values of $\mathrm{tr}_i(e_m \alpha_i^t v_i h(\alpha_i) c_i)$ for all $t = 0, \ldots, s-1$ and all $m = 1, \ldots, p_i$ from the set of elements in (5.14). By definition, $e_1, \ldots, e_{p_i}$ is a basis of the subspace S_i over the field F_i. According to (5.8), $\mathbb{K} = S_i + S_i\alpha_i + \cdots + S_i\alpha_i^{s-1}$. Therefore, the set $\{e_m \alpha_i^t : t = 0, \ldots, s-1;\ m = 1, \ldots, p_i\}$ forms a basis of $\mathbb{K}$ over F_i and so does the set $\{e_m \alpha_i^t v_i h(\alpha_i) : t = 0, \ldots, s-1;\ m = 1, \ldots, p_i\}$ (recall that $v_i \cdot h(\alpha_i) \neq 0$). Hence the mapping

$$\begin{aligned} &\mathbb{K} \to F_i^{sp_i} \\ &\gamma \mapsto (\mathrm{tr}_i(e_m \alpha_i^t v_i h(\alpha_i) \gamma), m = 1, \ldots, p_i; t = 0, \ldots, s-1). \end{aligned}$$

is a bijection. This means that c_i is uniquely determined by the set of values

$$\{\mathrm{tr}_i(e_m \alpha_i^t v_i h(\alpha_i) c_i), m = 1, \ldots, p_i; t = 0, \ldots, s-1\},$$

validating our repair scheme.

It is also clear that the construction meets the cut-set bound. Indeed, $c_j \in \mathbb{K}$ for all j and $[\mathbb{K} : F_i] = sp_i$, so the amount of information required from each helper node (5.14) is exactly $(1/s)$th fraction of its contents.

This completes the proof of Theorem 5.4. □

In the proof above we assumed the existence of the vector space S_i that satisfies (5.8) for all $i \in [n]$. In the next lemma we construct such a space and establish its properties.

For a vector space V over a field F and a set of vectors $A = (a_1, \ldots, a_l) \subset V$, let $\mathrm{Span}_F(A) = \{\sum_{i=1}^{l} \gamma_i a_i, \gamma_i \in F\}$ be the span of A over F.

Lemma 5.5. *Let β be a generating element of $\mathbb{K}$ over $\mathbb{F} = \mathbb{F}_p(\alpha_1, \ldots, \alpha_n)$. Given $i \in [n]$, define the following vector spaces over F_i:*

$$S_i^{(1)} = \mathrm{Span}_{F_i}\left(\beta^u \alpha_i^{u+qs}, u = 0, 1, \ldots, s-1; q = 0, 1, \ldots, \tfrac{p_i-1}{s} - 1\right)$$

$$S_i^{(2)} = \mathrm{Span}_{F_i}\left(\sum_{t=0}^{s-1} \beta^t \alpha_i^{p_i-1}\right)$$

$$S_i = S_i^{(1)} + S_i^{(2)}.$$

Then

$$\dim_{F_i} S_i = p_i, \quad S_i + S_i\alpha_i + \cdots + S_i\alpha_i^{s-1} = \mathbb{K}.$$

Proof. Let $K := S_i + S_i\alpha_i + \cdots + S_i\alpha_i^{s-1}$. If $K = \mathbb{K}$, then $\dim_{F_i} S_i = p_i$ easily follows. Indeed, by definition $\dim_{F_i} S_i \leq p_i$. On the other hand, $[\mathbb{K} : F_i] = sp_i$ and $K = \mathbb{K}$ together imply that $\dim_{F_i} S_i \geq p_i$.

Let us prove that $K = \mathbb{K}$. Clearly K is a vector space over F_i, and $K \subseteq \mathbb{K}$. Let us show the reverse inclusion, namely that $\mathbb{K} \subseteq K$. To prove this, recall that $\mathbb{K}$ is a vector space of dimension s over $\mathbb{F}$ (see (5.7)), and the set $1, \beta, \ldots, \beta^{s-1}$ forms a basis, i.e., $\mathbb{K} = \oplus_{u=0}^{s-1} \beta^u \mathbb{F}$. Thus, the lemma will be proved if we show that $\beta^u \mathbb{F} \subseteq K$ for all $u = 0, 1, \ldots, s-1$. To prove this inclusion we will use induction on u.

For the induction base, let $u = 0$. In this case, we have $\alpha_i^{qs} \in S_i^{(1)}$ for all $0 \leq q < \frac{p_i-1}{s}$. Therefore $\alpha_i^{qs+j} \in S_i^{(1)}\alpha_i^j$ for all $0 \leq q < \frac{p_i-1}{s}$. As a result, $\alpha_i^{qs+j} \in K$ for all $0 \leq q < \frac{p_i-1}{s}$ and all $0 \leq j \leq s-1$. In other words,

$$\alpha_i^t \in K, \ t = 0, 1, \ldots, p_i - 2. \tag{5.15}$$

Next we show that also $\alpha_i^{p_i-1} \in K$. For every $t = 1, \ldots, s-1$ we have $0 \leq \lfloor \frac{p_i-1-t}{s} \rfloor < \frac{p_i-1}{s}$. As a result,

$$\beta^t \alpha_i^{t+\lfloor \frac{p_i-1-t}{s} \rfloor s} \in S_i^{(1)}, \ t = 1, \ldots, s-1.$$

We obtain, for each $t = 1, \ldots, s-1$,

$$\beta^t \alpha_i^{p_i-1} = \beta^t \alpha_i^{t+\lfloor \frac{p_i-1-t}{s} \rfloor s} \alpha_i^{p_i-1-t-\lfloor \frac{p_i-1-t}{s} \rfloor s} \in S_i \alpha_i^{p_i-1-t-\lfloor \frac{p_i-1-t}{s} \rfloor s} \subseteq K.$$

At the same time,

$$\sum_{t=0}^{s-1} \beta^t \alpha_i^{p_i-1} \in S_i^{(2)} \subseteq K.$$

The last two statements together imply that

$$\alpha_i^{p_i-1} = \sum_{t=0}^{s-1} \beta^t \alpha_i^{p_i-1} - \sum_{t=1}^{s-1} \beta^t \alpha_i^{p_i-1} \in K.$$

Combining this with (5.15), we conclude that $\alpha_i^t \in K$ for all $t = 0, 1, \ldots, p_i - 1$. Recall that $1, \alpha_i, \ldots, \alpha_i^{p_i-1}$ is a basis of $\mathbb{F}$ over F_i, and that K is a vector space over F_i, so $\mathbb{F} \subseteq K$. This establishes the induction base.

Now let us fix $u \geq 1$ and let us assume that $\beta^{u'}\mathbb{F} \subseteq K$ for all $u' < u$. To prove the induction step, we need to show that $\beta^u\mathbb{F} \subseteq K$. Mimicking the argument that led to (5.15), we can easily show that

$$\beta^u \alpha_i^{u+t} \in K, \ t = 0, 1, \ldots, p_i - 2. \tag{5.16}$$

Let us show that (5.16) is also true for $t = p_i - 1$, i.e., that $\beta^u \alpha_i^{u+p_i-1} \in K$. For every $1 \leq t \leq s-1-u$, we have $0 \leq \lfloor \frac{p_i-1-t}{s} \rfloor < \frac{p_i-1}{s}$. As a result,

$$\beta^{u+t} \alpha_i^{u+t+\lfloor \frac{p_i-1-t}{s} \rfloor s} \in S_i^{(1)}, \ t = 1, \ldots, s-1-u.$$

Therefore, for all such t

$$\beta^{u+t}\alpha_i^{u+p_i-1} = \beta^{u+t}\alpha_i^{u+t+\lfloor \frac{p_i-1-t}{s} \rfloor s} \alpha_i^{p_i-1-t-\lfloor \frac{p_i-1-t}{s} \rfloor s} \in S_i \alpha_i^{p_i-1-t-\lfloor \frac{p_i-1-t}{s} \rfloor s} \subseteq K \tag{5.17}$$

By the induction hypothesis, $\beta^{u'}\mathbb{F} \subseteq K$ for all $u' = 0, 1, \ldots, u-1$. As a result,

$$\beta^{u'} \alpha_i^{u+p_i-1} \in K, \ u' = 0, 1, \ldots, u-1. \tag{5.18}$$

At the same time,

$$\sum_{t=0}^{s-1} \beta^t \alpha_i^{u+p_i-1} = \Big(\sum_{t=0}^{s-1} \beta^t \alpha_i^{p_i-1} \Big) \alpha_i^u \in S_i^{(2)} \alpha_i^u \subseteq K. \tag{5.19}$$

Combining (5.17), (5.18) and (5.19), we obtain

$$\beta^u \alpha_i^{u+p_i-1} = \sum_{t=0}^{s-1} \beta^t \alpha_i^{u+p_i-1} - \sum_{u'=0}^{u-1} \beta^{u'} \alpha_i^{u+p_i-1} - \sum_{t=1}^{s-1-u} \beta^{u+t} \alpha_i^{u+p_i-1} \in K.$$

Now on account of (5.16) we can conclude that $\beta^u \alpha_i^{u+t} \in K$ for all $t = 0, 1, \ldots, p_i - 1$. Therefore, $\beta^u \mathbb{F} \subseteq K$. This establishes the induction step and completes the proof of the lemma. □

The value of sub-packetization of the constructed codes is given in the following obvious proposition.

Proposition 5.6. *The sub-packetization of our construction is* $l = [\mathbb{K} : \mathbb{F}_p] = s \prod_{i=1}^n p_i$, *where* p_i*'s are the smallest* n *distinct primes satisfying* (5.6).

The proof follows immediately from the fact that the repair of the i-th coordinate is performed over the field F_i, so the repair field of our construction is $\cap_{i=1}^n F_i = \mathbb{F}_p$. To estimate the asymptotics of l for $n \to \infty$, recall that our discussion of Dirichlet's prime number theorem in Sec. 5.1.4 above implies that, for fixed s, $l = e^{(1+o(1))n \log n}$. This proves the upper bound in (5.2).

5.4 A lower bound on the sub-packetization of scalar linear MSR codes

In this section we prove a lower bound on the sub-packetization value l of (n, k) scalar linear MSR codes, which implies that $l \geq e^{(1+o(1))k \log k}$. In contrast, for MSR array codes, a much smaller sub-packetization value $l = r^{\lceil n/(r+1) \rceil}$ is achievable [68]. This shows that limiting oneself to scalar linear codes necessarily leads to a much larger sub-packetization, and constructing such codes in real storage systems is even less feasible than their array code counterparts. The main result of this section is the following theorem:

Theorem 5.7. *Let $F = \mathbb{F}_q$ and $E = \mathbb{F}_{q^l}$ for a prime power q. Let d be an integer between $k+1$ and $n-1$. Let $\mathcal{C} \subseteq E^n$ be an (n, k) scalar linear MDS code with a linear repair scheme over F. Suppose that the repair bandwidth of the scheme achieves the cut-set bound (2.1) with equality for the repair of any single node from any d helper nodes. Then the sub-packetization l is at least*

$$l \geq \prod_{i=1}^{k-1} p_i$$

where p_i is the i-th smallest prime.

As discussed above in Sec. 5.1.4, this theorem implies the asymptotic lower bound $l \geq e^{(1+o(1))k \log k}$.

In the proof of Theorem 5.7, we will need the following auxiliary lemmas.

Lemma 5.8. (Subfield criterion [34, Theorem 2.6]) *Each subfield of the field $\mathbb{F}_{p^n}$ is of order p^m, where $m|n$. For every positive divisor m of n there exists a unique subfield of $\mathbb{F}_{p^n}$ that contains p^m elements.*

Lemma 5.9. *Let E be an extension field of $\mathbb{F}_q$ and let $\alpha_1, \ldots, \alpha_n \in E$. Then*

$$[\mathbb{F}_q(\alpha_1, \ldots, \alpha_n) : \mathbb{F}_q] = \operatorname{lcm}(d_1, \ldots, d_n),$$

where $d_i = [\mathbb{F}_q(\alpha_i) : \mathbb{F}_q]$.

Proof: Obvious.

Lemma 5.10. *Let $a_1, a_2, \ldots, a_n \in F^m$ and $b_1, b_2, \ldots, b_n \in F^m$ be two sets of vectors over a field F, and let A and B denote their spans over F. Let $c_i = a_i + b_i, i = 1, \ldots, n$ then*

$$\dim_F(c_1, \ldots, c_n) \leq \dim A + \dim B. \tag{5.20}$$

The lemma follows immediately from the fact that, for any two subspaces A and B of a linear space,

$$\dim(A + B) + \dim(A \cap B) = \dim A + \dim B. \tag{5.21}$$

In the next lemma $\mathcal{S}_F(\cdot)$ refers to the row space of the matrix argument over the field F.

Lemma 5.11. *Let E be an extension of a finite field F of degree l. Let $A = (a_{i,j})$ be an $m \times n$ matrix over E. Then*

$$\dim(\mathcal{S}_F(A)) \leq \sum_{j=1}^{n} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}). \tag{5.22}$$

Moreover, if (5.22) holds with equality, then for every $\mathcal{J} \subseteq [n]$,

$$\dim(\mathcal{S}_F(A_{\mathcal{J}})) = \sum_{j \in \mathcal{J}} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}) \tag{5.23}$$

where $A_{\mathcal{J}}$ is the restriction of A to the columns with indices in $\mathcal{J}$.

Proof. Inequality (5.22) is an immediate consequence of Lemma 5.10. Indeed, suppose that $n = 2$ and view the ith row of A as the sum of two 2-dimensional vectors over E, namely $(a_{i,1}|0)$ and $(0|a_{i,2})$, $i = 1, \ldots, m$; then (5.22) is the same as (5.20). The extension to $n > 2$ follows by straightforward induction.

Now let us prove the second part of the claim. Suppose that

$$\dim(\mathcal{S}_F(A)) = \sum_{j=1}^{n} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}).$$

Then for every $\mathcal{J} \subseteq [n]$,

$$\begin{aligned} &\sum_{j \in \mathcal{J}} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}) + \sum_{j \in \mathcal{J}^c} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}) \\ &= \dim(\mathcal{S}_F(A)) \leq \dim(\mathcal{S}_F(A_{\mathcal{J}})) + \dim(\mathcal{S}_F(A_{\mathcal{J}^c})). \end{aligned}$$

But according to (5.22),

$$\begin{aligned} \dim(\mathcal{S}_F(A_{\mathcal{J}})) &\leq \sum_{j \in \mathcal{J}} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}), \\ \dim(\mathcal{S}_F(A_{\mathcal{J}^c})) &\leq \sum_{j \in \mathcal{J}^c} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}). \end{aligned}$$

Therefore

$$\dim(\mathcal{S}_F(A_{\mathcal{J}})) = \sum_{j \in \mathcal{J}} \dim_F(a_{1,j}, a_{2,j}, \ldots, a_{m,j}).$$

This completes the proof of the lemma. □

Now we are ready to prove Theorem 5.7.

Proof of Theorem 5.7: Let $\mathcal{C}$ be an (n, k) MSR code with $(1, d)$-optimal repair property. By puncturing the code $\mathcal{C}$ to any $d + 1$ coordinates, we obtain a $(d + 1, k)$ MSR code with $(1, d)$-optimal repair property. Therefore without loss of generality below we assume that $d = n - 1$.

Let $H = [M|I_r]$ be the parity-check matrix of the code $\mathcal{C}$ over E, written in systematic form, where M is an $r \times k$ matrix and I_r is the $r \times r$ identity matrix. Let h_{ij} be the entry of H in position (i, j). Since $\mathcal{C}$ is an MDS code, every square submatrix of M is invertible. In particular, every entry of M is nonzero, so without loss of generality we may assume that $h_{1,j} = 1, j = 1, 2, \ldots, k$. Since $d \geq k + 1$, we also have $n \geq k + 2$, and therefore H contains at least two rows.

The theorem will follow from the following claim.

Claim 5.1. *For $j = 1, \ldots, k-1$ define $\alpha_j := \frac{h_{2,j}}{h_{2,k}}$. Then for every $j = 1, \ldots, k-1$,*

$$\alpha_j \notin \mathbb{F}_q\big(\{\alpha_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\}\}\big). \tag{5.24}$$

In other words, α_j is not generated by the remaining α_i's over $\mathbb{F}_q$.

We first show that this claim indeed implies the theorem. Let $d_i = [\mathbb{F}_q(\alpha_i) : \mathbb{F}_q]$ be the degree of the field extension generated by α_i. We prove by contradiction that for all $j = 1, 2, \ldots, k-1$, d_j does not divide $\operatorname{lcm}(d_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\})$. Suppose the contrary, i.e., that there is a j such that $d_j | \operatorname{lcm}(d_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\})$. According to Lemma 5.9,

$$[\mathbb{F}_q\big(\{\alpha_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\}\}\big) : \mathbb{F}_q] = \operatorname{lcm}(d_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\}).$$

Then by Lemma 5.8, there is a subfield

$$F_j \subseteq \mathbb{F}_q\big(\{\alpha_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\}\}\big) \tag{5.25}$$

such that $[F_j : \mathbb{F}_q] = d_j$. Notice that $E = \mathbb{F}_{q^l}$ contains all $\alpha_u, u = 1, 2, \ldots, k-1$. So both F_j and $\mathbb{F}_q(\alpha_j)$ are subfields of E, and they have the same order q^{d_j}. Consequently, $\mathbb{F}_q(\alpha_j) = F_j$. Then from (5.25) we conclude that $\alpha_j \in \mathbb{F}_q\big(\{\alpha_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\}\}\big)$, which contradicts (5.24). Thus, our assumption is wrong, and $d_j \nmid \operatorname{lcm}(d_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\})$. As an immediate corollary,

$$l = [E : \mathbb{F}_q] \geq [\mathbb{F}_q(\{\alpha_i : i = 1, \ldots, k-1\}) : \mathbb{F}_q] = \operatorname{lcm}(d_1, \ldots, d_{k-1}) \geq \prod_{i=1}^{k-1} p_i.$$

Thus we have shown that Claim 5.1 indeed implies the theorem. Now let us prove the claim.

Proof of the Claim 5.1: Consider the repair of the j-th node of the code $\mathcal{C}$ for some $j \in \{1, 2, \ldots, k-1\}$. Since $\mathcal{C}$ can be viewed as an $(n, k, n-1, l)$ MSR code with a linear repair scheme over $\mathbb{F}_q$, node c_j can be repaired by downloading $(n-1)l/r$ symbols of $\mathbb{F}_q$ from all the remaining nodes $\{c_i : i \in [n] \setminus \{j\}\}$, where $r = n - k$. Therefore by Theorem 5.1, there exist l codewords

$$(c_{t,1}, c_{t,2}, \ldots, c_{t,n}) \in \mathcal{C}^{\perp}, t = 1, 2, \ldots, l$$

such that

$$\dim_{\mathbb{F}_q}(c_{1,j}, c_{2,j}, \ldots, c_{l,j}) = l, \text{ and} \tag{5.26}$$

$$\sum_{i\neq j} \dim_{\mathbb{F}_q}(c_{1,i}, c_{2,i}, \ldots, c_{l,i}) = \frac{(n-1)l}{r}. \tag{5.27}$$

Since H is a generator matrix of $\mathcal{C}^{\perp}$, for each $t = 1, 2, \ldots, l$ there is a column vector $b_t \in E^r$ such that $(c_{t,1}, c_{t,2}, \ldots, c_{t,n}) = b_t^T H$. We define an $l \times r$ matrix B over the field E as $B = [b_1 b_2 \ldots b_l]^T$. We claim that the $\mathbb{F}_q$-rank of the row space of B is l. Indeed, assume the contrary, then there exists a nonzero vector $w \in \mathbb{F}_q^l$ such that $wB = 0$. Therefore,

$$wBH = w \begin{bmatrix} c_{1,1} & c_{1,2} & \ldots & c_{1,n} \\ c_{2,1} & c_{2,2} & \ldots & c_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{l,1} & c_{l,2} & \ldots & c_{l,n} \end{bmatrix} = 0.$$

This implies that $w(c_{1,j}, c_{2,j}, \ldots, c_{l,j})^T = 0$, contradicting (5.26). Thus we conclude that B has l linearly independent rows over $\mathbb{F}_q$.

Now we want to show that there exists an $l \times l$ invertible matrix A over $\mathbb{F}_q$ such that the matrix AB is an $r \times r$ block-diagonal matrix $\mathrm{Diag}(a_1, \ldots, a_r)$, where each block a_i is formed of a column vector of length $\frac{l}{r}$. In other words, by performing elementary row operations over $\mathbb{F}_q$, B can be transformed into an $r \times r$ block-diagonal matrix $\mathrm{Diag}(a_1, \ldots, a_r)$. Indeed, for $i \in [n]$, let h_i be the i-th column of the matrix H, and define

$$t_i = \dim_{\mathbb{F}_q}(Bh_i) = \dim_{\mathbb{F}_q}(c_{1,i}, c_{2,i}, \ldots, c_{l,i}).$$

By (5.27), we have

$$\sum_{i\neq j}^{n} t_i = \frac{(n-1)l}{r}. \tag{5.28}$$

Since H generates an (n, r) MDS code, for any subset of indices $\mathcal{J} \subseteq [n]$ of size $|\mathcal{J}| = r$, the matrix $H_{\mathcal{J}}$ is of full rank. Therefore, the $l \times r$ matrix $BH_{\mathcal{J}}$ satisfies the conditions

$$l = \dim(\mathcal{S}_{\mathbb{F}_q}(B)) = \dim(\mathcal{S}_{\mathbb{F}_q}(BH_{\mathcal{J}})) \leq \sum_{i\in\mathcal{J}} \dim_{\mathbb{F}_q}(Bh_i), \tag{5.29}$$

where the last inequality follows from Lemma 5.11. Summing both sides of (5.29) over all subsets $\mathcal{J} \subseteq [n]\backslash\{j\}$ of size $|\mathcal{J}| = r$, we obtain that

$$\begin{aligned} l\binom{n-1}{r} &\leq \sum_{\substack{\mathcal{J}\subseteq[n]\backslash\{j\} \\ |\mathcal{J}|=r}} \sum_{i\in\mathcal{J}} \dim_{\mathbb{F}_q}(Bh_i) \\ &= \binom{n-2}{r-1} \sum_{i\neq j} t_i \\ &\overset{(5.28)}{=} \binom{n-2}{r-1} \frac{(n-1)l}{r} \\ &= l\binom{n-1}{r}, \end{aligned} \tag{5.30}$$

This implies that the inequality above is in fact an equality, and therefore, on account of (5.29) for every subset $\mathcal{J} \subseteq [n]\backslash\{j\}, |\mathcal{J}| = r$ we have

$$l = \sum_{i\in\mathcal{J}} \dim_{\mathbb{F}_q}(Bh_i) = \sum_{i\in\mathcal{J}} t_i. \tag{5.31}$$

From (5.31) we obtain that for all $i \in [n] \setminus \{j\}$

$$\dim_{\mathbb{F}_q}(Bh_i) = t_i = l/r. \tag{5.32}$$

Moreover, since (5.29) holds with equality, we can use the second part of Lemma 5.11 to claim that, for $\mathcal{J} \subseteq [n]\backslash\{j\}$ of size $|\mathcal{J}| \leq r$,

$$\dim(\mathcal{S}_{\mathbb{F}_q}(BH_{\mathcal{J}})) = \sum_{i\in\mathcal{J}} \dim_{\mathbb{F}_q}(Bh_i) = \frac{|\mathcal{J}|l}{r}. \tag{5.33}$$

Let us take $\mathcal{J}$ to be a subset of $\{k+1, k+2, \ldots, n\}$. Since the last r columns of H form an identity matrix, (5.33) becomes

$$\dim(\mathcal{S}_{\mathbb{F}_q}(B_{\mathcal{J}})) = \frac{|\mathcal{J}|l}{r} \text{ for all } \mathcal{J} \subseteq [r] \text{ with size } |\mathcal{J}| \leq r. \tag{5.34}$$

Now we are ready to prove that by performing elementary row operations over $\mathbb{F}_q$, B can be transformed into an $r \times r$ block diagonal matrix $\mathrm{Diag}(a_1, \ldots, a_r)$, where each block a_i is a single column vector of length $\frac{l}{r}$. We proceed by induction. More specifically, we prove that for $i = 1, 2, \ldots, r$, we can use elementary row operations over $\mathbb{F}_q$ to transform the first i columns of B into the following form:

$$\begin{bmatrix} a_1 & 0 & \ldots & 0 \\ 0 & a_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & a_i \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix},$$

where each $\mathbf{0}$ in the last row of the above matrix is a column vector of length $l(1 - \frac{i}{r})$.

Let $i = 1$. According to (5.34), each column of B has dimension l/r over $\mathbb{F}_q$. Thus the induction base holds trivially. Now assume that there is an $l \times l$ invertible matrix A over $\mathbb{F}_q$ such that

$$AB_{[i-1]} = \begin{bmatrix} a_1 & 0 & \ldots & 0 \\ 0 & a_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & a_{i-1} \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix},$$

where each $\mathbf{0}$ in the last row of this matrix is a column vector of length $l(1 - \frac{i-1}{r})$. Let us write the i-th column of AB as $(v_1, v_2, \ldots, v_l)^T$. Since each column of B has dimension

l/r over $\mathbb{F}_q$, $(v_1, v_2, \ldots, v_l)^T$ also has dimension l/r over $\mathbb{F}_q$. Since the last $l(1 - \frac{i-1}{r})$ rows of the matrix $AB_{[i-1]}$ are all zero, we can easily deduce that

$$\dim(\mathcal{S}_{\mathbb{F}_q}(AB_{[i]})) \leq \frac{i-1}{r} l + \dim_{\mathbb{F}_q}(v_{(i-1)l/r+1}, v_{(i-1)l/r+2}, \ldots, v_l).$$

By (5.34), $\dim(\mathcal{S}_{\mathbb{F}_q}(AB_{[i]})) = \dim(\mathcal{S}_{\mathbb{F}_q}(B_{[i]})) = \frac{il}{r}$. As a result,

$$\dim_{\mathbb{F}_q}(v_{(i-1)l/r+1}, v_{(i-1)l/r+2}, \ldots, v_l) \geq l/r = \dim_{\mathbb{F}_q}(v_1, v_2, \ldots, v_l).$$

In other words, $(v_{(i-1)l/r+1}, v_{(i-1)l/r+2}, \ldots, v_l)$ contains a basis of the set $(v_1, v_2, \ldots, v_l)$ over $\mathbb{F}_q$. This implies that we can use elementary row operations on the matrix AB to eliminate all the nonzero entries v_m for $m \leq (i-1)l/r$, and thus obtain the desired block-diagonal structure for the first i columns. This establishes the induction step.

We conclude that there exists an $l \times l$ invertible matrix A over $\mathbb{F}_q$ such that $AB = \mathrm{Diag}(a_1, \ldots, a_r)$, where each block a_i is a single column vector of length $\frac{l}{r}$. For $u \in [r]$, let A_u be the vector space spanned by the entries of a_u over $\mathbb{F}_q$. According to (5.32), for all $i \in [n] \setminus \{j\}$

$$\dim_{\mathbb{F}_q}(ABh_i) = \dim_{\mathbb{F}_q}(Bh_i) = l/r.$$

Since

$$\begin{aligned} \dim_{\mathbb{F}_q}(ABh_i) &= \dim_{\mathbb{F}_q}(\mathrm{Diag}(a_1, \ldots, a_r)h_i) \\ &= \dim_{\mathbb{F}_q}(A_1 h_{1,i} + \cdots + A_r h_{r,i}), \quad i = 1, 2, \ldots, n, \end{aligned}$$

for all $i \in [n] \setminus \{j\}$ we have

$$\dim_{\mathbb{F}_q}(A_1 h_{1,i} + \cdots + A_r h_{r,i}) = l/r.$$

Since each column of B has dimension l/r over $\mathbb{F}_q$, A_u also has dimension l/r over $\mathbb{F}_q$ for every $u \in [r]$. Recall that $h_{u,i} \neq 0$ for all $u \in [r]$ and all $i \in [k]$. Thus

$$\dim_{\mathbb{F}_q}(A_u h_{u,i}) = l/r = \dim_{\mathbb{F}_q}(A_1 h_{1,i} + \cdots + A_r h_{r,i})$$

for all $u = 1, \ldots, r$ and $i \in [k] \setminus \{j\}$. Therefore,

$$A_1 h_{1,i} = A_2 h_{2,i} = \cdots = A_r h_{r,i} \text{ and all } i \in [k] \setminus \{j\}.$$

Since $h_{1,i} = 1$ for all $i = 1, 2, \ldots, k$, we have

$$A_2 h_{2,i} = A_1 \text{ for all } i \in [k] \setminus \{j\}. \tag{5.35}$$

Equivalently,

$$A_2 \alpha_i = A_2 \text{ for all } i \in \{1, 2, \ldots, k-1\} \setminus \{j\}.$$

By definition A_2 is a vector space over $\mathbb{F}_q$, so

$$A_2 \gamma = A_2 \text{ for all } \gamma \in \mathbb{F}_q(\{\alpha_i : i \in \{1, 2, \ldots, k-1\} \setminus \{j\}\}). \tag{5.36}$$

On the other hand,

$$\begin{aligned}\dim_{\mathbb{F}_q}(A_1h_{1,j}+\cdots+A_rh_{r,j}) &= \dim_{\mathbb{F}_q}(\mathrm{Diag}(a_1,\ldots,a_r)h_j) = \dim_{\mathbb{F}_q}(ABh_j)\\ &= \dim_{\mathbb{F}_q}(Bh_j) = \dim_{\mathbb{F}_q}\{c_{1,j},c_{2,j},\ldots,c_{l,j}\} = l,\end{aligned} \tag{5.37}$$

while

$$\dim_{\mathbb{F}_q}(A_uh_{u,j}) = l/r, \quad u=1,2,\ldots,r. \tag{5.38}$$

Equations (5.37) and (5.38) together imply that the vector spaces $A_1h_{1,j}, A_2h_{2,j},\ldots,A_rh_{r,j}$ are pairwise disjoint. In particular, $A_1\cap A_2h_{2,j}=\{0\}$. On account of (5.35), we therefore have $A_2h_{2,k}\cap A_2h_{2,j}=\{0\}$. This implies that $A_2\alpha_j\neq A_2$. By (5.36), we conclude that $\alpha_j\notin\mathbb{F}_q(\{\alpha_i : i\in\{1,2,\ldots,k-1\}\setminus\{j\}\})$. This completes the proof of the claim.

5.5 A family of Reed-Solomon Codes with asymptotically optimal repair bandwidth

In this section we couple the linear repair scheme of [26] (see Section 5.1.2 for a recap) with the r-ary expansion idea of [6, 61] to construct a family of RS codes with asymptotically optimal repair bandwidth.

Given any n and k, we will specify a symbol field E, which is a degree l finite field extension over some finite field F, and a set of evaluation points Λ, and view the $\mathrm{RS}_E(n,k,\Lambda)$ codes as (n,k,l) array codes over F. We will show that they have repair bandwidth bounded above by $\frac{l(n+1)}{n-k}$ over the base field F. Since the optimal repair bandwidth for an (n,k,l) MDS array code is $\frac{l(n-1)}{n-k}$, we conclude that when $n\to\infty$, the ratio between the actual and the optimal repair bandwidth approaches 1 (the corresponding quantity of the construction in [26] is about 1.5).

5.5.1 The choice of symbol field and evaluation points

Here we explain how to find a symbol field E and a set of evaluation points Λ such that the corresponding RS code has nearly optimal repair bandwidth.

Suppose that n and k are arbitrary fixed numbers. Let F be a finite field and let $h(x)\in F[x]$ be a degree l irreducible polynomial over F, where $l=r^n, r=n-k$. Let β be a root of $h(x)$ and set the symbol field to be $E=F(\beta)$, i.e., the field generated by β over F. Clearly $\{1,\beta,\beta^2,\ldots,\beta^{l-1}\}$ is a basis for E over F. Choose the set of evaluation points to be $\Lambda=\{\beta^{r^0},\beta^{r^1},\ldots,\beta^{r^{n-1}}\}$.

Theorem 5.12. *The repair bandwidth of the code $RS_E(n,k,\Lambda)$ over F is less than $l\frac{n+1}{n-k}$.*

Proof. We need to show that for every $i\in[n]$, we can find polynomials $f_{i,j}$ with $\deg(f_{i,j})<r, j=1,\ldots,l$ such that $f_{i,1}(\beta^{r^{i-1}}),\ldots,f_{i,l}(\beta^{r^{i-1}})$ form a basis for E over F and

$$\sum_{0\le t<n, t\neq i-1}\dim_F(\{f_{i,j}(\beta^{r^t})\}_{j\in[l]}) < \frac{l(n+1)}{n-k}.$$

For $a = 0, 1, \ldots, l-1$, write its r-ary expansion as $a = (a_n, a_{n-1}, \ldots, a_1)$, where a_i is the i-th digit from the right. Define the set of l polynomials $\{f_{i,j}\}_{j\in[l]} = \{\beta^a x^s : a_i = 0, s = 0, 1, \ldots, r-1\}$.

It is easy to verify that

$$\{f_{i,j}(\beta^{r^{i-1}}) : j \in [l]\} = \{1, \beta, \beta^2, \ldots, \beta^{l-1}\}$$

(as sets), so the elements $\{f_{i,j}(\beta^{r^{i-1}})\}_{j\in[l]}$ form a basis for E over F. When $t < i-1$, we have

$$\{f_{i,j}(\beta^{r^t})\}_{j\in[l]} = \{\beta^a : a_i = 0\} \bigcup \Big(\bigcup_{u=0}^{r-2} \{\beta^a : a_i = 1, a_{i-1} = \cdots = a_{t+2} = 0, a_{t+1} = u\} \Big).$$

Thus $\dim_F(\{f_{i,j}(\beta^{r^t})\}_{j\in[l]}) \leq \frac{l}{r} + (r-1)\frac{l}{r^{i-t}}$ if $t < i-1$. When $t > i-1$, we have

$$\{f_{i,j}(\beta^{r^t})\}_{j\in[l]} = \{\beta^a : a_i = 0\} \bigcup \Big(\bigcup_{u=0}^{r-2} \{\beta^{l+a} : a_n = \cdots = a_{t+2} = 0, a_{t+1} = u, a_i = 0\} \Big).$$

Thus $\dim_F(\{f_{i,j}(\beta^{r^t})\}_{j\in[l]}) \leq \frac{l}{r} + (r-1)\frac{l}{r^{n-t+1}}$ for $t > i-1$. An upper bound on the sum of the dimensions is given by:

$$\begin{aligned}
&\sum_{0\leq t<n, t\neq i-1} \dim_F(\{f_{i,j}(\beta^{r^t})\}_{j\in[l]}) \\
&\leq (n-1)\frac{l}{r} + (r-1)\sum_{t=0}^{i-2} \frac{l}{r^{i-t}} + (r-1)\sum_{t=i}^{n-1} \frac{l}{r^{n-t+1}} \\
&= l\Big(\frac{n-1}{r} + \frac{r^{i-1}-1}{r^i} + \frac{r^{n-i}-1}{r^{n-i+1}}\Big) \\
&< l\frac{n+1}{n-k}.
\end{aligned}$$

The proof is complete. □

5.6 Concluding remarks

In this chapter, we constructed optimal-repair RS code with sub-packetization $l = \exp((1+o(1))n\log n)$. We also proved an almost matching lower bound on l, showing that the super-exponential scaling is both necessary and sufficient for achieving the cut-set bound (2.1) using linear repair schemes. Finally, we constructed an $(n, k = n-r)$ RS code with sub-packetization $l = r^n$ (much smaller than the first construction) that asymptotically achieves (2.1) for the repair of any single failed node from all the other $n-1$ surviving nodes.

A possible future direction of research is to study the optimal trade-off between the repair bandwidth of scalar codes and the sub-packetization value. From the results in this chapter we know that if we allow ourselves $l = r^n$, then the bandwidth can approach (2.1). It is therefore of interest to study the best achievable repair bandwidth of RS codes with smaller sub-packetization.

Chapter 6: Fractional decoding: Error correction from partial information

In this chapter we consider error correction by MDS codes based on a part of the received codeword. Our problem is motivated by applications in distributed storage. While efficiently correcting erasures by MDS storage codes (the "repair problem") has been widely studied in recent literature, the problem of correcting errors in a similar setting seems to represent a new question in coding theory.

Suppose that k data symbols are encoded using an (n, k) MDS code, and some of the codeword coordinates are located on faulty storage nodes that introduce errors. We want to recover the original data from the corrupted codeword under the constraint that the decoder can download only an α proportion of the codeword (*fractional decoding*). For any (n, k) code we show that the number of correctable errors under this constraint is bounded above by $\lfloor (n - k/\alpha)/2 \rfloor$. We also present two families of MDS array codes which achieve this bound with equality under a simple decoding procedure. The decoder downloads an α proportion of each of the codeword's coordinates, and provides a much larger decoding radius compared to the naive approach of reading some αn coordinates of the codeword. One of the code families is formed of Reed-Solomon (RS) codes with well-chosen evaluation points, while the other is based on folded RS codes.

Finally, we show that folded RS codes also have the optimal list decoding radius under the fractional decoding constraint.

The results presented in this chapter appear in [63].

6.1 Introduction

In this chapter we extend the setting of erasure correction from partial information to the problem of *error correction*. In a distributed system, we usually face a limitation on the disk input/output operations as well as on the amount of information transmitted for the purpose of decoding (decoding bandwidth). Under no limitations on the decoding bandwidth, it is possible to recover the information from any $\lfloor (d-1)/2 \rfloor$ errors, where d is the distance of the code. Assuming that the system permits the decoder to utilize only an $\alpha < 1$ proportion of the whole codeword, we face the natural question of how many errors we can guarantee to correct in this setup. In other words, how much do we give up in terms of error-correcting capability by limiting the decoding bandwidth?

Informal description of the problem. Let $\mathcal{C}$ be a code of length n and dimension k over a field F. Let $y = c + e$ be equal to a codeword c plus an error vector $e \in F^n$. Suppose

that the decoder attempts to recover the data encoded in c based on a part of the vector y. More specifically, suppose that the input to the decoder is formed as some functions f_i of the symbols $y_i, i = 1, \ldots, n$, each defined on an individual coordinate, and suppose that the total number of field symbols available to the decoder is $\alpha n, \alpha \leq 1$. Clearly we should take $\alpha \geq k/n$ because the codeword encodes k data symbols, and even without errors to recover the data the decoder needs at least as many input symbols. If $\alpha = 1$, we return to the standard decoding problem, so our goal is to study error correction for α in the range $\frac{k}{n} \leq \alpha < 1$.

We call the number of errors correctable from an α proportion of the codeword the *α-decoding radius*, and denote it by $r_\alpha(n, k)$. Our first result, proved in Section 6.2, is an upper bound on $r_\alpha(n, k)$. Then in Sections 6.3 and 6.4 we present two code constructions that achieve this upper bound with equality. The code families that we consider belong to the class of array codes, and therefore we phrase our definitions and results in terms of such codes. At the same time, the general problem of fractional decoding as well as the upper bound on the α-decoding radius in Theorem 6.2 apply to all codes, and do not depend on the specific setting considered below.

An (n, k, l) array code $\mathcal{C}$ is formed of $l \times n$ matrices $C = (C_1, \ldots, C_n) \in (F^l)^n$, where F is a finite field. Each column C_i of the matrix is a codeword coordinate, and the parameter l that determines the dimension of the column vector C_i is called *sub-packetization*. We may also consider $\mathcal{C}$ as a code over the alphabet F^l, and then one error amounts to an incorrect column C_i. Accordingly, correcting up to t errors means correcting any combination of errors $E = (E_1, E_2, \ldots, E_n) \in (F^l)^n$ of Hamming weight $w(E) := |\{i : E_i \neq 0\}| \leq t$, where the received codeword is the matrix $C + E = (C_i + E_i, i = 1, \ldots, n)$.

Definition 6.1 (α-decoding radius)**.** Let $\mathcal{C}$ be an (n, k, l) array code over the field F. We say that $\mathcal{C}$ corrects up to t errors by downloading αnl symbols of F if there exist functions

$$f_i : F^l \to F^{\alpha_i l}, i = 1, \ldots, n \text{ and } g : F^{(\sum_{i=1}^n \alpha_i)l} \to F^{nl} \tag{6.1}$$

such that $\sum_{i=1}^n \alpha_i \leq n\alpha$ and for any codeword $C \in \mathcal{C}$ and any error $E \in (F^l)^n, w(E) \leq t$

$$g(f_1(C_1 + E_1), f_2(C_2 + E_2), \ldots, f_n(C_n + E_n)) = (C_1, C_2, \ldots, C_n). \tag{6.2}$$

For $\alpha \geq k/n$, we define the α-decoding radius of $\mathcal{C}$ as the maximum number of errors that $\mathcal{C}$ can correct by downloading αnl symbols of F, and denote it as $r_\alpha(\mathcal{C})$.

Finally, define the *α-decoding radius* $r_\alpha(n, k)$ as follows:

$$r_\alpha(n, k) = \max_{\mathcal{C} \in \mathcal{M}_{n,k}} r_\alpha(\mathcal{C}),$$

where $\mathcal{M}_{n,k}$ is the set of all (n, k) codes.

Remark 6.1. Note that the part of the codeword that we access may be larger than αn, and we only require that the decoding function g uses no more than αnl symbols of F. The motivation comes from distributed storage where what matters is the amount of information transmitted over the network. In the context of regenerating codes and

erasure correction, this quantity is usually called the repair bandwidth. If the portion of the codeword accessed for the repair purposes is the same as the minimum possible repair bandwidth, then the codes are referred to as *optimal-access* regenerating codes (see [67, 78]). In this chapter we resort to the same language for the problem of error correction.

For any code $\mathcal{C}$ we have $r_1(\mathcal{C}) \leq \lfloor (n-k)/2 \rfloor$, with equality if $\mathcal{C}$ is an MDS code. Thus, $r_1(n,k) = \lfloor (n-k)/2 \rfloor$. We also have the following obvious lower bound on $r_\alpha(n,k)$.

Lemma 6.1. *For any $k \leq n$ and $k/n \leq \alpha \leq 1$*

$$r_\alpha(n,k) \geq \lfloor (\alpha n - k)/2 \rfloor. \tag{6.3}$$

To see this, take an (n,k) MDS code and pick αn coordinates to form a punctured code $\mathcal{C}$. The punctured code $\mathcal{C}$ is an MDS code of length αn and dimension k, so (6.3) follows by definition.

In this chapter we show that

$$r_\alpha(n,k) = \lfloor (n - k/\alpha)/2 \rfloor \tag{6.4}$$

for any n, k and α and we give two families of explicit constructions of MDS codes together with decoding schemes for which the optimal value in (6.4) is achieved. The optimal α-decoding radius in (6.4) improves upon the lower bound (6.3) obtained from the naive decoding strategy by a factor of $1/\alpha$ (see Fig. 1 which shows $r_\alpha(n,k)/n$ vs α).

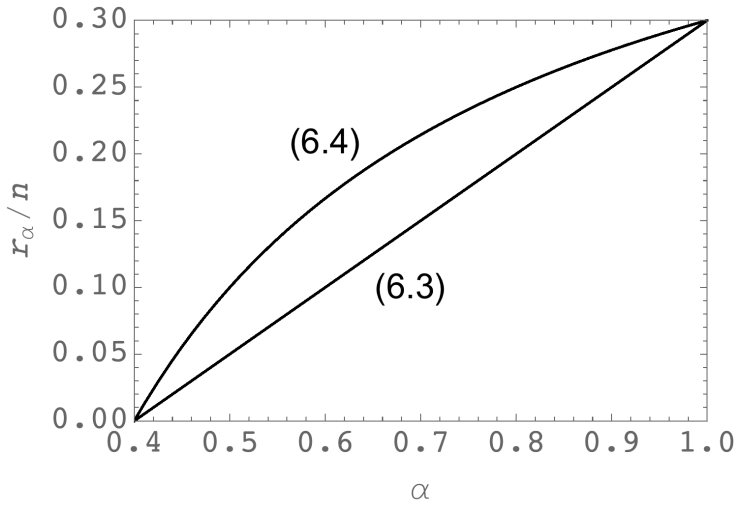

Figure 6.1: Normalized α-decoding radius: Trivial bound (6.3) and improved bound (6.4) for codes of rate $R = 0.4$.

The underlying idea behind our code constructions is as follows. Let ϵ be the proportion of erroneous coordinates in the codeword. The naive decoding procedure suggests to choose some αn coordinates and decode based on their entire contents. However, in

the worst case the number of errors in the chosen coordinates can be as large as ϵn, and we will have only $(\alpha - \epsilon)n$ error-free coordinates in the chosen subset. At the same time, by symmetrizing the decoding procedure and reading exactly an α proportion of each coordinate's contents, we can increase the proportion of error-free downloaded contents regardless of the location of the erroneous coordinates and improve the chances for successful decoding.

One code family that we construct is formed of Reed-Solomon (RS) codes with carefully chosen evaluation points, and the other one is based on Folded RS codes of Guruswami and Rudra [25].

In Sect. 6.4 we also introduce the notion of α-list decoding capacity, and show that Folded Reed-Solomon codes can achieve the α-list decoding capacity.

6.2 Upper bound on the α-decoding radius

In this section we prove the following result.

Theorem 6.2.

$$r_\alpha(n,k) \le \lfloor (n - k/\alpha)/2 \rfloor. \tag{6.5}$$

Proof. We need to show that if an (n,k,l) code $\mathcal{C}$ over the field F can correct up to t errors by downloading αnl symbols of F (see Def. 6.1), then $t \le \lfloor (n - k/\alpha)/2 \rfloor$. By assumption, for any codeword $C = (C_1, \dots, C_n) \in \mathcal{C}$ and any error vector $E = (E_1, E_2, \dots, E_n)$ of weight $\le t$ there exist $n+1$ functions $f_i : F^l \to F^{\alpha_i l}, i = 1, 2, \dots, n$ and $g : F^{(\sum_{i=1}^n \alpha_i)l} \to F^{nl}$ that satisfy (6.1)-(6.2).

We claim that $\sum_{i\in\mathcal{I}} \alpha_i \ge k$ for any set $\mathcal{I} \subseteq \{1, 2, \dots, n\}$ with cardinality $|\mathcal{I}| = n - 2t$. Assume toward a contradiction that there is a set $\mathcal{I}_0 \subseteq \{1, 2, \dots, n\}, |\mathcal{I}_0| = n - 2t$ such that $\sum_{i\in\mathcal{I}_0} \alpha_i < k$. Without loss of generality, assume that $\mathcal{I}_0 = \{1, 2, \dots, n - 2t\}$. Let us partition the set $\{1, 2, \dots, n\} \backslash \mathcal{I}_0$ into two disjoint sets $\mathcal{J}_1$ and $\mathcal{J}_2$ such that $|\mathcal{J}_1| = |\mathcal{J}_2| = t$. Since the dimension of $\mathcal{C}$ is k, there are a total of $|F|^{kl}$ codewords. At the same time, the vector $(f_i(C_i), i = 1, \dots, n - 2t)$ takes at most $\prod_{i\in\mathcal{I}_0} |F|^{\alpha_i l} = |F|^{(\sum_{i=\mathcal{I}_0} \alpha_i)l} < |F|^{kl}$ different values, so there exist two distinct codeword $\hat{C}$ and $\tilde{C}$ for which these vectors coincide:

$$(f_1(\hat{C}_1), f_2(\hat{C}_2), \dots, f_{n-2t}(\hat{C}_{n-2t})) = (f_1(\tilde{C}_1), f_2(\tilde{C}_2), \dots, f_{n-2t}(\tilde{C}_{n-2t})). \tag{6.6}$$

Define error vectors $\hat{E}$ and $\tilde{E}$ by setting

$$\hat{E}_i = \begin{cases} \tilde{C}_i - \hat{C}_i & \text{if } i \in \mathcal{J}_1 \\ 0 & \text{if } i \notin \mathcal{J}_1 \end{cases}, \quad \tilde{E}_i = \begin{cases} \hat{C}_i - \tilde{C}_i & \text{if } i \in \mathcal{J}_2 \\ 0 & \text{if } i \notin \mathcal{J}_2 \end{cases}. \tag{6.7}$$

Clearly, the weight of both $\hat{E}$ and $\tilde{E}$ is at most t. By (6.2), we have

$$g(f_1(\hat{C}_1 + \hat{E}_1), f_2(\hat{C}_2 + \hat{E}_2), \dots, f_n(\hat{C}_n + \hat{E}_n)) = \hat{C}$$
$$g(f_1(\tilde{C}_1 + \tilde{E}_1), f_2(\tilde{C}_2 + \tilde{E}_2), \dots, f_n(\tilde{C}_n + \tilde{E}_n)) = \tilde{C}$$

According to (6.6)-(6.7), $f_j(\hat{C}+\hat{E}) = f_j(\tilde{C}+\tilde{E}), j = 1, 2, \ldots, n$. As a result, $\hat{C} = \tilde{C}$, in contradiction to our assumption. We conclude that $\sum_{i\in\mathcal{I}} \alpha_i \geq k$ for any set $\mathcal{I} \subseteq \{1, 2, \ldots, n\}$ of size $|\mathcal{I}| = n - 2t$.

Let $\mathcal{I}$ be an $(n-2t)$-subset of $\{1, \ldots, n\}$ such that the quantity $\sum_{i\in\mathcal{I}} \alpha_i$ is the smallest among all $(n-2t)$-subsets. By the above argument the average proportion of information transmitted from a coordinate in the set $\mathcal{I}$ is at least $k/(n-2t)$. On the other hand, by definition, the average proportion transmitted from all the coordinates is at most α, which is at least $k/(n-2t)$ because of the property the set $\mathcal{I}$ satisfies. Hence we get $k/(n-2t) \leq \alpha$, and this concludes the proof.

□

6.3 A Reed-Solomon code construction

In this section we construct an evaluation point set Ω such that the corresponding RS code achieves the optimal α-decoding radius (6.4). Let $F = GF(q^l)$ be an l-degree extension of $B = GF(q)$ and let

$$\mathrm{tr}_{F/B}(\beta) = \beta + \beta^q + \beta^{q^2} + \cdots + \beta^{q^{l-1}}.$$

be the trace function. Let $\zeta_0, \zeta_1, \ldots, \zeta_{l-1}$ be a basis of F over B and let $\nu_0, \nu_1, \ldots, \nu_{l-1}$ be the trace-dual basis (i.e., $\mathrm{tr}_{F/B}(\nu_i\zeta_j) = \delta_{ij}$ for all i, j). Then every element $\beta \in F$ can be calculated from its l projections $\{\mathrm{tr}_{F/B}(\zeta_i\beta)\}_{i=0}^{l-1}$ on B as follows:

$$\beta = \sum_{i=0}^{l-1} \mathrm{tr}_{F/B}(\zeta_i\beta)\nu_i.$$

Let $\alpha = m/l < 1$, where m and l are positive integers. Let $q \geq n$ be the cardinality of the alphabet B and suppose that $m|k$. We show that an (n, k) RS code $\mathrm{RS}_F(n, k, \Omega) \subseteq F^n$ with all the evaluation points $\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\} \subseteq B$ has the optimal α-decoding radius.

To link our construction with the definitions made earlier, we view each codeword coordinate as a vector of dimension l over B. Thus $\mathrm{RS}_F(n, k, \Omega)$ can be viewed as an (n, k, l) MDS array code over the base field B. Our decoding scheme will be based on downloading m symbols of B from each of the codeword coordinates.

Before proceeding we need to introduce some notation. We write the encoding polynomial as

$$h(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + \cdots + a_0,$$

then the i-th coordinate of the codeword is[1]

$$c_i = h(\omega_i) = a_{k-1}\omega_i^{k-1} + a_{k-2}\omega_i^{k-2} + \cdots + a_0. \tag{6.8}$$

[1]For the time being we view the codeword coordinates as scalars in F rather than vectors over B, and use lowercase c_i instead of C_i in Def. 6.1 to denote them.

For $j = 0, 1, \ldots, l-1$, we further define

$$h_j(x) = \sum_{i=0}^{k-1} \mathrm{tr}_{F/B}(\zeta_j a_i) x^i.$$

Since the coefficients of $\{h_j(x)\}_{j=0}^{l-1}$ contain all the projections of the coefficients of $h(x)$ on the elements of the basis $\zeta_j, j = 0, \ldots, l-1$, the coefficients of $h(x)$ can be calculated from the coefficients of $\{h_j(x)\}_{j=0}^{l-1}$. In other words, to recover the codeword, we only need to recover the l polynomials $h_j(x)$.

Let $A_0, A_1, \ldots, A_{m-1} \subseteq B$ be m pairwise disjoint subsets of the field B, each of size k/m. For $j = 0, 1, \ldots, m-1$, define the annihilator polynomial of the set A_j to be

$$p_j(x) = \prod_{\omega \in A_j} (x - \omega).$$

The m symbols we download from the i-th coordinate are

$$d_i^{(j)} = \mathrm{tr}_{F/B}(\zeta_{l-m+j} c_i)(p_j(\omega_i))^{l-m} + \sum_{u=0}^{l-m-1} \mathrm{tr}_{F/B}(\zeta_u c_i)(p_j(\omega_i))^u, \quad j = 0, 1, \ldots, m-1. \tag{6.9}$$

Clearly, $d_i^{(j)} \in B$ for all $j = 0, 1, \ldots, m-1$. Substituting (6.8) into (6.9) and noting that $\omega_i \in B$, we see that $d_i^{(j)} = g_j(\omega_i), j = 0, 1, \ldots, m-1$, where

$$g_j(x) = h_{l-m+j}(x)(p_j(x))^{l-m} + \sum_{u=0}^{l-m-1} h_u(x)(p_j(x))^u.$$

Since $\deg(p_j) = k/m$, we have $\deg(g_j) < lk/m$. Thus for every $j = 0, 1, \ldots, m-1$,

$$(d_1^{(j)}, d_2^{(j)}, \ldots, d_n^{(j)}) \in \mathrm{RS}_B(n, lk/m, \Omega).$$

As a result, we can recover all the coefficients of polynomials $\{g_j(x)\}_{j=0}^{m-1}$ as long as there are no more than $\lfloor (n - lk/m)/2 \rfloor$ errors in the original codeword $(c_1, c_2, \ldots, c_n)$. Now we only need to show that given polynomials $\{g_j(x)\}_{j=0}^{m-1}$, we can recover the polynomials $\{h_j(x)\}_{j=0}^{l-1}$. To see this, we notice that for $j = 0, 1, \ldots, m-1$,

$$g_j(\omega) = h_0(\omega) \text{ for all } \omega \in A_j.$$

Consequently, we know the evaluations of $h_0(x)$ at all the points in $\cup_{j=0}^{m-1} A_j$. There are k distinct points in the set $\cup_{j=0}^{m-1} A_j$ and the degree of $h_0(x)$ is less than k, so we can recover $h_0(x)$. From $h_0(x)$ and $\{g_j(x)\}_{j=0}^{m-1}$, we can calculate the polynomials

$$\begin{aligned} g_j^{(1)}(x) &= \frac{g_j(x) - h_0(x)}{p_j(x)} \\ &= h_{l-m+j}(x)(p_j(x))^{l-m-1} + \sum_{u=1}^{l-m-1} h_u(x)(p_j(x))^{u-1}, \quad j = 0, 1, \ldots, m-1. \end{aligned}$$

Since $g_j^{(1)}(\omega) = h_1(\omega)$ for all $\omega \in A_j$, we know the evaluations of $h_1(x)$ at all the points in $\cup_{j=0}^{m-1} A_j$. So we can also recover $h_1(x)$. From $h_0(x), h_1(x)$ and $\{g_j(x)\}_{j=0}^{m-1}$, we can calculate the polynomials

$$\begin{aligned} g_j^{(2)}(x) &= \frac{g_j^{(1)}(x) - h_1(x)}{p_j(x)} \\ &= h_{l-m+j}(x)(p_j(x))^{l-m-2} + \sum_{u=2}^{l-m-1} h_u(x)(p_j(x))^{u-2}, \quad j = 0, 1, \ldots, m-1. \end{aligned}$$

Since $g_j^{(2)}(\omega) = h_2(\omega)$ for all $\omega \in A_j$, we know the evaluations of $h_2(x)$ at all the points in $\cup_{j=0}^{m-1} A_j$. So we can also recover $h_2(x)$. It is clear that we can repeat this procedure until we recover $\{h_j(x)\}_{j=0}^{l-m-1}$. Then the polynomials $\{h_{l-m+j}(x)\}_{j=0}^{m-1}$ can be easily recovered from

$$h_{l-m+j}(x) = \frac{g_j(x) - \sum_{u=0}^{l-m-1} h_u(x)(p_j(x))^u}{(p_j(x))^{l-m}}.$$

This shows that we can recover the polynomials $\{h_j(x)\}_{j=0}^{l-1}$ from the polynomials $\{g_j(x)\}_{j=0}^{m-1}$, and consequently recover the original codeword.

6.4 Fractional decoding of Folded RS codes

Guruswami and Rudra [25] introduced Folded Reed-Solomon (FRS) code to show that it achieves the list decoding capacity. In this section we show that FRS codes also have optimal α-decoding radius among all the codes of the same length and dimension. We begin with recalling the definition of FRS codes.

Definition 6.2. Let F be a finite field of cardinality $|F| > nl$. Let γ be a primitive element of F. An FRS code $\text{FRS}(n, k, l) \subseteq F^{nl}$ is an MDS array code with codewords given by

$$\begin{aligned} &\{(C_1, C_2, \ldots, C_n) : \\ &\quad C_i = \left(h(\gamma^{(i-1)l}), h(\gamma^{(i-1)l+1}), \ldots, h(\gamma^{il-1})\right) \in F^l, h \in F[x], \deg h \leq kl - 1\}. \end{aligned}$$

We assume throughout that αl is an integer.

6.4.1 Unique decoding

Proposition 6.3.

$$r_\alpha(\text{FRS}(n, k, l)) = \lfloor (n - k/\alpha)/2 \rfloor.$$

Proof. We shall define $n + 1$ functions $f_i, i = 1, 2, \ldots, n$ and g that satisfy (6.1)-(6.2). Define $f : F^l \to F^{\alpha l}$ as follows: For any $(d_1, d_2, \ldots, d_l) \in F^l$,

$$f((d_1, d_2, \ldots, d_l)) = (d_1, d_2, \ldots, d_{\alpha l}). \tag{6.10}$$

Let $f_i = f$ for $1 \leq i \leq n$. Define a new code

$$\mathcal{C}^{\alpha} = \{(C_1^{\alpha}, C_2^{\alpha}, \ldots, C_n^{\alpha}) = (f(C_1), f(C_2), \ldots, f(C_n)) : (C_1, C_2, \ldots, C_n) \in \text{FRS}(n,k,l)\}. \tag{6.11}$$

It is easy to see that C^{α} defined above has the following equivalent description:

$$C^{\alpha} = \{(C_1^{\alpha}, C_2^{\alpha}, \ldots, C_n^{\alpha}) : C_i^{\alpha} = \left(h(\gamma^{(i-1)l}), h(\gamma^{(i-1)l+1}), \ldots, h(\gamma^{(i-1)l+\alpha l-1})\right) \in F^l, \\ 1 \leq i \leq n, h \in F[x], \deg h \leq kl-1\}. \tag{6.12}$$

Since any k/α coordinates of $\mathcal{C}^{\alpha}$ contain $(k/\alpha)(\alpha l) = kl$ evaluations of the encoding polynomial h of degree less than kl, we can recover h and thus the whole codeword from any k/α coordinates of $\mathcal{C}^{\alpha}$. We thus conclude that $\mathcal{C}^{\alpha}$ is an $(n, k/\alpha, \alpha l)$ MDS array code, and so it can correct up to $\lfloor (n-k/\alpha)/2 \rfloor$ errors.

□

Remark 6.2. Suppose that there are several values $\alpha_1, \alpha_2, \ldots, \alpha_m$ such that $\alpha_i l$ are integer for all $i = 1, \ldots, m$. Then the code $\text{FRS}(n,k,l)$ achieves the optimal α_i-decoding radius for all $i = 1, \ldots, m$ simultaneously.

We can use the decoding method described above to give more general code constructions achieving the optimal α-decoding radius. In fact, we can take any (nl, kl) scalar MDS code $\mathcal{C}^s$ over finite field F and bundle together every l coordinates of $\mathcal{C}^s$ into a vector in F^l. It is clear that we obtain an (n,k,l) MDS array code $\mathcal{C}^{(\text{arr})}$ in this way. Moreover, by reading αl symbols of F from each of the coordinates of $\mathcal{C}^{(\text{arr})}$ we obtain an $(n, k/\alpha, \alpha l)$ MDS array code $\mathcal{C}^{\alpha}$ which can correct up to $\lfloor (n-k/\alpha)/2 \rfloor$ errors. Thus the code $\mathcal{C}^{(\text{arr})}$ can correct up to $\lfloor (n-k/\alpha)/2 \rfloor$ errors by downloading $\alpha n l$ symbols of F.

In conclusion note that the FRS codes accomplish fractional decoding with the optimal access property discussed briefly in Remark 6.1.

6.4.2 Fractional list decoding and α-list decoding capacity

So far we have focused on the unique decoding problem under the constraint of fractional decoding. In this section we consider list decoding from partial information.

We say that a code $\mathcal{C}$ of length n corrects ρn errors under list-of-L decoding (has normalized list decoding radius ρ) if every sphere of radius ρn in the space F^n contains at most L codewords of $\mathcal{C}$. In other words, there exists a decoder of $\mathcal{C}$ that outputs a list of $\leq L$ codewords including the transmitted one as long as the channel introduces no more that ρn errors. To maintain low complexity of decoding, we require that L be a polynomial function of n.

Let $R = k/n$ denote the rate of the code. It is clear that for any sequence of codes with fixed rate R and growing length n, the list decoding radius does not exceed $1-R$. As shown in [25], FRS codes of growing length have list decoding radius that for $n \to \infty$ approaches the optimal value of $1-R$ (as [25] puts it, FRS codes achieve the list decoding capacity).

In this section we show that the conclusion about the optimal list decoding radius of FRS codes extends to the case of fractional decoding. We begin with defining the α-list decoding radius of the code.

Definition 6.3 (α-list decoding capacity)**.** Let $\mathcal{C}$ be an (n, k, l) array code, where each codeword $C = (C_1, \ldots, C_n)$ is an $l \times n$ matrix with $C_i \in F^l, i = 1, \ldots, n$. We say that $\mathcal{C}$ corrects up to t errors under list-of-L decoding by downloading αnl symbols of F if there exist functions

$$\begin{aligned} f_i &: F^l \to F^{\alpha_i l}, i = 1, 2, \ldots, n, \text{ and} \\ g_i &: F^{(\sum_{i=1}^n \alpha_i)l} \to F^{nl}, i = 1, 2, \ldots, L \end{aligned} \tag{6.13}$$

such that $\sum_{i=1}^n \alpha_i \leq n\alpha$, and for any codeword $C \in \mathcal{C}$ and any error $E \in F^{ln}, w(E) \leq t$ the decoding list

$$\{g_i(f_1(C_1 + E_1), f_2(C_2 + E_2), \ldots, f_n(C_n + E_n))\}_{i=1}^L \tag{6.14}$$

contains the codeword C.

For $\alpha \geq k/n$, we define the (α, L)-*list decoding radius* $r_{\alpha,L}(\mathcal{C})$ of $\mathcal{C}$ as the maximum number of errors that can be corrected under list-of-L decoding by downloading αnl symbols of F. Let $r_{\alpha,L}(n, k) = \max_{\mathcal{C}} r_{\alpha,L}(\mathcal{C})$, where the maximum is taken over all codes of length n and dimension k.

For $\alpha \geq R$, we further define the α-*list decoding capacity* of codes of rate R as

$$\rho_\alpha(R) = \sup_{m \in \mathbb{N}} \limsup_{n \to \infty} \frac{r_{\alpha,n^m}(n, Rn)}{n}.$$

Using an argument that closely follows the proof of Theorem 6.2 , we can show that $\rho_\alpha(R) \leq 1 - R/\alpha$. On the other hand, there exists a family of FRS codes of increasing length n and sub-packetization l that are (α, L)-list decodable from a fraction arbitrarily close to $1 - R/\alpha$ of errors with list size L polynomial in n. This leads to the following statement.

Theorem 6.4. *We have*

$$\rho_\alpha(R) = 1 - R/\alpha,$$

and this bound is achievable by a family of FRS codes.

To show that the FRS codes achieve the α-list decoding capacity, we need to define the functions $f_i, i = 1, 2, \ldots, n$ and $g_i, i = 1, 2, \ldots, L$ that satisfy (6.13)-(6.14). We again use the functions $f_1 = f_2 = \cdots = f_n = f$ defined in (6.10), i.e., we still download an αl symbols of F from each of the codeword coordinates. Then we obtain the code $\mathcal{C}^\alpha$ defined in (6.12) whose rate is R/α. When the code length n and sub-packetization l of the FRS code become large enough, we can use the list decoding algorithm introduced in [25] to decode $\mathcal{C}^\alpha$ up to a fraction arbitrarily close to $1 - R/\alpha$ of errors with list size L polynomial in n.

Note that for sufficiently large n and l, the FRS codes achieve the α-list decoding capacity uniformly for all values of α.

Remark 6.3. The code $\mathcal{C}^{\alpha}$ differs from FRS codes in the sense that the evaluation points in two consecutive coordinates are not consecutive powers of the primitive element. However, the list decoding algorithm introduced in [25] only requires that within each codeword coordinate, the evaluation points are consecutive powers of the primitive element. The code $\mathcal{C}^{\alpha}$ satisfies this constraint, so the list decoding algorithm in [25] still applies to it.

Remark 6.4. Although FRS codes achieve the α-list decoding capacity, they require larger node (codeword coordinate) size than the RS construction given in Sect. 6.3. The base field in RS construction (the field B in Sect. 6.3) has to be only of size $q \geq n$, while the base field in this section has to be of size at least nl. As a result, the node size in the RS construction is $l \log n$ bits, while the codeword coordinates of the FRS codes contain $l \log(nl)$ bits.

6.5 Concluding remarks

In this chapter we studied the problem of decoding from errors under the constraint of downloading only a part of the received codeword. We gave two families of optimal code constructions and their error correction (decoding) procedures. The idea behind the constructions and recovery schemes is rather similar to regenerating codes: in regenerating codes, we can repair the same number of erasures by downloading a smaller proportion of the codeword if we connect to more helper nodes; here we can correct the same number of errors by downloading a smaller proportion of the codeword if we utilize all codeword coordinates. Equivalently, for the same amount of downloaded information we can decode from a larger number of errors.

A natural further question to ask is whether it is possible to construct an MDS code that corrects both erasures and errors (nearly) optimally in terms of the bandwidth, where the optimality that corresponds to erasures is measured by the cut-set bound given in [12], and the optimality of correcting errors is measured by the α-decoding radius defined in this chapter.

Chapter 7: Some open problems

In this thesis we presented various constructions of MSR codes with some additional favorable properties, such as small field size, small sub-packetization and optimal access property. We also constructed explicit scalar linear MSR codes, which resolve the open problem about the existence of such codes. Finally, we extended the concept of repair bandwidth to the context of error correction and introduced a new problem in coding theory. Below we list several open problems that constitute potential directions for future research.

1. CONSTRUCT EXPLICIT MSR CODES WITH SUB-PACKETIZATION $l = r^{n/(r+1)}$.
   For simplicity, let us consider the case $(r+1)|n$. As shown in [68], there exist MSR codes with sub-packetization $l = r^{n/(r+1)}$, and this is the best known achievable sub-packetization value. For the case $r = 2$ and 3, explicit constructions with this sub-packetization are given in [68] and [44], respectively. For general values of r, it remains an open problem to construct explicit MSR codes with small field size (e.g., linear in n) achieving this sub-packetization value.

2. CLOSE THE GAP BETWEEN ACHIEVABLE SUB-PACKETIZATION VALUE AND THE LOWER BOUND.
   The best known lower bound on the sub-packetization value of MSR codes is due to [22]. It has the following form:

   $$2\log_2 l(\log_{r/(r-1)} l + 1) \geq k.$$

   In contrast, constructions of codes with the smallest currently known sub-packetization achieve $\log_r l = n/(r+1)$ [68]. The gap between the bound and the known constructions calls for either improving the bound or finding better codes, and this constitutes an intriguing open problem concerning regenerating codes.

3. SUB-PACKETIZATION DEPENDENT BOUND ON REPAIR BANDWIDTH.
   For simplicity, let us assume that $d = n - 1$. Namely, we repair the failed node by connecting to all the surviving nodes. The cut-set bound is derived without any constraints on how large the sub-packetization l can be. According to [22], in order to achieve the cut-set bound, l needs to be larger than $2^{\sqrt{k/(2r-1)}}$. This might not always be possible in practice due to the complexity consideration. Therefore it is of interest to derive a tight (achievable) lower bound on the repair bandwidth under an additional constraint that $l < L$. The cut-set bound corresponds to the case $L = \infty$. It is interesting to see how the lower bound changes as L decreases. We

note that [5] states some relevant results in this direction, but this question is still far from being fully understood.

Bibliography

[1] I. Ahmad and C.-C. Wang. When can helper node selection improve regenerating codes I-II. eprints arXiv:1604.08230 and 1604:08231.

[2] R. Bitar and S. El Rouayheb. Staircase codes for secret sharing with optimal communication and read overheads. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 1396–1400. IEEE, 2016.

[3] M. Blaum, J. Brady, J. Bruck, and J. Menon. EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures. *IEEE Transactions on computers*, 44(2):192–202, 1995.

[4] M. Blaum, P. G. Farell, and H. C. A. van Tilborg. Array codes. In *Handbook of Coding Theory*, volume II, chapter 22, pages 1855–1909. Elsevier Science, 1998.

[5] V. Cadambe and A. Mazumdar. Alphabet-size dependent bounds for exact repair in distributed storage. In *Proc IEEE Information Theory Workshop - Fall (ITW 2015)*, pages 1–3, 2015.

[6] V. R. Cadambe, C. Huang, and J. Li. Permutation code: Optimal exact-repair of a single failed node in MDS code based distributed storage systems. In *2011 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 1225–1229. IEEE, 2011.

[7] V. R. Cadambe, C. Huang, J. Li, and S. Mehrotra. Polynomial length MDS codes with optimal repair in distributed storage. In *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pages 1850–1854. IEEE, 2011.

[8] V. R. Cadambe, S. A. Jafar, H. Maleki, K. Ramchandran, and Changho S. Asymptotic interference alignment for optimal repair of MDS codes in distributed storage. *IEEE Transactions on Information Theory*, 59(5):2974–2987, 2013.

[9] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar. Row-diagonal parity for double disk failure correction. In *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*, pages 1–14, 2004.

[10] H. Dau, I. Duursma, H. M. Kiah, and O. Milenkovic. Repairing Reed-Solomon codes with multiple erasures, 2016. arXiv:1612.01361.

[11] H. Dau and O. Milenkovic. Optimal repair schemes for some families of full-length Reed-Solomon codes. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 346–350. IEEE, 2017.

[12] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran. Network coding for distributed storage systems. *IEEE Transactions on Information Theory*, 56(9):4539–4551, 2010.

[13] D. S. Dummit and R. M. Foote. *Abstract Algebra*. Prentice Hall, Englewood Cliffs, 1991.

[14] N. B. Ellison, C. Steinfield, and C. Lampe. The benefits of Facebook friends: Social capital and college students use of online social network sites. *Journal of Computer-Mediated Communication*, 12(4):1143–1168, 2007.

[15] M. Elyasi and S. Mohajer. Determinant coding: A novel framework for exact-repair regenerating codes. *IEEE Transactions on Information Theory*, 62(12):6683–6697, 2016.

[16] M. Elyasi, S. Mohajer, and R. Tandon. Linear exact repair rate region of (k+ 1, k, k) distributed storage systems: A new approach. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2061–2065. IEEE, 2015.

[17] R. Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.

[18] S. Ghemawat, H. Gobioff, and S. Leung. The Google file system. *ACM SIGOPS Ooperating Systems Review*, 37(5):29–43, 2003.

[19] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin. On the locality of codeword symbols. *IEEE Transactions on Information Theory*, 58(11):6925–6934, 2012.

[20] S. Goparaju, S. El Rouayheb, and R. Calderbank. New codes and inner bounds for exact repair in distributed storage systems. In *2014 IEEE International Symposium on Information Theory (ISIT)*, pages 1036–1040. IEEE, 2014.

[21] S. Goparaju, A. Fazeli, and A. Vardy. Minimum storage regenerating codes for all parameters. *IEEE Transactions on Information Theory*, 2017.

[22] S. Goparaju, I. Tamo, and R. Calderbank. An improved sub-packetization bound for minimum storage regenerating codes. *IEEE Transactions on Information Theory*, 60(5):2770–2779, 2014.

[23] T. C. Gulcu, M. Ye, and A. Barg. Construction of polar codes for arbitrary discrete memoryless channels. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 51–55. IEEE, 2016.

[24] V. Guruswami and A. S. Rawat. MDS code constructions with small sub-packetization and near-optimal repair bandwidth. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2109–2122. Society for Industrial and Applied Mathematics, 2017.

[25] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

[26] V. Guruswami and M. Wootters. Repairing Reed-Solomon Codes. *IEEE Transactions on Information Theory*, 2017.

[27] H. D. L. Hollmann. On the minimum storage overhead of distributed storage codes with a given repair locality. In *Proc. 2014 IEEE Internat. Sympos. Inform. Theory, Honolulu, HI*, pages 1041–1045, 2014.

[28] K. Huang, U. Parampalli, and M. Xian. On secrecy capacity of minimum storage regenerating codes. *IEEE Transactions on Information Theory*, 63(3):1510–1524, 2017.

[29] W. Huang and J. Bruck. Secret sharing with optimal decoding and repair bandwidth. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1813–1817. IEEE, 2017.

[30] W. Huang, M. Langberg, J. Kliewer, and J. Bruck. Communication efficient secret sharing. *IEEE Transactions on Information Theory*, 62(12):7195–7206, 2016.

[31] H. Iwaniec and E. Kowalski. *Analytic number theory*, volume 53. American Mathematical Society Providence, RI, 2004.

[32] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar. Codes with local regeneration and erasure correction. *IEEE Transactions on Information Theory*, 60(8):4637–4660, 2014.

[33] J. Li, X. Tang, and C. Tian. A generic transformation for optimal repair bandwidth and rebuilding access in MDS codes. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1623–1627. IEEE, 2017.

[34] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and Their Applications*. Cambridge University Press, 1994.

[35] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. Elsevier, 1977.

[36] D. S. Papailiopoulos and A. G. Dimakis. Locally repairable codes. *IEEE Transactions on Information Theory*, 60(10):5843–5855, 2014.

[37] D. S. Papailiopoulos, A. G. Dimakis, and V. R. Cadambe. Repair optimal erasure codes through hadamard designs. *IEEE Transactions on Information Theory*, 59(5):3021–3037, 2013.

[38] S. Pawar, S. El Rouayheb, and K. Ramchandran. Securing dynamic distributed storage systems against eavesdropping and adversarial attacks. *IEEE Transactions on Information Theory*, 57(10):6734–6753, 2011.

[39] R. Pedarsani, S. H. Hassani, I. Tal, and E. Telatar. On the construction of polar codes. In *Proc. 2011 IEEE Internat. Sympos. Inform. Theory*, pages 11–15, St. Petersburg, Russia, 2011.

[40] N. Prakash and M. N. Krishnan. The storage-repair-bandwidth trade-off of exact repair linear regenerating codes for the case $d = k = n - 1$. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 859–863. IEEE, 2015.

[41] K. V. Rashmi, N. B. Shah, D. Gu, H. Kuang, D. Borthakur, and K. Ramchandran. A Hitchhiker's guide to fast and efficient data reconstruction in erasure-coded data centers. In *ACM SIGCOMM Computer Communication Review*, volume 44, pages 331–342. ACM, 2014.

[42] K. V. Rashmi, N. B. Shah, and P. V. Kumar. Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction. *IEEE Transactions on Information Theory*, 57(8):5227–5239, 2011.

[43] K. V. Rashmi, N. B. Shah, K. Ramchandran, and P. V. Kumar. Regenerating codes for errors and erasures in distributed storage. In *2012 IEEE International Symposium on Information Theory (ISIT)*, pages 1202–1206, 2012.

[44] N. Raviv, N. Silberstein, and T. Etzion. Constructions of high-rate minimum storage regenerating codes over small fields. *IEEE Transactions on Information Theory*, 63(4):2015–2038, 2017.

[45] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath. Optimal locally repairable and secure codes for distributed storage systems. *IEEE Transactions on Information Theory*, 60(1):212–236, 2014.

[46] A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath. Centralized repair of multiple node failures with applications to communication efficient secret sharing, 2016. arXiv:1603.04822.

[47] A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath. Progress on high-rate MSR codes: Enabling arbitrary number of helper nodes. In *2016 Information Theory and Applications Workshop (ITA)*, pages 1–6. IEEE, 2016.

[48] A. S. Rawat, I. Tamo, V. Guruswami, and K. Efremenko. ϵ-MSR codes with small sub-packetization. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2043–2047. IEEE, 2017.

[49] B. Sasidharan, G. K. Agarwal, and P. V. Kumar. A high-rate MSR code with polynomial sub-packetization level. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2051–2055. IEEE, 2015.

[50] B. Sasidharan, K. Senthoor, and P. V. Kumar. An improved outer bound on the storage-repair-bandwidth tradeoff of exact-repair regenerating codes. In *2014 IEEE International Symposium on Information Theory (ISIT)*, pages 2430–2434. IEEE, 2014.

[51] B. Sasidharan, M. Vajha, and P. V. Kumar. An explicit, coupled-layer construction of a high-rate MSR code with low sub-packetization level, small field size and all-node repair. 2016. arXiv:1607.07335.

[52] B. Sasidharan, M. Vajha, and P. V. Kumar. An explicit, coupled-layer construction of a high-rate MSR code with low sub-packetization level, small field size and $d < (n-1)$. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2048–2052. IEEE, 2017.

[53] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran. Distributed storage codes with repair-by-transfer and nonachievability of interior points on the storage-bandwidth tradeoff. *IEEE Transactions on Information Theory*, 58(3):1837–1852, 2012.

[54] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran. Interference alignment in regenerating codes for distributed storage: Necessity and code constructions. *IEEE Transactions on Information Theory*, 58(4):2134–2158, 2012.

[55] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[56] K. Shanmugam, D. S. Papailiopoulos, A. G. Dimakis, and G. Caire. A repair framework for scalar MDS codes. *IEEE Journal on Selected Areas in Communications*, 32(5):998–1007, 2014.

[57] N. Silberstein, A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath. Optimal locally repairable codes via rank-metric codes. In *Proceedings 2013 IEEE International Symposium on Information Theory (ISIT)*, pages 1819–1823. IEEE, 2013.

[58] C. Suh and K. Ramchandran. Exact-repair MDS code construction using interference alignment. *IEEE Transactions on Information Theory*, 57(3):1425–1442, 2011.

[59] I. Tamo and A. Barg. A family of optimal locally recoverable codes. *IEEE Transactions on Information Theory*, 60(8):4661–4676, 2014.

[60] I. Tamo, D. S. Papailiopoulos, and A. G. Dimakis. Optimal locally repairable codes and connections to matroid theory. *IEEE Transactions on Information Theory*, 62(12):6661–6671, 2016.

[61] I. Tamo, Z. Wang, and J. Bruck. Zigzag codes: MDS array codes with optimal rebuilding. *IEEE Transactions on Information Theory*, 59(3):1597–1616, 2013.

[62] I. Tamo, Z. Wang, and J. Bruck. Access versus bandwidth in codes for storage. *IEEE Transactions on Information Theory*, 60(4):2028–2037, 2014.

[63] I. Tamo, M. Ye, and A. Barg. Fractional decoding: Error correction from partial information. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 998–1002. IEEE, 2017.

[64] I. Tamo, M. Ye, and A. Barg. Optimal repair of Reed-Solomon codes: Achieving the cut-set bound. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, 2017, (arXiv:1706.00112).

[65] C. Tian. Characterizing the rate region of the (4, 3, 3) exact-repair regenerating codes. *IEEE Journal on Selected Areas in Communications*, 32(5):967–975, 2014.

[66] Z. Wang, A. G. Dimakis, and J. Bruck. Rebuilding for array codes in distributed storage systems. In *2010 IEEE GLOBECOM Workshops (GC Wkshps)*, pages 1905–1909. IEEE, 2010.

[67] Z. Wang, I. Tamo, and J. Bruck. On codes for optimal rebuilding access. In *Proceedings of the 49th Annual Allerton Conference on Communication, Control and Computing*, pages 1374–1381, 2011.

[68] Z. Wang, I. Tamo, and J. Bruck. Explicit minimum storage regenerating codes. *IEEE Transactions on Information Theory*, 62(8):4466–4480, 2016.

[69] Z. Wang, I. Tamo, and J. Bruck. Optimal rebuilding of multiple erasures in MDS codes. *IEEE Transactions on Information Theory*, 63(2):1084–1101, 2017.

[70] Y. Wu and A.G. Dimakis. Reducing repair traffic for erasure coding-based storage via interference alignment. In *2009 IEEE International Symposium on Information Theory (ISIT)*, pages 2276–2280, 2009.

[71] L. Xiang, Y. Xu, J. Lui, and Q. Chang. Optimal recovery of single disk failure in RDP code storage systems. *ACM SIGMETRICS Performance Evaluation Review*, 38(1):119–130, 2010.

[72] M. Ye and A. Barg. Universal source polarization and an application to a multi-user problem. In *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 805–812. IEEE, 2014.

[73] M. Ye and A. Barg. Polar codes for distributed hierarchical source coding. *Advances in Mathematics of Communications*, 9(1):87–103, 2015.

[74] M. Ye and A. Barg. Polar codes using dynamic kernels. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 231–235. IEEE, 2015.

[75] M. Ye and A. Barg. Explicit constructions of MDS array codes and RS codes with optimal repair bandwidth. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 1202–1206. IEEE, 2016.

[76] M. Ye and A. Barg. Asymptotically optimal private estimation under mean square loss. 2017. arXiv:1708.00059.

[77] M. Ye and A. Barg. Explicit constructions of high-rate MDS array codes with optimal repair bandwidth. *IEEE Transactions on Information Theory*, 63(4):2001–2014, 2017.

[78] M. Ye and A. Barg. Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization. *IEEE Transactions on Information Theory*, 2017.

[79] M. Ye and A. Barg. Optimal schemes for discrete distribution estimation under local differential privacy. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 759–763. IEEE, 2017.

[80] M. Ye and A. Barg. Optimal schemes for discrete distribution estimation under locally differential privacy. 2017. arXiv:1702.00610.