ABSTRACT

Title of Dissertation: CODES WITH EFFICIENT ERASURE CORRECTION

Zitan Chen
Doctor of Philosophy, 2020

Dissertation Directed by: Professor Alexander Barg
Department of Electrical and Compute Engineering
Institute for Systems Research

Distributed storage systems are becoming increasingly ubiquitous in the emerging era of Internet of Things. Major internet technology companies employ large-scale distributed storage systems to accommodate the massive amounts of data generated and requested by global users. The need of reliable and efficient storage of immense amounts of data calls for new applications and development of classical error-correcting codes.

This dissertation is devoted to a study of codes with efficient erasure correction for distributed storage systems. The efficiency of erasure correction is often assessed by two performance metrics, *bandwidth* and *locality*. In this dissertation we address several problems for each of these two metrics. We construct families of codes with optimal communication complexity for erasure correction ("repair bandwidth") for a heterogeneous storage model, and derive several results for the problem of optimal repair of Reed-Solomon codes. We also construct families of cyclic and convolutional codes with locality, extending the range of parameters for which such families were previously known.

CODES WITH EFFICIENT ERASURE CORRECTION

by

Zitan Chen

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
2020

Advisory Committee:
Professor Alexander Barg, Chair/Advisor
Professor Behtash Babadi
Professor Prakash Narayan
Professor Sennur Ulukus
Professor William Gasarch

Acknowledgments

First and foremost, I would like to thank my PhD advisor, Professor Alexander Barg for his guidance and support for the past five years. He always made himself available to discuss any questions or ideas I had and provide helpful suggestions to me based on his broad knowledge and rich research experience. It is a great fortune for me to have him as my advisor. I am also indebted to Sidharth Jaggi and Michael Langberg, my undergraduate research advisors, who led to me into the fascinating realm of information theory and coding theory. Without their introduction, I would not have chosen to pursue a doctoral degree in these fields, let alone complete this dissertation.

I would like to acknowledge financial support that made it possible for me to focus on my studies and research and to have the opportunity to learn from so many incredible individuals at the University of Maryland, College Park. I feel fortunate to have met so many excellent people on this beautiful campus. Thanks for all the memories we created together. I own my deepest thanks to all the friends, here at College Park or far in other places, who spoke with me, listened to me, and accompanied me during most difficult times in this journey.

My most important acknowledgment is to my parents for their constant love, understanding and support, and for all they have done for me. There are no words in the world that can express my thanks to them.

Lastly, I am grateful to my dissertation committee members, who kindly agreed to serve on the panel, for their time and consideration.

Table of Contents

List of Figures

List of Abbreviations

GRS	Generalized Reed-Solomon (codes)	Section 2.2.1
H-LRC	Hierarchical Locally Recoverable (codes)	Section 1.2.5
LRC	Locally Recoverable (codes)	Section 1.2.4
MDS	Maximum Distance Separable (codes)	Section 1.2.1
MSR	Minimum Storage Regenerating (codes)	Section 1.2.1
OA	Optimal Access	Section 1.2.2
RS	Reed-Solomon (codes)	Section 1.2.1

Chapter 1: Introduction

1.1 Motivation

Large-scale distributed storage systems are arguably the backbone of numerous cutting-edge technologies in our contemporary society. These systems operate on an increasingly large scale, and their reliability becomes the central consideration of the system design. For example, parts of the data stored in the systems may be inaccessible due to events such as unreliable network connections, power outages, and disk-failures. Moreover, such events are the norm rather than the exception in the course of daily operations of the system. To improve system reliability and to combat data loss, several classes of erasure codes have been brought into play in distributed storage systems in practice.

Existing coding schemes in distributed storage systems are usually capable of correcting multiple erasures. However, the most common erasure pattern in distributed storage systems in reality is the case of a single erasure. Since most of the existing coding schemes and their erasure correction procedures do not distinguish between correcting single and multiple erasures, they fall short of recovering a single erasure *efficiently*. Thus, the problem of efficient erasure correction in various classes of algebraic codes, also known as the *repair problem*, has recently attracted renewed

attention. In this dissertation, we address a series of questions related to the general problem of codes with efficient erasure correction.

1.2 Preliminaries and prior work

In this section, we introduce some basic definitions and terminology related to coding for distributed storage, and review prior work most relevant to the results in this dissertation.

A distributed storage system is formed of a collection of n nodes that are used to store the data. Loss of data in a node is called a node failure and the recovery of the data in a failed node is termed node repair. Using the coding-theoretic language, a distributed storage system is a code of length n, and therefore, a node corresponds to a coordinate of the code. In the same vein, a failed node refers to an erasure of the coordinates and erasure correction is another term to describe node repair.

The performance of efficient node repair can be measured by different metrics, among which we are most interested in *bandwidth* and *locality*.

1.2.1 Efficiency in terms of bandwidth

To a large extent, efficient repair of failed nodes critically depends on the volume of communication exchanged between the nodes. The constraint on the amount of communication, termed "repair bandwidth," adds new features to the erasure correction problem, and has motivated a large amount of research in coding theory in the last decade.

The repair problem under restriction of low repair bandwidth was initially introduced in the well-known paper [20] which casts the capacity problem of distributed storage as a network coding problem where the necessary conditions for the repair of failed nodes were derived by considering the information flow in the network that occurred in the course of repair. These conditions imply a bound on the minimum number of symbols required for repair of a single failed node, which is known as the cut-set bound on the repair bandwidth. Paper [20] further considered a variety of data coding schemes, termed regenerating codes, that optimize either storage or repair bandwidth, as well as the tradeoff between these two quantities.

Consider an (n, k, l) array code $\mathcal{C}$ over a finite field F,[1] i.e., a collection of codewords $c = (c_1, \ldots, c_n)$, where $c_i = (c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})^T \in F^l, i = 1, \ldots, n$. A node $c_i, i \in [n]$ can be repaired from a subset of $d \geqslant k$ helper nodes $\{c_j : j \in \mathcal{R}\}, \mathcal{R} \subseteq [n]\backslash\{i\}$, by downloading $\beta_i(\mathcal{R})$ symbols of F if there are numbers $\beta_{ij}, j \in \mathcal{R}$, functions $f_{ij} : F^l \to F^{\beta_{ij}}, j \in \mathcal{R}$, and a function $g_i : F^{\sum_{j\in\mathcal{R}} \beta_{ij}} \to F^l$ such that

$$c_i = g_i(\{f_{ij}(c_j), j \in \mathcal{R}\}) \text{ for all } c = (c_1, \ldots, c_n) \in \mathcal{C}$$

and

$$\sum_{j\in\mathcal{R}} \beta_{ij} = \beta_i(\mathcal{R}).$$

Codes that we consider form linear spaces over F. If $\mathcal{C}$ is not linear over F^l, it is also called a *vector* code, while if it is, it is called *scalar* to stress the linearity property. A code $\mathcal{C}$ is called *maximum distance separable* or MDS if any k coordinates

[1] We also denote a finite field by $\mathbb{F}_q$ to indicate that its order is q.

$\{c_{j_i}, i = 1, \ldots, k\}$ of the codeword suffice to recover its remaining $n - k$ coordinates.

It is well known [20] that for any MDS code $\mathcal{C}$ (scalar or vector), any $i \in [n]$, and any $\mathcal{R} \subseteq [n] \backslash \{i\}$ of cardinality $|\mathcal{R}| \geqslant k$, we have

$$\beta_i(\mathcal{R}) \geqslant \frac{|\mathcal{R}|l}{|\mathcal{R}| - k + 1}. \tag{1.1}$$

For an MDS code $\mathcal{C}$, we define the minimum bandwidth of repair of a node from a d-subset $\mathcal{R}$ of helper nodes as $\beta(d) = \max_{i \in [n]} \min_{\mathcal{R} \subset [n] \backslash \{i\}, |\mathcal{R}| = d} \beta_i(\mathcal{R})$. It follows immediately from (1.1) that

$$\beta := \beta(d) \geqslant \frac{dl}{d - k + 1}. \tag{1.2}$$

An MDS code that attains the bound (1.2) with equality is said to afford optimal repair, and a repair scheme that attains this bound is called optimal. Such codes are also termed *minimum storage regenerating codes* or MSR codes, and the parameter l is called node size or sub-packetization. Multiple constructions of vector MDS codes with optimal repair are available in the literature, including papers [56], [74], [84, 85], [26], [58].

While the aforementioned papers mostly deal with vector codes, we are also interested in the repair problem for scalar MDS codes, more specifically, for *Reed-Solomon codes* or RS codes. This code family continues to attract attention in multiple aspects of theoretical research such as list decoding of variants of RS codes, and it is also one of the most used coding methods in a vast variety of practical

systems. The first work to isolate and advance the repair problem for RS codes was [28] which itself followed and developed the ideas in [66]. In [28], the authors view each coordinate of RS codes as a vector over some subfield and characterize linear repair schemes of RS codes over this subfield. For RS codes (and more generally for scalar codes), the node size l is defined as the degree of extension of the symbol field over the subfield. Following [28], several papers attempted to optimize the repair bandwidth of RS codes [17], [18], [51]. A family of optimal-repair RS codes in the case of repairing a single failed node as well as multiple nodes was constructed in [77].

1.2.2 Stronger notions of optimal repair

The basic repair problem of MDS codes has been extended to the case that some of the helper nodes provide erroneous information (or arbitrary nature). Suppose that a subset of e nodes out of d helpers provide erroneous information and define $\beta(d, e)$ to be the minimum number of symbols needed to repair a failed node in the presence of such errors. It was shown in [57], [53] that for $d \geqslant k + 2e$,

$$\beta(d, e) \geqslant \frac{dl}{d - 2e - k + 1}. \tag{1.3}$$

A repair scheme that achieves this bound is said to have *optimal error correction capability*. Constructions of MDS array codes with optimal error correction capability are presented, for instance, in [84].

Another parameter of erasure codes for distributed storage that affects the

system performance is the so-called access, or input-output cost of repair. Indeed, while the code may support parsimonious exchange between the helper nodes and the repair center, generation of the symbols to be transmitted from the helper node may require reading the entire contents of the node (trivial access), which increases delays in the system. The smallest number of symbols accessed on each of the helper nodes in an MSR code is $l/(d-k+1)$, and such codes are said to have the *optimal access property* or OA property. Advantages of having this property are well recognized in the literature starting with [65], and a number of papers were devoted to constraints that it imposes on the code parameters such as sub-packetization [75], [3]. Many families of MSR codes including early constructions in [8, 74] as well as code families for general parameters in [84, 85], [81] have the optimal access property.

The optimal-access repair and optimal error correction capability can be combined. According to (1.3), we say that a code family/repair scheme have both properties if repair can be performed in the presence of e errors, while the number of symbols accessed on each of the helper nodes equals $1/(d-2e-k+1)$ proportion of the contents of each of d helper nodes.

1.2.3 Optimal repair under connectivity constraints

Initially the problem of repair bandwidth was formulated for the so-called centralized repair model which assumes that the failed nodes are repaired by a single data collector that receives information from the helper nodes and performs the

recovery within a single location, having full access to all the downloaded information and the intermediate results of the calculations [9, 59, 84]. Another well-known model assumes cooperative repair, when the failed nodes are restored at different physical locations, and the information downloaded to each of them as well as the exchange of intermediate results between them are counted toward the overall repair bandwidth [40, 43, 67, 86].

The problems of centralized and cooperative repair have been addressed in multiple recent papers, and there are explicit constructions of optimal-repair regenerating codes that cover the entire range of admissible parameters, require small-size ground alphabet compared to the length n of the encoding block, and attain the smallest possible repair bandwidth [56, 74],[84],[64, 81, 85],[44] (more references are given in a recent survey [2]). The availability of optimal constructions has motivated a shift of attention toward studying data recovery not only under communication, but also *connectivity constraints*, in other words, storage models in which communication cost between nodes differs depending on their location in the storage cluster.

Erasure coding for clustered architectures affords several extensions from the basic setting of homogeneous storage. One of the first questions analyzed for heterogeneous storage models was related to repair under the condition that the system contains a group of nodes, downloading information from which contributes more to the repair bandwidth than downloading the same amount of information from the other nodes [1]. Later works [23, 54] observed that a more realistic version of non-homogeneous storage should assume that the cost of downloading information depends on the relative location of the failed node in the system. In this case, down-

loading information from the group that contains the failed node (also called the *host group*) contributes less to the cost than inter-cluster downloads. The authors of [23, 54] have assumed that the storage is formed of two clusters and derived versions of the cut-set bound for the minimum repair bandwidth. The two-cluster model was further developed in recent papers [68, 69] which assumed that the encoded data is placed in a number of clusters (generally more than two), and derived a cut-set type bound on the repair bandwidth for this case. Moreover, [69] showed existence of optimal-repair codes for their model, and [68] gave an explicit construction of MDS array codes for the case when the code dimension is equal to the size of the cluster. Paper [55] considers several versions of node repair for clustered (rack-aware) storage, but does not address the general case of rack-aware MSR codes. We also mention [61, 79, 84] which discuss other variations of clustered storage architectures and are less related to our work in this dissertation.

The rack-aware storage that we address in this dissertation assumes that k data blocks are encoded into a codeword of length $n = \bar{n}u$ and stored across n nodes. The nodes are organized into $\bar{n}$ groups, also called racks. Suppose that a node has failed and call the rack that contains it the *host rack*. To perform the repair, the system downloads information from the nodes in the host rack (called below *local nodes*), as well as information from the other racks. The rack-oriented storage model is distinguished from the other clustered storage architectures in that the information from nodes that share the same rack, can be processed before communicating it to the failed node. Communication within the racks, including the host rack, does not incur any cost toward the repair bandwidth. The main benefit of rack-aware

coding is related to reducing the bandwidth required for repair compared to coding for homogeneous storage.

This model was introduced in [32, 33]. Specifically, the authors of [32] derived a version of the cut-set bound of [20] adapted for this case and showed existence of MSR codes with optimal repair for the rack model. A more expanded study of codes for this model was undertaken in a recent paper [31], which showed existence of codes with optimal repair bandwidth for a wide range of parameters. At the same time, there are very few explicit constructions of MSR codes for this model known in the literature. We mention [33] which presented such codes for 3 racks and for the case when the number of parities of the code equals the size of the rack u.

1.2.4 Efficiency in terms of locality

In addition to the repair bandwidth, locality is another important metric for assessing the performance of efficient erasure correction. Codes with locality, also known as *locally recoverable codes* or LRC codes, are able to correct one or several erasures in the codeword based on the contents of a subset of other code coordinates, whose cardinality is much smaller than the dimension of the code. In other words, LRC codes support repair of a failed node by contacting a small number of other nodes in the storage system. The problem of local repair was first isolated in [25], and, similarly to the problem of optimal-bandwidth repair, it has been actively studied in the last decade.

Definition 1 (LRC CODES)**.** *A linear code $\mathcal{C} \subset \mathbb{F}_q^n$ is locally recoverable with locality*

r if for every $i \in \{1, 2, \ldots, n\}$ there exists an r-element subset $I_i \subset \{1, 2, \ldots, n\} \backslash \{i\}$ and a linear function $\phi_i : \mathbb{F}_q^r \to \mathbb{F}_q$ such that for every codeword $c \in \mathcal{C}$ we have $c_i = \phi_i(c_{j_1}, \ldots, c_{j_r})$, where $j_1 < j_2 < \cdots < j_r$ are the elements of I_i.

The coordinates in I_i are called the *recovering set* of i, and the set $\{i\} \cup I_i$ is called a *repair group*. Below we refer to a linear LRC code of length n, dimension k, and locality r as an (n, k, r) LRC code. Since the code is occasionally used to correct a large number of erasures (such as in the event of massive system failure), another parameter of interest is the maximum number of erasures that it can tolerate. This is controlled by the minimum distance $d(\mathcal{C})$ of the code, for which there are several bounds known in the literature. We will be interested in the generalized Singleton bound of [25] which states that for any LRC code $\mathcal{C}$,

$$d(\mathcal{C}) \leqslant n - k - \left\lceil \frac{k}{r} \right\rceil + 2. \tag{1.4}$$

LRC codes can be constructed in a number of ways. A connection between LRC codes and the well-studied family of RS codes was put forward in [71], where codes with large distance were constructed as certain subcodes of RS codes. The results of [71] paved the way for using powerful algebraic techniques of coding theory for constructing other families of LRC codes including algebraic geometric codes [5, 46, 47]. In [73] it was observed that a particular class of the codes in [71] can be represented in cyclic form, and the distance and locality properties of cyclic LRC codes were described in terms of the zeros of the code. This established a framework for cyclic LRC codes that was advanced in a number of ways in several recent works

[6, 11, 30, 50].

1.2.5 Further extensions of local repair

While in most situations repairing a single failed node restores the system to the functional state, occasionally there may be a need to recover the data from several concurrent node failures. The following extension of the previous definition is due to [39].

Definition 2 ((r, δ) LOCALITY)**.** *For any $\delta \geqslant 2$ we say that a linear code $\mathcal{C}$ has* (r, δ) locality *if every coordinate $i \in \{1, \ldots, n\}$ is contained in a subset $J_i \subset \{1, \ldots, n\}$ of size at most $r + \delta - 1$ such that the restriction $\mathcal{C}_{J_i}$ to the coordinates in J_i forms a code of distance at least δ.*

Note that in the case of $\delta = 2$ the codes defined here are exactly the codes of Def. 1 above. The case of $\delta > 2$ was also studied in [71], where constructions of RS-like LRC codes with (r, δ) locality were presented. The approach of [73] was later extended by [11] to construct codes with (r, δ) locality, designing a cyclic representation of the polynomial evaluation codes from [71] for the general case of $\delta \geqslant 2$.

An intermediate situation arises when the code is designed to correct a single erasure by contacting a small number r_1 of helper nodes, while at the same time supporting local recovery of multiple erasures. Extending this idea to multiple levels of local protection, the authors of [63] introduced the concept of *hierarchical LRC codes* or H-LRC codes, which are defined as follows.

Definition 3 (H-LRC CODES)**.** *Let* $h \geqslant 1$, $0 < r_1 < r_2 < \ldots < r_h < k$, *and* $1 < \delta_1 \leqslant \ldots \leqslant \delta_h \leqslant d$ *be integers. A linear code* $\mathcal{C} \subset \mathbb{F}_q^n$ *is said to have* h*-level hierarchical locality* $(r_1, \delta_1), \ldots, (r_h, \delta_h)$ *if for every* $1 \leqslant i \leqslant h$ *and every coordinate of the code* $\mathcal{C}$ *there is a punctured code* $\mathcal{C}^{(i)}$ *such that the coordinate is in the support of* $\mathcal{C}^{(i)}$ *and*

(a) $\dim(\mathcal{C}^{(i)}) \leqslant r_i$,

(b) $d(\mathcal{C}^{(i)}) \geqslant \delta_i$,

(c) the i*-th local code* $\mathcal{C}^{(i)}$ *has* $(i-1)$*-level hierarchical locality* $(r_1, \delta_1), \ldots, (r_{i-1}, \delta_{i-1})$.

The authors of [63] proved the following extension of the bound (1.4): The minimum distance of an h-level H-LRC code with locality satisfies the inequality

$$d \leqslant n - k + \delta_h - \sum_{i=1}^{h} \left\lceil \frac{k}{r_i} \right\rceil (\delta_i - \delta_{i-1}), \tag{1.5}$$

where $\delta_0 = 1$. In particular, for $h = 1$ this gives a version of the bound (1.4) for the distance of an (n, k) code with (r, δ) locality:

$$d \leqslant n - k + \delta - \left\lceil \frac{k}{r} \right\rceil (\delta - 1). \tag{1.6}$$

We call an H-LRC code *optimal* if its distance attains the bound (1.5). Note that it is possible that the code is optimal while its local codes $\mathcal{C}^{(i)}$ for some or even all $i \leqslant h-1$ are not. We say that an H-LRC code $\mathcal{C}$ is *strongly optimal* if in every level $i, 1 \leqslant i \leqslant h$, the i-th local codes are optimal H-LRC codes with $i-1$ levels. For

$h = 1$ the distance of the optimal code with (r, δ) locality attains the bound (1.6) with equality.

In addition to defining the problem and deriving a bound on the parameters of H-LRC codes, the authors of [63] extended the construction of [71] to the hierarchical case. Their construction was further generalized to algebraic geometric codes in [4]. On the other hand, several recent studies presented families of codes with multiple levels of erasure correction [22, 27, 83], not necessarily within the framework of the above definition. As far as H-LRC codes are concerned, the only general family of optimal H-LRC codes that we are aware of was presented in [4, 63]. This construction essentially followed the approach of [71], relying on multivariate polynomials that are constant on the blocks that form the support of the code $\mathcal{C}^{(i)}$ in Def. 3. The codes of [4, 63] form a family of strongly optimal H-LRC codes which can be constructed for any code length $n \leqslant q$, dimension k, and any values of $r_i, i = 1, \ldots, h$ as long as $r_i | r_{i+1}, i = 1, \ldots, h-1$ and $r_h | k$. The divisibility constraint is essential for the constructions discussed, and it limits the possible choices of the code parameters.

Another extension of the local repair is the problem of *availability* which calls for constructing LRC codes with several disjoint recovering sets for each code coordinate. LRC codes with this property are defined as follows [60, 82].

Definition 4 (LRC CODES WITH AVAILABILITY)**.** *Let $t \geqslant 1$ and $(r_1, \delta_1), \ldots, (r_t, \delta_t)$ be integers. A code $\mathcal{C} \subset \mathbb{F}_q^n$ is said to have availability t with locality $(r_1, \delta_1), \ldots, (r_t, \delta_t)$ if for every coordinate $j, 1 \leqslant j \leqslant n$ of the code $\mathcal{C}$ there are t punctured codes $\mathcal{C}^{(i)}, 1 \leqslant i \leqslant t$ such that $j \in \mathrm{supp}(\mathcal{C}^{(i)}), i = 1, \ldots, t$ and*

(a) $\dim(\mathcal{C}^{(i)}) \leqslant r_i$,

(b) $d(\mathcal{C}^{(i)}) \geqslant \delta_i$,

(c) $\bigcap_{i=1}^{t} \operatorname{supp}(\mathcal{C}^{(i)}) = \{j\}$.

We note that the known bounds on the code parameters for multiple recovering sets [41, 60, 72, 82] do not support a conclusive picture, and we are not aware of general families of codes with availability whose distance attains one of the known upper limits.

1.2.6 LRC convolutional codes

LRC convolutional codes form another class of erasure codes, which was considered in several previous works [15, 35, 87] before being thoroughly analyzed in a recent paper [52]. The results of [52] focused on the so-called sliding window repair property of convolutional codes, and the authors observed that certain families of convolutional codes, notably the so-called codes with the maximum distance profile [24, 80], suggest an approach to constructing codes with locality. They also presented a family of LRC convolutional codes with sliding window repair for the case of *column locality*.[2]

1.3 Contributions

We address a range of questions on codes with optimal repair bandwidth and on codes with locality, corresponding to the two metrics of interest for efficient

[2]See Sec. 4.4.1 for more details.

erasure correction: bandwidth and locality.

1.3.1 Codes with optimal repair bandwidth

The repair problem of scalar MDS codes.

First, we address two problems related to RS repair, namely, (i) repair schemes of RS codes with optimal error correction, and (ii) RS codes with optimal-access repair. Error correction during repair of failed nodes was previously only considered for vector codes [57], [53], [84]. The problem of low-access RS codes was studied in [16, 19, 45]. In particular, the last of these works analyzed the access (input/output) cost of the family of RS codes of [77], providing an estimate of this parameter, but stopping short of achieving optimal access.

Our results provide a solution to problems (i)-(ii). Specifically, we construct a repair scheme for RS codes in [77] that has optimal error correction capability (i.e., attains the bound (1.3)). This is accomplished by enforcing the information provided by helper nodes for repair to form codewords of an appropriate MDS code, and thus the error correction capability of the MDS code gives rise to an optimal error-tolerance repair scheme. We also construct a family of RS codes with optimal access repair for any single failed node. The starting point of our optimal-access construction is the observation that the RS code family presented in [77] actually affords optimal-access repair of one node but not all the other nodes. In view of this, we impose additional structures on the underlying finite field that generalize what is needed for the optimal-access repair of one node, which makes it possible

for every node to be repaired with optimal access. Additionally, combining both ideas discussed herein, we prove that the constructed codes with optimal access can be furnished with a repair scheme that supports both optimal error correction and optimal-access repair simultaneously.

Apart from this, we also show that any scalar MDS code with optimal repair of a single node from d helpers, $k \leqslant d \leqslant n-1$, affords a repair scheme with optimal access, and this includes the RS codes in [77]. While our arguments do not provide an explicit construction, we give a combinatorial search procedure, showing that it exists for any scalar MSR code. The resulting optimal access codes have the same sub-packetization as the original MDS codes.

The rack-aware storage model.

We present constructions of MSR codes for the rack-aware storage model that have optimal repair bandwidth and cover all admissible parameters, such as the code rate k/n, the size and number of the racks. The only restriction that we assume is the natural condition that the racks are of equal size u and that the codeword is written on $\bar{n}$ racks such that u coordinates of the codeword are placed on each of them. This assumption is also consistent with the literature [31, 32]. The main idea that underlies these constructions is the multiplicative group structure of finite fields. More precisely, the multiplicative structure enables one to aggregate and compress the information provided by the nodes within in a helper rack that is needed for repair of the failed node. The compressed information, together with the free information from the local nodes in the host rack, enables the optimal repair of

the failed node.

We present two families of MDS array codes that support optimal repair in the rack model. The first family gives an explicit construction of optimal-bandwidth codes for repairing a *single node* from the nodes located in $\bar{d}$ helper racks for any $\lfloor k/u \rfloor \leqslant \bar{d} \leqslant \bar{n} - 1$. The underlying finite field of our construction is of size at most n^2/u where u is the size of the rack, and the node size (sub-packetization) equals $l \approx (\bar{d} - \frac{k}{u})^{n/u}$. The construction is phrased in terms of the parity-check equations of the code, as in [84, 85], and relies on the multiplicative structure of the field to account for the rack model considered here.

The second code family constructed in this paper, in addition to optimal repair, addresses the question of reducing the number of symbols accessed on each of the helper racks. The code construction is presented in two steps. First, we present a new family of optimal-access codes for the standard repair problem (homogeneous storage), constructing codes with arbitrary repair degree $d, k \leqslant d \leqslant n-1$ over a field F of size at least $d - k + 1$. These parameters are similar to optimal-access codes constructed in [84], and in fact require a slightly larger field F. At the same time, the new construction can be modified for the rack model, resulting in codes with low access. The additional ingredient that enables low access is that we devise the parity check equations carefully such that the content of any single node and the contents of a $1/(d-k+1)$ fraction of the other nodes constitute codewords of certain MDS codes of dimension d. Thus, any d helper nodes suffice to repair the failed node with optimal access under the homogeneous storage model. Furthermore, with the multiplicative group structure discussed above, this leads to a family of low-access

codes for the rack-aware storage model.

We also present a family of (scalar) RS codes that can be optimally repaired in the rack model. The construction is a modified version of the RS code family constructed in [77] for the case of homogeneous storage. Furthermore, extending this approach and utilizing ideas that form the basis of the RS codes with optimal error correction and optimal access mentioned above, we are able to construct a family of RS codes with optimal error correction capability and low access for the rack-aware storage model.

Apart from the code constructions, we examine the structure of codes with optimal repair or optimal access for the rack model. Because of intra-rack processing, the definition of optimal access is not as explicit as in the homogeneous case. We prove a lower bound on the number of accessed symbols for codes that support optimal repair. At the same time, the codes that we construct fall short of attaining this bound, and it is not clear what is the correct value of this quantity.

Finally, we derive a lower bound on the node size for optimal-repair codes in the rack model, modifying for this purpose the approach of the recent work [3], where a similar bound was proved for the homogeneous case.

We note that the results of [69] and [68] do not allow data processing within clusters (racks) in the course of the repair task, and thus are not directly comparable with our findings. Subsequent to our work [13], the repair problem of RS codes for rack-aware storage was consider in [36] for a different range of storage parameters.

1.3.2 Codes with locality

Cyclic LRC codes.

In the part on cyclic codes we focus on several aspects of LRC codes that have not been previously addressed in the literature. The first of these is codes with hierarchical locality (H-LRC codes). The starting point of our constructions is a cyclic version of the RS-like codes with locality designed in [71]. As noted above, RS codes over $\mathbb{F}_q$ can be alternatively described in terms of polynomial evaluation and (in the case that the code length n divides $q-1$) as cyclic codes of the BCH type.

We first derive conditions on the zeros of a cyclic code that support hierarchical locality. As a result, we construct families of cyclic H-LRC codes for any levels of hierarchy. We also derive conditions that are sufficient for our codes to be (strongly) optimal which do not rely on the divisibility assumptions prevailed in the literature, thus yielding a new range of code parameters.

Furthermore, we examine two other problems for LRC codes that benefit from the cyclic code construction. The first of them is the problem of maximum length of optimal LRC codes put forward in [29]. Answering the challenge of constructing optimal LRC codes of length larger than q, the authors of [6, 10, 29, 50] constructed several families of optimal cyclic codes of large, and in some cases even unbounded length, and [10] extended these results to the case of several erasures. Here we follow the lead of [50] and construct an infinite family of H-LRC codes over a given finite

field and establish conditions for their optimality in terms of the bound (1.5).

Finally, we consider the problem of *availability*, namely, LRC codes with multiple recovering sets for each coordinate of the code. We note that multidimensional cyclic LRC codes (product codes of cyclic LRC codes) naturally yield several recovering sets for the coordinates. We use a description of bi-cyclic codes in terms of their zeros together with a special version of code concatenation [62] to construct codes with availability and rate higher than the rate of product codes.

Although we do not pursue this direction here, let us note that the methods of constructing cyclic codes with locality presented in this dissertation enable one to construct codes with both hierarchical locality and availability. We remark that constructions of LRC codes that have both properties were presented in [4], where the main tools were fiber products and covering maps of algebraic curves.

LRC convolutional codes.

The results on H-LRC cyclic codes also enable us to connect the construction of H-LRC cyclic codes and convolutional codes with locality. In fact, the recent work [52] suggested that there may be a connection between H-LRC codes and LRC convolutional codes. We show that this connection indeed leads to fruitful results, designing LRC convolutional codes for the case of *row locality* (defined in Sec 4.4).

The lower bounds on the column distance[3] of the codes constructed here and in [52] are the same; however the alphabet size of our codes is much smaller than in [52]. We also derive an upper bound on the column distance of LRC convolutional codes

[3] A distance measure of interest for convolutional codes. See Def. 10 in Sec. 4.4 for more details.

with locality; however, our construction falls short of attaining it. The method that we use relies on the characterization of zeros of cyclic block H-LRC codes. We observe that several levels of hierarchy enable one to put our cyclic H-LRC codes in an appropriate quasicyclic form, and then use a classic connection between quasicyclic codes and convolutional codes [70] to construct convolutional codes with locality.

1.4 Organization

The dissertation is organized as follows. Following this introductory chapter, the main part of this dissertation is divided into three chapters, presenting our main results in detail. Chapter 2 and 3 are devoted to the studies of codes with optimal repair bandwidth, while Chapter 4 focuses on codes with various types of locality.

Chapter 2 is dedicated to the repair problem of scalar MSR codes, in particular, to the studies of the properties of error correction and optimal access for repairing RS codes. This chapter is based on the paper [14].

Chapter 3 concentrates on the rack-aware storage model, analyzing properties that are unique for the model and construct families of MSR codes tailor-made for the model. This chapter is built upon the paper [13].

Chapter 4 examines cyclic codes with hierarchical locality and availability as well as convolutional codes with locality. This chapter is formed on the paper [12].

Finally, Chapter 5 concludes this dissertation and points out some open problems. Proofs omitted from the main text are collected in the Appendices.

Chapter 2: Enabling Optimal Access and Error Correction for the Repair of Reed-Solomon Codes

2.1 Introduction

In this chapter we study the repair problem of scalar MDS codes. RS codes were shown to possess a repair scheme that supports repair of failed nodes with optimal repair bandwidth. We extend this result in two directions. First, we propose a new repair scheme for the RS codes constructed in [77] and show that repair is robust to erroneous information provided by the helper nodes while maintaining the optimal repair bandwidth. Second, we construct a new family of RS codes with optimal access for the repair of any single failed node. We also show that the constructed codes can accommodate both features, supporting optimal-access repair with optimal error-correction capability.

Going beyond RS codes, we also prove that any scalar MDS code with repair bandwidth attaining the cutset bound affords a repair scheme with optimal access property.

2.1.1 Organization

The constructions are technically involved, and we begin with illustrating them in an example in Sec. 2.2. In Sec. 2.3 we present a repair scheme that supports optimal error correction for the RS codes family constructed in [76]. Then in Sec. 2.4 we construct a new family of RS codes that affords optimal-access repair and this family is shown to admit an explicit repair scheme that supports both optimal error correction and optimal access simultaneously. Sec. 2.5 that follows proves that any scalar MSR code has a repair scheme with optimal access property.

2.2 A simple example

In this section, we construct an RS code together with a repair scheme that can recover its *first node* with both optimal access and optimal error correction capability.

2.2.1 Preliminaries

1) We begin with some standard definitions. Recall that a *generalized RS code* or GRS code of length n and dimension k over a finite field F is obtained by fixing a set of n distinct evaluation points $\Omega := \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \subset F$ and a vector $(v_1, \ldots, v_n) \in (F^*)^n$ with no zero coordinates. Then the GRS code is the set of vectors

$$\mathrm{GRS}_F(n, k, v, \Omega) = \{(v_1 f(\alpha_1), v_2 f(\alpha_2), \ldots, v_n f(\alpha_n)) : f \in F[x], \deg f < k\}.$$

In particular, if $(v_1, \ldots, v_n) = (1, \ldots, 1)$, then the GRS code is called the Reed-Solomon (RS) code and is denoted by $\mathrm{RS}_F(n, k, \Omega)$. It is a classic fact that the dual code $(\mathrm{RS}_F(n, k, \Omega))^\perp$ is $\mathrm{GRS}_F(n, n-k, v, \Omega)$, where $v \in (F^*)^n$ is some vector. In particular, if $c = (c_1, \ldots, c_n) \in F^n$ is a vector such that $\sum_{i=1}^n c_i h(\alpha_i) = 0$ for every polynomial $h(x)$ of degree $\leqslant k-1$, then c is contained in a GRS_F code of dimension $n-k$. Rephrasing this, we have the following obvious proposition that will be frequently used below.

Proposition 1. *Let $c = (c_1, \ldots, c_n) \in F^n$ and suppose that $\sum_{i=1}^n c_i \alpha_i^t = 0$ for all $t = 0, 1, \ldots, k-1$. Then the vector c is contained in a code $GRS_F(n, n-k, v, \Omega)$, where $v \in (F^*)^n$ and $\Omega = \{\alpha_1, \ldots, \alpha_n\}$.*

Let E be an algebraic extension of F of degree s. The *trace mapping* $\mathrm{tr}_{E/F}$ is given by $x \mapsto 1 + x^{|F|} + x^{|F|^2} + \cdots + x^{|F|^{s-1}}$. For any basis $\gamma_0, \ldots, \gamma_{s-1}$ of E over F there exists a *trace-dual basis* $\delta_0, \ldots, \delta_{s-1}$, which satisfies $\mathrm{tr}_{E/F}(\gamma_i \delta_j) = \mathbb{1}_{\{i=j\}}$ for all pairs i, j. For an element $x \in E$ the coefficients of its expansion in the basis (γ_i) are found using the dual basis, specifically, $x = \sum_{i=0}^{s-1} \mathrm{tr}_{E/F}(x\delta_i)\gamma_i$. As a consequence, for any basis (δ_i) the mapping $E \to F^s$ given by $x \mapsto (\mathrm{tr}(x\delta_i), i = 0, \ldots, s-1)$ is a bijection.

2) Before we define the RS code that will be considered below, let us fix the parameters of the repair scheme. We attempt to repair a failed node using information from d helper nodes. Suppose that at most e of them provide erroneous information. Assume that $d-2e \geqslant k$, and let $s := d-2e-k+1$. Let F be a finite field of size $|F| \geqslant n-1$. Choose a set of distinct evaluation points $\Omega := \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$

such that $\alpha_i \in F$ for all $2 \leqslant i \leqslant n$ and α_1 is an algebraic element of degree s over F (which means that the extension field $E := F(\alpha_1)$ forms an s-dimensional vector space over F). Consider the code

$$\mathcal{C} := \mathrm{RS}_E(n, k, \Omega).$$

In this section we present a repair scheme of the code $\mathcal{C}$ that can repair the first node of $\mathcal{C}$ over the field F; in other words, we represent the coordinates of $\mathcal{C}$ as s-dimensional vectors over F in some basis of E over F. Thus, the node size of this code is s. We note that the code $\mathcal{C}$ represented in this way is still a scalar code.

The repair scheme presented below has the following two properties:

- the optimal error correction capability, i.e., the repair bandwidth achieves the bound (1.3) for any pair (d, e) such that $d - 2e = s + k - 1$;
- in the absence of errors it has the optimal access property, i.e., the number of symbols accessed during the repair process is d. Thus, in this case $e = 0$ and $s = d - k + 1$.

2.2.2 Repair scheme with optimal error correction capability

Let $c = (c_1, c_2, \ldots, c_n) \in \mathcal{C}$ be a codeword and suppose that c_1 is erased. Since $\mathcal{C}^\perp = \mathrm{GRS}_E(n, n-k, v, \Omega)$ for some $v \in (E^*)^n$, we have

$$v_1\alpha_1^t c_1 + v_2\alpha_2^t c_2 + \cdots + v_n\alpha_n^t c_n = 0, \quad t = 0, 1, \ldots, n-k-1,$$

or

$$v_1\alpha_1^t c_1 = -v_2\alpha_2^t c_2 - \cdots - v_n\alpha_n^t c_n, \quad t = 0, 1, \ldots, n-k-1. \tag{2.1}$$

Evaluating the trace $\mathrm{tr} = \mathrm{tr}_{E/F}$ on both sides of (2.1), we obtain the relation

$$\begin{aligned}\mathrm{tr}(v_1\alpha_1^t c_1) &= -\mathrm{tr}(v_2\alpha_2^t c_2) - \cdots - \mathrm{tr}(v_n\alpha_n^t c_n) \\ &= -\alpha_2^t \mathrm{tr}(v_2 c_2) - \cdots - \alpha_n^t \mathrm{tr}(v_n c_n), \quad t = 0, 1, \ldots, n-k-1, \end{aligned} \tag{2.2}$$

where the second equality follows from the fact that $\alpha_2, \ldots, \alpha_n \in F$. Therefore, knowing the values of $(\mathrm{tr}(v_2c_2), \ldots, \mathrm{tr}(v_nc_n))$ enables us to compute $\mathrm{tr}(v_1\alpha_1^t c_1)$ for all $0 \leqslant t \leqslant n-k-1$. Since $\deg_F(\alpha_1) = s$, the elements $1, \alpha_1, \ldots, \alpha_1^{s-1}$ form a basis of E over F. As a consequence, one can recover c_1 from the values of $\{\mathrm{tr}(v_1\alpha_1^t c_1) : 0 \leqslant t \leqslant s-1\}$. By definition, $s - 1 = d - 2e - k \leqslant n - k - 1$, so $\{\mathrm{tr}(v_1\alpha_1^t c_1) : 0 \leqslant t \leqslant s-1\} \subseteq \{\mathrm{tr}(v_1\alpha_1^t c_1) : 0 \leqslant t \leqslant n-k-1\}$. Combining this with (2.2), we see that the value c_1 is fully determined by the set of elements $(\mathrm{tr}(v_2c_2), \ldots, \mathrm{tr}(v_nc_n))$.

Recalling our problem, we will show that in order to repair c_1, it suffices to acquire the values $\mathrm{tr}(v_ic_i)$ from any d helper nodes provided that at least $d - e = (d + s + k - 1)/2$ of these values are correct. This will follow from the following proposition.

Proposition 2. *Let $f(x) \in F[x]$ be the minimal polynomial of α_1. For any $s < n - k$ and any $c = (c_1, \ldots, c_n) \in \mathcal{C}$ the vectors $(f(\alpha_2)\,\mathrm{tr}(v_2c_2), \ldots, f(\alpha_n)\,\mathrm{tr}(v_nc_n))$*

are contained in an $(n-1, s+k-1)$ *GRS code over* F.

Proof. Let $\mathcal{T} := \{0, 1, \ldots, n-k-s-1\}$. Since $\alpha_i \in F, i = 2, \ldots, n$ by definition we have $f(\alpha_i) \neq 0$ for all such i. Next, $\deg(f) = s$, and thus for all $t \in \mathcal{T}$

$$(v_1\alpha_1^t f(\alpha_1), v_2\alpha_2^t f(\alpha_2), \ldots, v_n\alpha_n^t f(\alpha_n)) \in \mathcal{C}^{\perp}.$$

This implies that for all $t \in \mathcal{T}$

$$v_1\alpha_1^t f(\alpha_1)c_1 + v_2\alpha_2^t f(\alpha_2)c_2 + \cdots + v_n\alpha_n^t f(\alpha_n)c_n = 0,$$

but $f(\alpha_1) = 0$, so taking the trace, we obtain

$$\alpha_2^t f(\alpha_2)\operatorname{tr}(v_2c_2) + \cdots + \alpha_n^t f(\alpha_n)\operatorname{tr}(v_nc_n) = 0, \quad t \in \mathcal{T}. \tag{2.3}$$

By Proposition 1, this implies that the vectors $(f(\alpha_2)\operatorname{tr}(v_2c_2), \ldots, f(\alpha_n)\operatorname{tr}(v_nc_n))$ are contained in a GRS code of length $n-1$ with $n-s-k$ parities. □

The GRS code identified in this proposition can be punctured to any subset $\mathcal{R}$ of d coordinates, retaining the dimension and the MDS property. This means that the punctured code is capable of correcting any $e = (d-s-k+1)/2$ errors. Therefore, as long as no more than e helper nodes provide incorrect information, we can always recover $(\operatorname{tr}(v_2c_2), \ldots, \operatorname{tr}(v_nc_n))$ by acquiring a subset $\{\operatorname{tr}(v_{i_j}c_{i_j}), j = 1, \ldots, d\}$ from any d helper nodes and correcting the errors based on any decoding procedure of the underlying MDS code. Finally note that the case $s = n-k$ can be added

trivially because then $d = n - 1$ and $e = 0$, so all the helper nodes provide accurate information, and no error correction is required (or possible).

2.2.3 Optimal access property

Following the discussion in the first part of this section, we show that the code $\mathcal{C} = \mathrm{RS}_E(n, k, \Omega)$ defined above supports optimal-access repair of the node c_1. In this part we assume that the helper nodes provide accurate information about their contents, and we do not attempt error correction.

To represent the code, we choose a pair of trace-dual bases $(b_i), (b_i^*)$ of E over F, where we assume w.l.o.g. that $b_0 = 1$. Next, represent the ith coordinate of the code, $i \in \{1, \ldots, n\}$, using the basis $(v_i^{-i} b_m, m = 0, \ldots, s-1)$, where $(v_1, \ldots, v_n)$ is defined by the code $\mathcal{C}^\perp$. Namely, for a codeword $c \in \mathcal{C}$ we have

$$c_i = v_i^{-1} \sum_{m=0}^{s-1} c_{i,m} b_m^*, \tag{2.4}$$

where $c_{i,m} \in F$ for all $m = 0, 1, \ldots, s-1$. We assume that each storage node contains the vector $(c_{i,0}, c_{i,1}, \ldots, c_{i,s-1})$.

As discussed above, the value c_1 can be recovered from any d-subset of the set of elements $\{\mathrm{tr}(v_i c_i), j = 2, \ldots, n\}$. Further, for all $i = 2, \ldots, n$ and $m = 0, \ldots, s-1$ we have $\mathrm{tr}(v_i c_i b_m) = c_{i,m}$, so in particular,

$$\mathrm{tr}(v_i c_i) = c_{i,0}.$$

Thus, to repair c_1 it suffices to access and download a single symbol $c_{i,0}$ from the chosen subset of d helper nodes. According to the bound (1.1), the minimum number of symbols downloaded from a helper node for optimal repair is the $(1/s)$th proportion of the node's contents. Overall this shows that the repair scheme considered above has the optimal access property.

The above discussion sets the stage for constructing RS codes with optimal-access repair for each of the n coordinates. Namely, we took a basis $1, b_1, \ldots, b_{s-1}$ of E over F and represented each c_i in the basis $(v_i^{-1} b_i^*)$. The only element of the helper coordinate that we access and download is $c_{i,0}$. For more complicated constructions of RS codes, e.g., the ones constructed in [77] and below in the chapter, we assume that E is an l-degree extension of F. The known repair schemes require to download elements of the form $\mathrm{tr}(v_i c_i a_0), \mathrm{tr}(v_i c_i a_1), \ldots, \mathrm{tr}(v_i c_i a_{(l/s)-1})$, where $a_0, a_1, \ldots, a_{(l/s)-1}$ are linearly independent over F. In this case, we can extend the set $a_0, a_1, \ldots, a_{(l/s)-1}$ to a basis (b_i) of E over F. Following the approach in (2.4), we store the code coordinate c_i as the vector of its coefficients $(c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})$ in the dual basis $(b_i^*, i = 0, \ldots, l-1)$ of the basis (b_i). Since $c_{i,m} = \mathrm{tr}(v_i c_i a_m)$ for all $m = 0, 1, \ldots, l/s - 1$, this choice of the basis enables one to achieve optimal access. This idea underlies the construction presented below in Sec. 2.4.1.

2.2.4 Optimal access with error correction

Thus far, we have assumed that errors are absent for optimal-access repair. To complete the picture, we address the case of codes with both optimal access and

optimal error correcting capability for the repair of node c_1. It is easily seen that both properties can be combined. Indeed, since $\mathrm{tr}(v_i c_i) = c_{i,0}$ for all $i = 2, \ldots, n$, and since by Proposition 2 these elements form a codeword of a GRS code, it is immediately clear that c_1 can be repaired with optimal error correction capability and optimal access. To enable this property for any c_i, below we add extra features to the general repair scheme with optimal access. Specifically, error correction and optimal access are based on two different structures supported by the code. We show that it is possible to realize the error-correction structure in an extension field located between the base field and the symbol field of the code. Further reduction to the base field enables us to perform repair with optimal access. These ideas are implemented in detail in Sec. 2.4.2 below.

2.3 Enabling error correction for repair of RS codes

In this section we propose a new repair scheme for the optimal-repair family of RS codes of [77] that supports the optimal error correction capability.

2.3.1 Preliminaries

We begin with briefly recalling the definition of the subfamily of RS codes of [77]. The construction depends on the number of helper nodes d used for the purpose of repair of a single node, $k \leqslant d \leqslant n-1$.

Definition 5 ([77])**.** *Let p be a prime, let $s := d-k+1$, and let $p_1, \ldots, p_n$ be distinct primes that satisfy the condition $p_i \equiv 1 \bmod s, i = 1, \ldots, n$, Let $\mathcal{C} := RS_{\mathbb{K}}(n, k, \Omega)$*

be a Reed-Solomon code, where

- $\Omega = \{\alpha_1, \ldots, \alpha_n\}$, *where* $\alpha_i, i = 1, \ldots, n$ *is an algebraic element of degree* p_i *over* $\mathbb{F}_p$,
- $\mathbb{K} = \mathbb{F}(\beta)$, *where* β *is an algebraic element of degree* s *over* $\mathbb{F} := \mathbb{F}_p(\alpha_1, \ldots, \alpha_n)$.

As shown in [77], this code supports optimal repair of any node i from any set of d helper nodes in $[n]\backslash\{i\}$. Below we use this construction, choosing the value of s based not only on the number of helpers but also on the target number of errors tolerated by the repair procedure.

In this section we consider an RS code $\mathcal{C}$ given by Def. 5, where we take $s = d - 2e - k + 1$. For this code we will present a new repair scheme that has the property of optimal error correction. This repair scheme as well as the original repair scheme developed in [77] rely on the following lemma:

Lemma 3 ([77], Lemma 1)**.** *Let* F *be a finite field. Let* r *be a prime such that* $r \equiv 1 \bmod s$ *for some* $s \geqslant 1$. *Let* α *be an element of degree* r *over* F *and* β *be of degree* s *over the field* $F(\alpha)$. *Let* $K = F(\alpha, \beta)$ *be the extension field of degree* rs. *Consider the* F*-linear subspace* S *of dimension* r *with the basis*

$$E := \{\beta^u \alpha^{u+qs} \mid u = 0, \ldots, s-1; q = 0, \ldots, \tfrac{r-1}{s} - 1\} \bigcup \Big\{ \sum_{u=0}^{s-1} \beta^u \alpha^{r-1} \Big\}.$$

Then $S + S\alpha + \cdots + S\alpha^{s-1} = K$, *and this is a direct sum.*

Without loss of generality, we only present the repair scheme for the first node c_1, and all the other nodes can be repaired in the same way (this is different

from the previous section where the code was designed to support optimal repair *only* of the node c_1). The scheme is complicated, and we take time to develop it, occasionally repeating similar arguments more than once rather than compressing the presentation.

The repair of c_1 is conducted over the field $F_1 := \mathbb{F}_p(\alpha_2, \alpha_3, \ldots, \alpha_n)$. It is clear that $\mathbb{F} = F_1(\alpha_1)$ and $\mathbb{K} = \mathbb{F}(\beta)$, where $\deg_{F_1}(\alpha_1) = p_1$ and $\deg_{\mathbb{F}}(\beta) = s$. Below we use $\mathrm{tr} = \mathrm{tr}_{\mathbb{K}/F_1}$ to denote the trace mapping from $\mathbb{K}$ to F_1.

Define the set

$$E_1 := \{\beta^u \alpha_1^{u+qs} \mid u = 0, \ldots, s-1; q = 0, \ldots, \tfrac{p_1-1}{s} - 1\} \bigcup \Big\{ \sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1} \Big\}. \quad (2.5)$$

Clearly, $|E_1| = p_1$, and we write the elements in E_1 as $e_0, e_1, \ldots, e_{p_1-1}$. Then Lemma 3 implies that the set of elements

$$\{e_i \alpha_1^j : i = 0, \ldots, p_1 - 1, j = 0, \ldots, s-1\} \quad (2.6)$$

forms a basis of $\mathbb{K}$ over F_1.

Let $\mathcal{C}^\perp = \mathrm{GRS}_{\mathbb{K}}(n, n-k, v, \Omega)$ be the dual code. For every codeword $(c_1, \ldots, c_n) \in \mathcal{C}$ we have

$$v_1 \alpha_1^t c_1 + v_2 \alpha_2^t c_2 + \cdots + v_n \alpha_n^t c_n = 0, \qquad t = 0, 1, \ldots, n-k-1.$$

Multiplying by e_i on both sides of the equation and evaluating the trace, we obtain

the relation

$$\begin{aligned}\operatorname{tr}(e_i v_1 \alpha_1^t c_1) &= -\sum_{j=2}^{n} \operatorname{tr}(e_i v_j \alpha_j^t c_j) \\ &= -\sum_{j=2}^{n} \alpha_j^t \operatorname{tr}(e_i v_j c_j), \quad t = 0, 1, \ldots, n-k-1, \end{aligned} \tag{2.7}$$

where the second equality follows since $\alpha_j \in F_1$ for all $2 \leqslant j \leqslant n$. Therefore, the elements $\{\operatorname{tr}(e_i v_j c_j) : 2 \leqslant j \leqslant n\}$ suffice to compute $\{\operatorname{tr}(e_i v_1 \alpha_1^t c_1) : 0 \leqslant t \leqslant n-k-1\}$. Since $s = d - 2e - k + 1 \leqslant d - k + 1 \leqslant n - k$, we can calculate $\{\operatorname{tr}(e_i v_1 \alpha_1^t c_1) : 0 \leqslant t \leqslant s-1\}$ from $\{\operatorname{tr}(e_i v_j c_j) : 2 \leqslant j \leqslant n\}$. Thus knowing the values of $\{\operatorname{tr}(e_i v_j c_j) : 2 \leqslant j \leqslant n, 0 \leqslant i \leqslant p_1 - 1\}$ suffices to find the set of elements

$$\{\operatorname{tr}(e_i v_1 \alpha_1^t c_1) : 0 \leqslant t \leqslant s-1, 0 \leqslant i \leqslant p_1 - 1\}. \tag{2.8}$$

Since the set (2.6) forms a basis of $\mathbb{K}$ over F_1, the set $\{e_i v_1 \alpha^t : 0 \leqslant i \leqslant p_1 - 1, 0 \leqslant t \leqslant s-1\}$ also forms a basis of $\mathbb{K}$ over F_1, and therefore we can recover c_1 from (2.8). In conclusion, to recover c_1, it suffices to know the set of elements $\{\operatorname{tr}(e_i v_j c_j) : 2 \leqslant j \leqslant n, 0 \leqslant i \leqslant p_1 - 1\}$.

2.3.2 The repair scheme

For $j = 2, 3, \ldots, n$ define the vector $r_j := (\operatorname{tr}(e_i v_j c_j), i = 0, \ldots, p_1 - 1)$. In this section we design invertible linear transformations M_j that send these vectors to a set of vectors z_j that support error correction. The following proposition underlies our repair scheme.

Proposition 4. *Consider the set of vectors* $z_j = (z_{j,0}, z_{j,1}, \ldots, z_{j,p_1-1}), j = 2, 3, \ldots, n$ *defined by*

$$z_j^T = M_j r_j^T, \tag{2.9}$$

where $M_2, \ldots, M_n$ *are invertible matrices of order* p_1*. Suppose that for every* $i = 0, 1, \ldots, p_1 - 1$*, the vector* $(z_{2,i}, z_{3,i}, \ldots, z_{n,i})$ *is contained in an MDS code of length* $n-1$ *and dimension* $s+k-1$*. Then there is a repair scheme of the code* $\mathcal{C}$ *that supports recovery of the node* c_1 *with optimal error correction capability.*

Note that, by the closing remark in Sec. 2.2.2, it suffices to assume that $s < n - k$.

Proof. If $(z_2, z_3, \ldots, z_n)$ is a codeword in an MDS array code of length $n-1$ and dimension $s+k-1$, then the punctured codeword $(z_j : j \in \mathcal{R})$ is contained in an MDS array code of length $d = |\mathcal{R}|$ and dimension $s + k - 1 = d - 2e$, and such the code can correct any e errors.

To repair the failed node c_1, we download p_1-dimensional vectors $\hat{r}_j, j \in \mathcal{R}$, where $\mathcal{R} \subset [n] \backslash \{1\}, |\mathcal{R}| = d$ is a set of d helper nodes. For all but e or fewer values of j, we have $\hat{r}_j = r_j$. The repair scheme consists of the following steps:

(i) Find the vectors $\hat{z}_j^T = M_j \hat{r}_j^T, j \in \mathcal{R}$,

(ii) Find the vectors $z_j, j \in \mathcal{R}$ using the error correction procedures of the underlying MDS codes,

(iii) For every $i = 0, \ldots, p_1 - 1$ use the d-subset $\{z_{j,i}, j \in \mathcal{R}\}$ to recover the codeword $(z_{2,i}, z_{3,i}, \ldots, z_{n,i})$,

(iv) Find the vectors $r_j^T = M_j^{-1} z_j^T, j = 2, \ldots, n-1$ and finally recover c_1.

Step (ii) is justified by the fact that, by assumption, at most e of the elements $\hat{z}_j$ are incorrect. In step (iii) we rely on the fact that d symbols of the MDS codeword suffice to recover the remaining $n-1-d$ symbols, and in step (iv) we use invertibility of the matrices M_j and recover c_1 using (2.7), (2.8).

The total number of downloaded symbols of F_1 equals $p_1 d$, and it is easy to verify that the repair bandwidth of our scheme meets the bound (1.3) with equality.

□

Why do we need the matrices M_j and why were they not involved in the example in Sec. 2.2.2? The answer is related to the fact that we need to remove the failed node from consideration and obtain a codeword of the MDS code that contains all the other nodes. In the example the degree of the minimal polynomial of α_1, denoted $f(x)$, is $s < n-k$, so the evaluations of $x^t f$ are dual codewords (see (2.3) in Prop. 2). This implies that the downloaded symbols form a codeword in an MDS code over F which supports error correction. Importantly, this codeword does not involve the erased coordinate.

Switching to the RS codes of [77] considered here, the element α_1 is of degree p_1 over the repair field $F(\alpha_2, \ldots, \alpha_n)$, and generally $p_1 > n-k-1$, so the minimal polynomial of α_1 is not a dual codeword. This requires us to modify the above idea. In general terms, we will find suitable elements of the set E_1 such that Eq. (2.7) yields linear relations between the entries of the form $\mathrm{tr}(e_i v_j c_j)$. The coefficients of these relations form the rows of the matrix M_j.

2.3.3 The matrices M_j

In this section we will construct the matrices M_j and the vector z_j, and also prove the full rank condition. Rather than writing the expressions at this point in the text, We proceed in stages, by deriving p_1 linear relations involving components of the vectors on both sides of (2.9). (the notation is rather complicated and would not be intuitive; if desired, the reader may nevertheless consult Sec. 2.3.4, particularly, Eq.(2.24)).

2.3.3.1 The first $p_1 - s - 1$ relations

Proposition 5. *For all* $0 \leqslant u \leqslant s-1$ *and* $0 \leqslant q \leqslant \frac{p_1-1}{s} - 2$, *the vector*

$$\left(\alpha_j^s \operatorname{tr}(\beta^u \alpha_1^{u+qs} v_j c_j) - \operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s} v_j c_j), j = 2, \ldots, n\right) \tag{2.10}$$

is a codeword in a GRS code of length $n-1$ *and dimension* $s+k-1$.

Proof. Let us write (2.7) for e_i of the form $e_i = \beta^u \alpha_1^{u+qs}$:

$$\operatorname{tr}(\beta^u \alpha_1^{u+qs+t} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^t \operatorname{tr}(\beta^u \alpha_1^{u+qs} v_j c_j), \quad t = 0, 1, \ldots, n-k-1.$$

(See also (2.5).) Writing this as

$$\operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s+t-s} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^{s+t-s} \operatorname{tr}(\beta^u \alpha_1^{u+qs} v_j c_j), \quad t = 0, 1, \ldots, n-k-1,$$

and performing the change of variable $(t-s) \mapsto t$, we obtain the relation

$$\operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s+t} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^{s+t} \operatorname{tr}(\beta^u \alpha_1^{u+qs} v_j c_j), \tag{2.11}$$

$$t = -s, -s+1, \ldots, -s+n-k-1.$$

On the other hand, substituting $e_i = \beta^u \alpha_1^{u+(q+1)s}$ into (2.7), we obtain

$$\operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s+t} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^{t} \operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s} v_j c_j), \tag{2.12}$$

$$t = 0, 1, \ldots, n-k-1.$$

Note that the left-hand sides of (2.11) and (2.12) conicide for $t = 0, 1, \ldots, n-k-s-1$, and thus so do the right-hand sides. We obtain

$$\sum_{j=2}^{n} \alpha_j^{s+t} \operatorname{tr}(\beta^u \alpha_1^{u+qs} v_j c_j) = \sum_{j=2}^{n} \alpha_j^{t} \operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s} v_j c_j)$$

or

$$\sum_{j=2}^{n} \alpha_j^{t} \big(\alpha_j^{s} \operatorname{tr}(\beta^u \alpha_1^{u+qs} v_j c_j) - \operatorname{tr}(\beta^u \alpha_1^{u+(q+1)s} v_j c_j) \big) = 0,$$

for $t = 0, 1, \ldots, n-k-s-1$. On account of Proposition 1 this implies the claim about the GRS code; moreover, since there are $n-k-s$ independent parity-check equations, the dimension of this code is $(n-1)-(n-k-s) = s+k-1$. □

We note that the components of the vector (2.10) are formed as linear combinations of the elements $\operatorname{tr}(e_i v_j c_j)$, and so this gives us $p_1 - s - 1$ vectors z_j.

2.3.3.2 One more relation

Proposition 6. *The vector*

$$\Big(\sum_{u=0}^{s-1}\alpha_j^{s-u}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j)-\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big),j=2,\ldots,n\Big) \tag{2.13}$$

is a codeword in a GRS code of length $n-1$ and dimension $s+k-1$.

Proof. Going back to (2.7), take $e_i=\beta^u\alpha_1^{u+p_1-s-1}$ for $u=0,1,\ldots,s-1$. We obtain the relation

$$\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1+t}v_1c_1)=-\sum_{j=2}^{n}\alpha_j^t\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j),\quad t=0,1,\ldots,n-k-1.$$

Changing the variable $(t+u-s)\mapsto t$ in the above equation, we obtain that for every $u=0,1,\ldots,s-1$,

$$\operatorname{tr}(\beta^u\alpha_1^{p_1-1+t}v_1c_1)=-\sum_{j=2}^{n}\alpha_j^{t-u+s}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j), \tag{2.14}$$

$$t=u-s,u-s+1,\ldots,u-s+n-k-1.$$

Since

$$\bigcap_{u=0}^{s-1}\{u-s,u-s+1,\ldots,u-s+n-k-1\}=\{-1,0,1,\ldots,n-k-s-1\}, \tag{2.15}$$

we have

$$\operatorname{tr}(\beta^u\alpha_1^{p_1-1+t}v_1c_1) = -\sum_{j=2}^{n}\alpha_j^{t-u+s}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j),$$

$$-1 \leqslant t \leqslant n-k-s-1, \quad 0 \leqslant u \leqslant s-1.$$

Taking the cue from (2.15), let us sum these equations on $u = 0, 1, \dots, s-1$, and we obtain

$$\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1+t}v_1c_1\Big) = -\sum_{j=2}^{n}\sum_{u=0}^{s-1}\alpha_j^{t-u+s}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j), \tag{2.16}$$

$$-1 \leqslant t \leqslant n-k-s-1.$$

Turning to (2.5) again, let us substitute the element $\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}$ into (2.7):

$$\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1+t}v_1c_1\Big) = -\sum_{j=2}^{n}\alpha_j^t\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big), \ 0 \leqslant t \leqslant n-k-1. \tag{2.17}$$

From (2.16) and (2.17) we deduce the equality

$$\sum_{j=2}^{n}\sum_{u=0}^{s-1}\alpha_j^{t-u+s}\operatorname{tr}\Big(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j\Big) = \sum_{j=2}^{n}\alpha_j^t\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big),$$

or

$$\sum_{j=2}^{n}\alpha_j^t\Big(\sum_{u=0}^{s-1}\alpha_j^{s-u}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j) - \operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big)\Big) = 0$$

for $0 \leqslant t \leqslant n-k-s-1$. By Proposition 1, the proof is complete. □

2.3.3.3 The remaining s relations

Following the plan outlined in Sec. 2.3.2, we have constructed $p_1 - s$ vectors z_j, listed in (2.10) and (2.13). In order to find the remaining s linear combinations of the elements $r_{i,j}$, we develop the idea used in the example in Sec. 2.2.2.

We begin with introducing some notation. Let $f(x)$ be the minimal polynomial of α_1 over F_1. For $h = 0, 1, \ldots, s-1$ define

$$f_h(x) = x^{p_1+h} (\mathrm{mod} f(x)), \tag{2.18}$$

then $\deg f_h < \deg f = p_1$ and $\alpha_1^{p_1+h} = f_h(\alpha_1)$. Let $f_{h,q} \in F_1[x], q = 0, \ldots, (p_1 - 1)/s - 1$ be the (uniquely defined) polynomials such that

(i) $\deg f_{h,q} \leqslant s-1, q = 0, 1, \ldots, \frac{p_1-1}{s} - 2$;

(ii) $\deg f_{h,(p_1-1)/s-1} \leqslant s$;

(iii)

$$f_h(x) = \sum_{q=0}^{(p_1-1)/s-1} x^{qs} f_{h,q}(x). \tag{2.19}$$

Proposition 7. *For every $h = 0, 1, \ldots, s-1$, the vector*

$$\Big(\sum_{q=0}^{(p_1-1)/s-1} \sum_{u=0}^{h} f_{h-u,q}(\alpha_j) \operatorname{tr}(\alpha_1^{u+qs} \beta^u v_j c_j) + \sum_{u=h+1}^{s-1} \alpha_j^{h+1-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j)$$

$$- \alpha_j^{h+1} \operatorname{tr}(\sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1} v_j c_j), j = 2, 3, \ldots, n\Big) \quad (2.20)$$

is contained in a GRS code of length $n-1$ *and dimension* $s+k-1$.

The proof of this proposition is rather long and technical, and is given in Appendix A.1.

Concluding, expressions (2.10), (2.13), and (2.20) yield p_1 linear combinations of the elements $(\operatorname{tr}(e_0 v_j c_j), \operatorname{tr}(e_1 v_j c_j), \ldots, \operatorname{tr}(e_{p_1-1} v_j c_j))$ for every $j \in \{2, 3, \ldots, n\}$. It is these linear combinations that we denote by $z_j = (z_{j,0}, z_{j,1}, \ldots, z_{j,p_1-1})$ in (2.9). We have shown that for every $i \in \{0, 1, \ldots, p_1 - 1\}$, the vector $(z_{2,i}, z_{3,i}, \ldots, z_{n,i})$ is contained in an MDS code of length $n-1$ and dimension $s+k-1$. The next subsection treats the remaining part of the assumptions in Proposition 4 above.

2.3.4 The matrices M_j are invertible

The object of this section is to show that the mapping

$$(\operatorname{tr}(e_0 v_j c_j), \operatorname{tr}(e_1 v_j c_j), \ldots, \operatorname{tr}(e_{p_1-1} v_j c_j)) \mapsto z_j = (z_{j,0}, z_{j,1}, \ldots, z_{j,p_1-1})$$

is invertible. In other words, we will show that $\operatorname{rank}(M_j) = p_1$ for all j. Let us first simplify the notation. Recall the set $E_1 = \{e_0, e_1, \ldots, e_{p_1-1}\}$ in (2.5) and let us order its elements in the order of increase of the powers of α_1 :

$$e_{u+qs} := \beta^u \alpha_1^{u+qs} \quad \text{for } u = 0, 1, \ldots, s-1 \text{ and } q = 0, 1, \ldots, \tfrac{p_1-1}{s} - 1,$$

$$e_{p_1-1} := \sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1}.$$

Using the notation $r_{i,j} = \mathrm{tr}(e_i v_j c_j)$ introduced above, the vectors in (2.10) can be written as

$$(\alpha_j^s r_{u+qs,j} - r_{u+qs+s,j}, j = 2, \ldots, n)$$

for $0 \leqslant u \leqslant s-1$ and $0 \leqslant q \leqslant \frac{p_1-1}{s} - 2$, or, writing $i = u + qs$, as

$$(\alpha_j^s r_{i,j} - r_{i+s,j}, j = 2, \ldots, n) \tag{2.21}$$

for $0 \leqslant i \leqslant p_1 - s - 2$. Similarly, the vector in (2.13) can be written as

$$\Big(\sum_{u=0}^{s-1} \alpha_j^{s-u} r_{u+p_1-s-1,j} - r_{p_1-1,j}, j = 2, \ldots, n \Big), \tag{2.22}$$

and the vectors in (2.20) can be written as

$$\Big(\sum_{q=0}^{(p_1-1)/s-1} \sum_{u=0}^{h} f_{h-u,q}(\alpha_j) r_{u+qs,j} + \sum_{u=h+1}^{s-1} \alpha_j^{h+1-u+s} r_{u+p_1-s-1,j} - \alpha_j^{h+1} r_{p_1-1,j}, j = 2, \ldots, n \Big), \tag{2.23}$$

for $0 \leqslant h \leqslant s-1$. For a fixed value of j, the entries in (2.21)–(2.23) form the vector $z_j = (z_{j,0}, z_{j,1}, \ldots, z_{j,p_1-1})$, and we list its coordinates according to the chosen order:

$$\left.\begin{aligned} z_{j,i} &:= \alpha_j^s r_{i,j} - r_{i+s,j} \quad \text{for } 0 \leqslant i \leqslant p_1 - s - 2, \\ z_{j,p_1-s-1} &:= \sum_{u=0}^{s-1} \alpha_j^{s-u} r_{u+p_1-s-1,j} - r_{p_1-1,j}, \\ z_{j,p_1-s+h} &:= \sum_{u=0}^{h} \sum_{q=0}^{(p_1-1)/s-1} f_{h-u,q}(\alpha_j) r_{u+qs,j} \\ &+ \sum_{u=h+1}^{s-1} \alpha_j^{h+1-u+s} r_{u+p_1-s-1,j} - \alpha_j^{h+1} r_{p_1-1,j} \end{aligned}\right\} \tag{2.24}$$

for $0 \leqslant h \leqslant s-1$. Our objective is to show that the linear mapping

$$(r_{0,j}, r_{1,j}, \ldots, r_{p_1-1,j}) \overset{M_j}{\to} (z_{j,0}, z_{j,1}, \ldots, z_{j,p_1-1})$$

is invertible. This will follow once we show that its kernel is trivial, i.e., that if $(z_{j,0}, z_{j,1}, \ldots, z_{j,p_1-1})$ is an all-zeros vector, then so is $(r_{0,j}, r_{1,j}, \ldots, r_{p_1-1,j})$. If $z_{j,i} = \alpha_j^s r_{i,j} - r_{i+s,j} = 0$ for $0 \leqslant i \leqslant p_1 - s - 2$, then

$$r_{u+qs,j} = \alpha_j^s r_{u+(q-1)s,j} = \cdots = \alpha_j^{qs} r_{u,j} \quad \text{for } 0 \leqslant u \leqslant s-1 \text{ and } 1 \leqslant q \leqslant \frac{p_1-1}{s} - 1. \tag{2.25}$$

Using (2.25) in the expression for $z_{j,p_1-s+h}, 0 \leqslant h \leqslant s-1$, we obtain the following s relations:

$$z_{j,p_1-s+h} = \sum_{u=0}^{h} \sum_{q=0}^{(p_1-1)/s-1} f_{h-u,q}(\alpha_j) \alpha_j^{qs} r_{u,j} + \sum_{u=h+1}^{s-1} \alpha_j^{h+1-u+s} \alpha_j^{p_1-s-1} r_{u,j} - \alpha_j^{h+1} r_{p_1-1,j}$$

$$= \sum_{u=0}^{h} f_{h-u}(\alpha_j) r_{u,j} + \sum_{u=h+1}^{s-1} \alpha_j^{p_1+h-u} r_{u,j} - \alpha_j^{h+1} r_{p_1-1,j}, \tag{2.26}$$

where the second equality follows from (2.19). Using (2.25) in the expression for z_{j,p_1-s-1}, we obtain

$$z_{j,p_1-s-1} = \sum_{u=0}^{s-1} \alpha_j^{s-u} \alpha_j^{p_1-s-1} r_{u,j} - r_{p_1-1,j} = \sum_{u=0}^{s-1} \alpha_j^{p_1-u-1} r_{u,j} - r_{p_1-1,j}. \tag{2.27}$$

Since we assumed that the z-vector is zero, coordinates $z_{p_1-u}, u = s+1, s, \ldots, 1$ that appear in (2.26), (2.27) are zero. Writing these conditions in matrix form using the above order, we obtain relation (2.28). We aim to show that the matrix on the left-hand side is invertible.

$$\begin{bmatrix} \alpha_j^{p_1-1} & \alpha_j^{p_1-2} & \alpha_j^{p_1-3} & \ldots & \alpha_j^{p_1-s} & -1 \\ f_0(\alpha_j) & \alpha_j^{p_1-1} & \alpha_j^{p_1-2} & \ldots & \alpha_j^{p_1-s+1} & -\alpha_j \\ f_1(\alpha_j) & f_0(\alpha_j) & \alpha_j^{p_1-1} & \ldots & \alpha_j^{p_1-s+2} & -\alpha_j^2 \\ f_2(\alpha_j) & f_1(\alpha_j) & f_0(\alpha_j) & \ldots & \alpha_j^{p_1-s+3} & -\alpha_j^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_{s-1}(\alpha_j) & f_{s-2}(\alpha_j) & f_{s-3}(\alpha_j) & \ldots & f_0(\alpha_j) & -\alpha_j^s \end{bmatrix} \begin{bmatrix} r_{0,j} \\ r_{1,j} \\ r_{2,j} \\ \vdots \\ r_{s-1,j} \\ r_{p_1-1,j} \end{bmatrix} = 0, \tag{2.28}$$

Recall that $f(x)$ is the minimal polynomial of α_1 and from (2.19), $f(x) + f_0(x) = x^{p_1}$. Since $f(x)$ is irreducible over F_1 and $\alpha_j \in F_1$, we have $f(\alpha_j) \neq 0$ for all $j = 2, \ldots, n$.

Multiplying the first row of the matrix in (2.28) by α_j and then subtracting

the second row from the first row, we obtain

$$\begin{bmatrix} f(\alpha_j) & 0 & 0 & \dots & 0 & 0 \\ f_0(\alpha_j) & \alpha_j^{p_1-1} & \alpha_j^{p_1-2} & \dots & \alpha_j^{p_1-s+1} & -\alpha_j \\ f_1(\alpha_j) & f_0(\alpha_j) & \alpha_j^{p_1-1} & \dots & \alpha_j^{p_1-s+2} & -\alpha_j^2 \\ f_2(\alpha_j) & f_1(\alpha_j) & f_0(\alpha_j) & \dots & \alpha_j^{p_1-s+3} & -\alpha_j^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_{s-1}(\alpha_j) & f_{s-2}(\alpha_j) & f_{s-3}(\alpha_j) & \dots & f_0(\alpha_j) & -\alpha_j^s \end{bmatrix}.$$

Since $f(\alpha_j) \neq 0$, we can use elementary row operations to erase the first column, obtaining

$$\begin{bmatrix} f(\alpha_j) & 0 & 0 & \dots & 0 & 0 \\ 0 & \alpha_j^{p_1-1} & \alpha_j^{p_1-2} & \dots & \alpha_j^{p_1-s+1} & -\alpha_j \\ 0 & f_0(\alpha_j) & \alpha_j^{p_1-1} & \dots & \alpha_j^{p_1-s+2} & -\alpha_j^2 \\ 0 & f_1(\alpha_j) & f_0(\alpha_j) & \dots & \alpha_j^{p_1-s+3} & -\alpha_j^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & f_{s-2}(\alpha_j) & f_{s-3}(\alpha_j) & \dots & f_0(\alpha_j) & -\alpha_j^s \end{bmatrix}.$$

Proceeding analogously, let us multiply the second row of this matrix by α_j and

then subtract the third row from the second one to obtain

$$\begin{bmatrix} f(\alpha_j) & 0 & 0 & \dots & 0 & 0 \\ 0 & f(\alpha_j) & 0 & \dots & 0 & 0 \\ 0 & f_0(\alpha_j) & \alpha_j^{p_1-1} & \dots & \alpha_j^{p_1-s+2} & -\alpha_j^2 \\ 0 & f_1(\alpha_j) & f_0(\alpha_j) & \dots & \alpha_j^{p_1-s+3} & -\alpha_j^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & f_{s-2}(\alpha_j) & f_{s-3}(\alpha_j) & \dots & f_0(\alpha_j) & -\alpha_j^s \end{bmatrix}.$$

As above, we can eliminate all the nonzeros in the second column except for $f(\alpha_j)$, and so on. In the end we obtain the matrix $\mathrm{diag}(f(\alpha_j), \dots, f(\alpha_j), -\alpha_j^s)$ with nonzero diagonal. This proves that the matrix in (2.28) is invertible. Therefore, $r_{0,j} = r_{1,j} = \dots = r_{s-1,j} = r_{p_1-1,j} = 0$. Combining this with (2.25), we conclude that $r_{i,j} = 0$ for all $0 \leqslant i \leqslant p_1 - 1$. This proves that the matrices $M_j, j = 2, \dots, n$ in (2.9) are invertible, providing the last missing element to the justification of the repair scheme with optimal error correction.

2.4 A family of optimal-access RS codes

In this section, we construct a new family of RS codes that is similar to the construction in [77] but affords repair with optimal access.

The input-output cost of node repair for the RS codes of [77] was analyzed in [45] for $d = n - 1$. According to (1.2), in this case the minimum access cost per helper node equals $\frac{l}{n-k}$. The authors of [45] showed that it is possible to adjust

the repair scheme so that the access cost is $(1 + \frac{n-k-1}{p_i})\frac{l}{n-k}$, i.e., at most twice the optimal value. However, more is true: namely, it turns out that any *fixed* node in the construction of [77] (Def. 5) can be repaired with optimal access. This observation, which is the starting point of the new construction, is based on the fact that it is possible to construct a basis of the field $\mathbb{K}$ over the base field that reduces the access cost of the repair of the chosen node. If the option of choosing the basis for each erased node were available, we could use the arguments in Sec. 2.2.3 to perform repair with optimal access. The difficulty arises because this would entail rewriting the storage contents, which should be avoided. To address this issue, we construct the code over a field that contains n elements β_i instead of a single element β, and this supports efficient repair of any single failed node. This idea is developed below.

2.4.1 New construction

Consider the following sequence of algebraic extensions of $\mathbb{F}_p$: let $K_0 = \mathbb{F}_p$ and for $i = 1, \ldots, n$ let

$$F_i = K_{i-1}(\alpha_i), K_i = F_i(\beta_i), \tag{2.29}$$

where α_i is an algebraic element of degree p_i over $\mathbb{F}_p$ and β_i is an element of degree $s = d - k + 1$ over F_i. In the end we obtain the field

$$\mathbb{K} := K_n = \mathbb{F}_p(\alpha_1, \ldots, \alpha_n, \beta_1, \ldots \beta_n). \tag{2.30}$$

We still assume that $p_1, \ldots, p_n$ are distinct primes satisfying the condition $p_i \equiv 1 \bmod s$ for all $i = 1, \ldots, n$. Consider the code $\mathcal{C} := RS_{\mathbb{K}}(n, k, \Omega)$, where as before, the set of evaluation points is given by $\Omega = \{\alpha_1, \ldots, \alpha_n\}$. We will show that the code $\mathcal{C}$ affords optimal-access repair.

The repair scheme follows the general approach of [28] and its implementation in [77]. Let $c = (c_1, \ldots, c_n) \in \mathcal{C}$ be a codeword. Suppose that the node i has failed (coordinate c_i is erased), and we would like to repair it from a set of helper nodes $\mathcal{R} \subseteq \{1, \ldots, n\} \backslash \{i\}$ with $|\mathcal{R}| = d$. Let

$$h(x) = \prod_{j \in \{1,\ldots,n\} \backslash (\mathcal{R} \cup \{i\})} (x - \alpha_j).$$

Clearly, we have $\deg(x^t h(x)) < n-k$ for $t = 0, \ldots, s-1$. Therefore, for some nonzero vector $v = (v_1, \ldots, v_n)$, we have $(v_1 \alpha_1^t h(\alpha_1), \ldots, v_n \alpha_n^t h(\alpha_n)) \in \mathcal{C}^{\perp}$ for $t = 0, \ldots, s-1$, where $\mathcal{C}^{\perp} = GRS_{\mathbb{K}}(n, k, v, \Omega)$. In other words, we have

$$v_i \alpha_i^t h(\alpha_i) c_i = -\sum_{\substack{j=1 \\ j \neq i}}^{n} v_j \alpha_j^t h(\alpha_j) c_j, \ t = 0, \ldots, s-1. \tag{2.31}$$

The repair scheme in [77] as well as in this chapter relies on this set of s dual codewords to recover the value of c_i.

Remark 1. The dual codewords $x^t h(x)$ have zero values in the complement of the set $\hat{\mathcal{R}} := \mathcal{R} \cup \{i\}$. In other words, they are contained in the shortened code $(\mathcal{C}^{\perp})^{\hat{\mathcal{R}}}$ of the dual code. Thinking dually, we can start with the code $\mathcal{C}^{\perp}$ and construct a repair scheme for its coordinates based on the punctured code $\mathcal{C}_{\hat{\mathcal{R}}}$ (coordinate projection of

$\mathcal{C}$ on $\hat{\mathcal{R}}$). This approach is equivalent to the scheme used in [77] and in this chapter because $((\mathcal{C}^{\perp})^{\hat{\mathcal{R}}})^{\perp} \cong C_{\hat{\mathcal{R}}}$.

Let us establish a few simple properties of the tower of fields defined above in (2.29), (2.30).

Lemma 8. *The extension degrees in the field tower* $\mathbb{F}_p = K_0 \subset \cdots \subset K_i \subset \cdots \subset K_n = \mathbb{K}$ *are as follows:*

$$[K_i : \mathbb{F}_p] = s^i \prod_{j=1}^{i} p_j, i = 1, \ldots, n$$
$$[\mathbb{K} : \mathbb{F}_p] = l := s^n \prod_{i=1}^{n} p_i.$$

Proof. The proof is obvious from the definition: for each i we adjoin two elements α_i, β_i to K_{i-1}, and their degrees over K_{i-1} are coprime, so they contribute sp_i to the result. □

We will use an explicit form of the basis of $\mathbb{K}$ over $\mathbb{F}_p$. For $m = 0, \ldots, l-1$, let us write

$$m = (m_n, m_{n-1}, \ldots, m_1, \bar{m}_n, \bar{m}_{n-1}, \ldots, \bar{m}_1) \tag{2.32}$$

where $m_i = 0, \ldots, p_i - 1$ and $\bar{m}_i = 0, \ldots, s-1$ for $i = 1, \ldots, n$.

Lemma 9. *Let*

$$A = \{a_m := \prod_{i=1}^{n} \alpha_i^{m_i} \prod_{j=1}^{n} \beta_j^{\bar{m}_j} \mid m_i = 0, \ldots, p_i - 1, \bar{m}_j = 0, \ldots, s-1; m = 0, 1, \ldots, l-1\}.$$

Then A is a basis for $\mathbb{K}$ over $\mathbb{F}_p$.

Proof. By co-primality, for $i = 1, \ldots, n$ we have $\deg_{K_{i-1}}(\alpha_i) = p_i$, and by construction, we have $\deg_{F_i}(\beta_i) = s$. Thus, the elements $a_m, m = 0, \ldots, l-1$ are linearly independent over $\mathbb{F}_p$. □

Lemma 10. *For $m = 0, \ldots, l-1$ let $\mathcal{J} = \{j \in [n] : (\bar{m}_j, m_j) = (s-1, p_j - 1)\}$ and let*

$$b_m = \prod_{i=1}^{n} \alpha_i^{m_i} \cdot \prod_{j \in \mathcal{J}} \Big(\sum_{u=0}^{s-1} \beta_j^u \Big) \cdot \prod_{j \notin \mathcal{J}} \beta_j^{\bar{m}_j}.$$

Then the set $B := \{b_m \mid m = 0, \ldots, l-1\}$ is a basis of $\mathbb{K}$ over $\mathbb{F}_p$.

Furthermore, for $i = 1, \ldots, n$, let $A_i = \{a_m \in A \mid (m_i, \bar{m}_i) = (0,0)\}$ and $B_i = \{b_m \in B \mid (m_i, \bar{m}_i) = (0,0)\}$, then

$$\mathrm{Span}_{\mathbb{F}_p} A_i = \mathrm{Span}_{\mathbb{F}_p} B_i.$$

Proof. Since $|B| = l$, to prove that B is a basis it suffices to show that the elements a_m can be expressed as linear combinations of the elements in B. Let $\mathcal{J} \subset [n]$ and let $A(\mathcal{J}) = \{a_m \in A : (\bar{m}_j, m_j) = (s-1, p_j - 1), j \in \mathcal{J}; (\bar{m}_j, m_j) \neq (s-1, p_j - 1), j \notin \mathcal{J}\}$. We argue by induction on $|\mathcal{J}|$. If m is such that $\mathcal{J} = \varnothing$, then $a_m \in B$, and there is nothing to prove. Now assume that for all $\mathcal{J} \subset [n], |\mathcal{J}| \leqslant J - 1$ the elements a_m are linearly generated by the elements in B, and let m be such that $|\mathcal{J}| = J$. We have

$$a_m = \prod_{i=1}^{n} \alpha_i^{m_i} \prod_{j \notin \mathcal{J}} \beta_j^{\bar{m}_j} \prod_{j \in \mathcal{J}} \beta_j^{s-1}$$

and

$$b_m = \prod_{i=1}^{n} \alpha_i^{m_i} \prod_{j\notin\mathcal{J}} \beta_j^{\bar{m}_j} \prod_{j\in\mathcal{J}} \sum_{u=0}^{s-1} \beta_j^u = \prod_{i=1}^{n} \alpha_i^{m_i} \Big(\prod_{j\notin\mathcal{J}} \beta_j^{\bar{m}_j} \Big) \Big(\sum_{t_1,\dots,t_J=0}^{s-1} \prod_{u=1}^{J} \beta_{j_u}^{t_u} \Big).$$

Multiplying out the sums on right-hand side, we note that the term with all $t_i = s-1$ equals a_m, while the remaining terms contain fewer than J factors of the form $\alpha_{j_u}^{p_{j_u}-1}\beta_{j_u}^{s-1}$. Each of such terms is contained in some $A(\mathcal{J})$ with $|\mathcal{J}| \leqslant J-1$, and is linearly generated by the elements b_m by the induction hypothesis. This implies that a_m is also expressible as a linear combination of the elements in B.

To prove the second claim, note that $\mathrm{Span}_{\mathbb{F}_p} A_i \supseteq \mathrm{Span}_{\mathbb{F}_p} B_i$. Therefore, to show that $\mathrm{Span}_{\mathbb{F}_p} A_i = \mathrm{Span}_{\mathbb{F}_p} B_i$, it suffices to show that for any $a = 0, \dots, n-1$ and any $\mathcal{J} \subseteq \{1,\dots,n\}\backslash\{i\}$, the set $A_i(\mathcal{J})$ can be generated linearly by the set B_i. This proof amounts essentially to the same calculation as above, and will be omitted. □

The role of the basis (b_m) is to eliminate as many terms on the right-hand side of (2.31) as possible. To repair the node c_i we use the dual basis (b_m^*) of (b_m), writing

$$c_i = v_i^{-1} \sum_{m=0}^{l-1} c_{i,m} b_m^*. \tag{2.33}$$

Below $\mathrm{tr} = \mathrm{tr}_{\mathbb{K}/\mathbb{F}_p}$ denotes the absolute trace.

Lemmas 8 and 3 immediately imply the following.

Proposition 11. *For $i = 1, \dots, n$, there exists vector space S_i over K_{i-1} such that $\dim_{K_{i-1}} S_i = p_i$ and $S_i + S_i\alpha_i + \cdots + S_i\alpha_i^{s-1} = K_i$. Furthermore, a basis for S_i over*

K_{i-1} is given by

$$E_i := \{\beta_i^u \alpha_i^{u+qs} \mid u = 0, \ldots, s-1; q = 0, \ldots, \tfrac{p_i-1}{s} - 1\} \bigcup \Big\{\alpha_i^{p_i-1} \sum_{u=0}^{s-1} \beta_i^u\Big\}.$$

We continue with the description of the repair scheme where we left in (2.31). As a remark, below we write the scheme over $\mathbb{F}_p$ rather than over its extensions (the latter approach was chosen in [77]). Multiplying both sides of (2.31) by $\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}}$, where $e_{i'} \in E_{i'}$ and $t_{j'} = 0, \ldots, s-1$, and evaluating the trace, we obtain

$$\begin{aligned}
\operatorname{tr}\Big(\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} v_i \alpha_i^t h(\alpha_i) c_i\Big) &= -\operatorname{tr}\Big(\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} \sum_{j \neq i}^n v_j \alpha_j^t h(\alpha_j) c_j\Big) \\
&= -\sum_{j \neq i}^n \operatorname{tr}\Big(\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} v_j \alpha_j^t h(\alpha_j) c_j\Big) \\
&= -\sum_{j \in \mathcal{R}} \operatorname{tr}\Big(\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} v_j \alpha_j^t h(\alpha_j) c_j\Big). \qquad (2.34)
\end{aligned}$$

On account of Proposition 11 and the fact that $v_i h(\alpha_i) \neq 0$, the set

$$\Big\{\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} v_i \alpha_i^t h(\alpha_i)\Big\}, \qquad (2.35)$$

where $e_{i'} \in E_{i'}, i' \in [n]; t = 0, \ldots, s-1; t_{j'} = 0, \ldots, s-1, j' \in [n] \backslash \{i\}$, is a basis of $\mathbb{K}$ over $\mathbb{F}_p$. Therefore, we can recover c_i once we know the right-hand side of (2.34).

For $j \in \mathcal{R}$, from (2.33) we have

$$\operatorname{tr}(\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} v_j \alpha_j^t h(\alpha_j) c_j) = \operatorname{tr}\Big(\prod_{i'=1}^n e_{i'} \prod_{j' \neq i}^n \alpha_{j'}^{t_{j'}} \alpha_j^t h(\alpha_j) \sum_{m=0}^{l-1} c_{j,m} b_m^*\Big)$$

$$= \sum_{m=0}^{l-1} \mathrm{tr}\Big(\prod_{i'=1}^{n} e_{i'} \prod_{j'\neq i}^{n} \alpha_{j'}^{t_{j'}} \alpha_j^t h(\alpha_j) b_m^*\Big) c_{j,m}. \qquad (2.36)$$

From (2.36), we see that in order to recover c_i we need to access only those symbols $c_{j,m}$ for which

$$\mathrm{tr}(\prod_{i'=1}^{n} e_{i'} \prod_{j'\neq i}^{n} \alpha_{j'}^{t_{j'}} \alpha_j^t h(\alpha_j) b_m^*) \neq 0.$$

Now, the element $\prod_{i'\neq i}^{n} e_{i'} \prod_{j'\neq i}^{n} \alpha_{j'}^{t_{j'}} \alpha_j^t h(\alpha_j)$ does not include α_i, β_i, and thus it can be written as an $\mathbb{F}_p$-linear combination of the elements in the set A_i. By Lemma 10, it can further be expressed as an $\mathbb{F}_p$-linear combination of the elements in the set B_i. Therefore, the elements $\prod_{i'=1}^{n} e_{i'} \prod_{j'\neq i}^{n} \alpha_{j'}^{t_{j'}} \alpha_j^t h(\alpha_j)$ for $e_{i'} \in E_{i'}$ and $t_{j'} = 0, \ldots, s-1$ can be linearly generated over $\mathbb{F}_p$ by the set

$$\bigcup_{e_i \in E_i} e_i B_i \subseteq B.$$

Since B and B^* are dual bases,

$$\mathrm{tr}\Big(\prod_{i'=1}^{n} e_{i'} \prod_{j'\neq i}^{n} \alpha_{j'}^{t_{j'}} \alpha_j^t h(\alpha_j) b_m^*\Big) \neq 0$$

if and only if $b_m \in \bigcup_{e_i \in E_i} e_i B_i$. It follows that to calculate the left hand side of (2.34), we need to access $\sum_{e_i \in E_i} |e_i B_i| = p_i l/sp_i = l/s$ symbols on each helper node $j \in \mathcal{R}$, which implies that the node c_i affords optimal-access repair.

In conclusion, we note that the repair scheme of each of the nodes i relies on its own element β_i. Looking back at the construction of [77], Sec. 2.3 above, it contains one such β. Thus, these codes can be furnished with a repair scheme that

has the optimal access property for any one (fixed) node in the encoding; see also the discussion at the end of Sec. 2.2.3.

2.4.2 Error correction with optimal access

In this section we present a repair scheme of the RS codes defined in the beginning of Sec. 2.4.1 that supports both the optimal access and optimal error correction properties. The scheme relies on a combination of ideas of Sections 2.4.1 and 2.3. A full presentation of the proof would require us to repeat the arguments in Sec. 2.3.3; we shall instead confine ourselves to pointing to the similarity of the starting point and argue that once this is recognized, the remaining part is reproduced directly following the proof in Sec. 2.3.3.

Let us modify the construction of RS codes of Sec. 2.4.1 as follows. Let us assume that the number of helper nodes is d. We will construct our RS code over the symbol field $\mathbb{K} = \mathbb{F}_p(\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n)$ (2.30), where as before, $\deg_{K_{i-1}}(\alpha_i) = p_i$ but $\deg_{F_i}(\beta_i) = s := d - 2e - k + 1$. Define the code $\mathcal{C} := RS_{\mathbb{K}}(n, k, \Omega)$, where $\Omega = \{\alpha_1, \ldots, \alpha_n\}$.

Without loss of generality suppose that the failed node is the first one and let $\mathcal{R} \subseteq \{2, 3, \ldots, n\}$ with $|\mathcal{R}| = d, 2e + k \leqslant d \leqslant n - 1$ be the subset of helper nodes. Consider a basis of $\mathbb{K}$ over $\mathbb{F}_p$ given by $\bigcup_{t=0}^{s-1} \alpha_1^t \Lambda$, where

$$\Lambda = \Big\{ \prod_{i=1}^{n} e_i \prod_{j=2}^{n} \alpha_j^{t_j} \mid e_i \in E_i, i \in [n]; t_j = 0, \ldots, s-1, j \in [n] \backslash \{1\} \Big\}.$$

That this is a basis is apparent from (2.35).

Next, note that $(v_1\alpha_1^t, \ldots, v_n\alpha_n^t) \in \mathcal{C}^\perp$ for some $v = (v_1, \ldots, v_n) \in (\mathbb{K}^*)^n$ and for $t = 0, \ldots, n-k-1$. Therefore, for every $\lambda \in \Lambda$ we have

$$\lambda v_1 \alpha_1^t c_1 = -\sum_{j=2}^{n} \lambda v_j \alpha_j^t c_j, \quad t = 0, \ldots, n-k-1.$$

Let $G_1 := \mathbb{F}_p(\alpha_2, \alpha_3, \ldots, \alpha_n)$. Evaluating the trace $\mathrm{tr}_{\mathbb{K}/G_1}$ on both sides of the above equation, we obtain

$$\mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_1 \alpha_1^t c_1) = -\sum_{j=2}^{n} \alpha_j^t \,\mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j), \quad t = 0, \ldots, n-k-1. \tag{2.37}$$

The repair scheme for the code $\mathcal{C}$ is based on (2.37) in exactly the same way as the repair scheme of Proposition 4 is based on (2.7). Namely, suppose that there are invertible linear transformations that map the vectors $(\mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j), \lambda \in \Lambda), j = 2, 3, \ldots, n$ to codevectors in an MDS code of length $n-1$ and dimension $s+k-1$. Then it is possible to correct e errors in the information collected from the helper nodes upon puncturing of this code to any d coordinates in the same way as is done in Proposition 4. Thus, the main step is to prove existence of such transformations. Here we observe that the terms involved in (2.37) are formed of e_1 times the remaining factors in λ. The element e_1 plays the same role as e_i in (2.7), and the multiplier in front of it in λ does not affect the proof. For this reason, the required proof closely follows the proof in Sec. 2.3.3, and we do not repeat it here.

Thus, the vectors $(\mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j), \lambda \in \Lambda), j \in \mathcal{R}$ suffice to recover the value of the failed node. We argue that these values can be calculated by accessing the smallest

possible number of symbols on the helper nodes, and thus support the claim of optimal access. Let $B = (b_m)$ be the basis of $\mathbb{K}$ over $\mathbb{F}_p$ defined in Lemma 10, let $B^* = (b_m^*)$ be its dual basis, and let $B_1 = \{b_m \in b | (m_1, \bar{m}_1) = (0,0)\}$. From (2.33), for every $\lambda \in \Lambda$ and all $j = 2, 3, \ldots, n$ we have the equality

$$\mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j) = \mathrm{tr}_{\mathbb{K}/G_1}\left(\lambda \sum_{m=0}^{l-1} b_m^*\right) c_{i,m}.$$

Let Γ be a basis for G_1 over $\mathbb{F}_p$. Then from the above equation, for every $\gamma \in \Gamma$ we have

$$\mathrm{tr}_{G_1/\mathbb{F}_p}(\gamma\, \mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j)) = \mathrm{tr}_{G_1/\mathbb{F}_p}\left(\gamma\, \mathrm{tr}_{\mathbb{K}/G_1}\left(\lambda \sum_{m=0}^{l-1} b_m^*\right)\right) c_{i,m}.$$

Since $\gamma \in G_1$ and $\mathrm{tr}_{G_1/\mathbb{F}_p} \circ \mathrm{tr}_{\mathbb{K}/G_1} = \mathrm{tr}_{\mathbb{K}/\mathbb{F}_p}$, it follows that

$$\mathrm{tr}_{\mathbb{K}/\mathbb{F}_p}(\gamma\lambda v_j c_j) = \mathrm{tr}_{\mathbb{K}/\mathbb{F}_p}\left(\gamma\lambda \sum_{m=0}^{l-1} b_m^*\right) c_{i,m}. \tag{2.38}$$

Note that the elements $\gamma\lambda = \gamma \prod_{i=1}^{n} e_i \prod_{j=2}^{n} \alpha_j^{t_j}$ can be written as $\mathbb{F}_p$-linear combinations of the elements in the set $\bigcup_{e_1 \in E_1} e_1 B_1 \subseteq B$. By the duality of B and B^*, the number of symbols that each helper node accesses to calculate the left hand side of (2.38) equals $|\bigcup_{e_1 \in E_1} e_1 B_1| = l/s$, which, as remarked in the introduction, is the smallest possible number of symbols. Further, since Γ is a basis of G_1 over $\mathbb{F}_p$, we can recover $\mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j)$ from the set $\{\mathrm{tr}_{\mathbb{K}/\mathbb{F}_p}(\gamma\lambda v_j c_j) \mid \gamma \in \Gamma\}$.

Finally, evaluating the trace $\mathrm{tr}_{G_1/\mathbb{F}_p}$ on both sides of (2.37), we obtain

$$\mathrm{tr}_{\mathbb{K}/\mathbb{F}_p}(\lambda v_1 \alpha_1^t c_1) = -\sum_{j=2}^{n} \mathrm{tr}_{G_1/\mathbb{F}_p}(\alpha_j^t \, \mathrm{tr}_{\mathbb{K}/G_1}(\lambda v_j c_j)), \quad t = 0, \ldots, s-1. \tag{2.39}$$

Since the set $\{\lambda v_1 \alpha_1^t \mid \lambda \in \Lambda; t = 0, \ldots, s-1\}$ forms a basis for $\mathbb{K}$ over $\mathbb{F}_p$, we conclude from (2.39) that we can perform optimal error correction for the code $\mathcal{C}$ with optimal access. As a final remark, the locations of the entries accessed on each helper node depend only on the index of the failed node, and are independent of the index of the helpers.

2.5 Every scalar MSR code affords optimal-access repair

This section is devoted to establishing the claim in the title. We begin with a discussion of repair schemes with a particular property of having constant repair subspaces and use it to show that every MSR code with this property can be repaired with optimal access. In the last part of the section we remove this assumption, establishing the general result, which is stated as follows.

Theorem 12. *Let $\mathcal{C}$ be an (n, k) scalar MDS code over a finite field K of length n such that any single failed node can be optimally repaired from any subset of d helper nodes, $k+1 \leqslant d \leqslant n-1$ with optimal repair bandwidth. Then there exists an explicit procedure that supports optimal-access repair of any single node from any subset of d helpers, $k+1 \leqslant d \leqslant n-1$.*

2.5.1 Constant repair subspaces

Observe that the repair scheme presented above in Sec. 2.4 has the property that for a given index of the failed node i, the procedure for recovering the node contents does not depend on the chosen subset of d helper nodes. Indeed, to repair node i, the scheme accesses symbols $\{c_{j,m} \mid m : b_m \in \bigcup_{e_i \in E_i} e_i B_i\}$ on the node j, i.e., the symbols $c_{j,m}$ with $m = (m_i, \bar{m}_i)$ and

$$(m_i, \bar{m}_i) \in \{(u + qs, u) \mid u = 0, \ldots, s-1; q = 0, \ldots, (p_i - 1)/s - 1\} \cup \{(p_i - 1, s - 1)\}.$$

Clearly the values of m are independent of $j \in \mathcal{R}$. This simplifies the implementation, and therefore represents a desirable property of the scheme. In this section, we generalize this observation and give conditions for it to hold.

Let $\mathcal{C}$ be an (n, k) linear scalar MDS code of length n over finite field K, and let $r = n - k$ be the number of parity nodes. Let F be a subfield of K such that $[K : F] = l$. For a subset $M \subset K$ we write $\dim_F(M)$ to refer to the dimension of the subspace spanned by the elements of M over F. The following result is a starting point of our considerations.

Theorem 13 ([28]). *The code $\mathcal{C}$ has an optimal linear repair scheme over F with repair degree $d = n - 1$ if and only if for every $i = 1, \ldots, n$ there exist l codewords $(c^{\perp}_{t,1}, \ldots, c^{\perp}_{t,n}) \in \mathcal{C}^{\perp}, t = 1, \ldots, l$ such that*

$$\dim_F(c^{\perp}_{1,i}, \ldots, c^{\perp}_{l,i}) = l,$$

$$\sum_{j\neq i}^{n} \dim_F(c_{1,j}^{\perp},\ldots,c_{l,j}^{\perp}) = \frac{(n-1)l}{r}.$$

We go on to define the main object of this section.

Definition 6. *Let $\mathcal{C}$ be a scalar MDS code that has a linear repair scheme for repair of a single node with optimal bandwidth, based on dual codewords $c_1^{\perp},\ldots,c_l^{\perp}$. The scheme is said to have constant repair subspaces if for every $i=1,\ldots,n$ and every $\mathcal{R}\subset[n]\backslash\{i\}, |\mathcal{R}|=d$, the information downloaded from a helper node $c_j, j\in\mathcal{R}$ to repair the failed node c_i does not depend on the index j. Namely, the subspace $\mathcal{S}_j^{(i)} := \mathrm{Span}_F(c_{1,j}^{\perp},\ldots,c_{l,j}^{\perp}), j\in\mathcal{R}$ is independent of the index j, i.e., $\mathcal{S}_j^{(i)}=\mathcal{S}^{(i)}$ for some linear subspace $\mathcal{S}^{(i)}\subseteq K$.*

The notion of constant repair subspaces was mentioned earlier in the literature on general MSR codes, for instance, see [75].

The algorithms below in this section rely on a proposition which we cite from [77].

Proposition 14. *Let $\mathcal{C}$ be an $(n,n-r)$ MDS code and let $[n]=J\cup J^c$, where $J, |J|=r$ is the set of parity coordinates. Let $H=(h_1,\ldots,h_n)$ be a parity-check matrix of $\mathcal{C}$, where h_i denote its columns. The code $\mathcal{C}$ has an optimal linear repair scheme over F with repair degree $d=n-1$ if and only if for each $j\in J^c$ there exist r vectors $a_u\in K^{l/r}, u=1,\ldots,r$ such that*

$$\dim_F(Ah_j) = l, \tag{2.40}$$

$$\dim_F(Ah_i) = \frac{l}{r}, \quad i\in\{1,\ldots,n\}\backslash\{j\}, \tag{2.41}$$

where $A := \mathrm{Diag}(a_1, \ldots, a_r)$ *is an* $l \times r$ *block-diagonal matrix with blocks formed by single columns. Furthermore for every subspace* $\mathcal{A}_u = \mathrm{Span}_F(a_u), u = 1, \ldots, r$ *(the* F*-linear span of the entries of* a_u*) we have*

$$\dim_F(\mathcal{A}_u) = \frac{l}{r}. \tag{2.42}$$

Remark 2. The matrix A in Proposition 14 depends on the matrix H and the choice of J, but we suppress this dependence from the notation for simplicity.

Before presenting the algorithms for finding a basis for optimal-access repair we briefly digress to state some conditions for an optimal linear repair scheme to have constant repair subspaces. First, we rephrase their definition based Proposition 14.

Definition 7. *An optimal linear repair scheme for the code* $\mathcal{C}$ *is said to have constant repair subspaces if for every* $j = 1, \ldots, n$ *there exists a vector* $h \in K^r$ *such that*

$$\mathrm{Span}_F(Ah_i) = \mathrm{Span}_F(Ah)$$

for every $i \in \{1, \ldots, n\} \backslash \{j\}$. *Here the matrix* A *is as in Proposition 14, and it depends on* H *and the particular choice of the information coordinates.*

Proposition 15. *Suppose that* $\mathcal{A}_1 = \mathcal{A}_2 = \cdots = \mathcal{A}_r$ *for each* $i = 1, \ldots, n$, *and that for every* $j \in \{1, \ldots, n\} \backslash \{i\}$ *there exists* $v \in \{1, \ldots, r\}$ *such that* $h_{v,j} \in F$, *then there exists an optimal linear repair scheme for the code* $\mathcal{C}$ *which has constant repair subspaces.*

Proof. Let $\mathcal{V}$ denote any of the (coinciding) repair subspaces. By Proposition 14,

we have $\dim_F(\mathcal{V}) = l/r$. Suppose that J is the subset of parity coordinates, and the matrix H is represented in systematic form. In this case, for every $j \in J^c$, $h_{u,j} \neq 0$ for all $u = 1 \ldots, r$, and we have $\dim_F(\mathcal{V}h_{u,j}) = l/r$. Note that

$$\mathrm{Span}_F(Ah_j) = \sum_{u=1}^{r} \mathcal{A}_u h_{u,j} = \sum_{u=1}^{r} \mathcal{V}h_{u,j}, \quad j \in \{1, \ldots, n\} \backslash \{i\}, \tag{2.43}$$

where the sum on the right is a sum of linear spaces. By Proposition 14, we also have $l/r = \dim_F(Ah_j) = \dim_F(\sum_{u=1}^{r} \mathcal{V}h_{u,j})$. Therefore,

$$\mathcal{V}h_{1,j} = \mathcal{V}h_{2,j} = \cdots = \mathcal{V}h_{r,j}, \quad j \in J^c \backslash \{i\}. \tag{2.44}$$

Since for each $j \neq i$ there exists $v \in \{1, \ldots, r\}$ such that $h_{v,j} \in F$, it follows that $\mathcal{V}h_{v,j} = \mathcal{V}$. On account of (2.43) and (2.44), we have $\mathrm{Span}_F(Ah_j) = \mathcal{V} = \mathrm{Span}_F(A \cdot \mathbf{1})$ for every $j \in \{1, \ldots, n\} \backslash \{i\}$, where $\mathbf{1}$ is the all-ones column vector of length r. By Definition 7 this completes the proof. □

The assumptions of this proposition are satisfied, for instance, for the RS subfamily of [77], which therefore have constant repair subspaces (this observation was previously not stated in published literature).

Proposition 16. *If there exists an optimal linear repair scheme for the code $\mathcal{C}$ which has constant repair subspaces, then $\mathcal{A}_1 = \mathcal{A}_2 = \cdots = \mathcal{A}_r$ for every $j = 1, \ldots, n$.*

Proof. Indeed, since H_J is the identity, for $j \in J$ we have $\mathrm{Span}_F(Ah_j) = \mathcal{A}_t$ for some $t \in \{1, \ldots, r\}$. It follows that $\mathcal{A}_1 = \mathcal{A}_2 = \cdots = \mathcal{A}_r$. □

2.5.2 Optimal access for the case of constant repair subspaces

The codes constructed in Sec. 2.4 above form essentially the only known example of RS codes that afford repair with optimal access. For instance, the optimal-repair RS codes in [77] are not known to support optimal access, and the repair scheme in [77] is far from having this property. Prior works on the problem of access cost for RS repair [16, 19, 45] also do not give examples of repair schemes with optimal access. In this section we show that any family of scalar MDS codes with optimal repair can be furnished with a repair scheme with optimal access, and this includes the code family in [77]. Unfortunately, our results are not explicit; rather, we present an algorithm that produces a basis for representing nodes of the codeword that supports optimal-access repair.

As in Sec. 2.5.1, let F be a subfield of K such that $[K : F] = l$. Let $\mathcal{C}$ be an $(n, k = n - r)$ linear scalar MDS code of length n over K equipped with a repair scheme over F that attains the bound (1.2) for repair of a single node. Let us represent $\mathcal{C}$ in systematic form, choosing a subset $J \subseteq \{1, \ldots, n\}, |J| = r$ for the parity symbols and J^c for the data symbols. Let H be an $r \times n$ parity-check matrix for $\mathcal{C}$ such that H_J is the $r \times r$ identity matrix,

In this section we assume that there exists an optimal repair scheme over F for $\mathcal{C}$ that has constant repair subspaces, and that the repair degree is $d = n - 1$. We will lift both assumptions and show that our result holds in general in the next section. For a given $j = 1, \ldots, n$ consider the subspaces $\mathcal{A}_i, i = 1, \ldots, r$ defined in Proposition 14. Under the assumption of constant repair subspaces, they coincide,

and we use the notation $\mathcal{V}_j$ to refer to any of them.

Consider the following procedure (Algorithm 1) that interatively collects vectors to form a basis of K/F that supports optimal-access repair.

Algorithm 1: Construction of an optimal basis

Input: Subspaces $\mathcal{V}_1, \ldots, \mathcal{V}_n$.
Output: A basis B for K over F.

for $j \leftarrow 1$ **to** n **do**
    $B_j \leftarrow \varnothing$;
    $\mathcal{B}_j \leftarrow \{0\}$;
for $i \leftarrow 0$ **to** $n-1$ **do**
    foreach $I \subseteq \{1, \ldots, n\}$ *such that* $|I| = i$ **do**
        $\bar{I} \leftarrow \{1, \ldots, n\} \backslash I$;
        $\mathcal{U}_I \leftarrow \bigcap_{j \in \bar{I}} \mathcal{V}_j$;
        for $j \leftarrow 1$ **to** n **do**
            if $j \in \bar{I}$ **then**
                $\mathcal{B}_j \leftarrow \mathcal{B}_j + \mathcal{U}_I$;
                Extend the set B_j to a basis of $\mathcal{B}_j$ over F;
$\bar{B} \leftarrow \bigcup_{j=1}^{n} B_j$;
Extend the set $\bar{B}$ to a basis B of K over F;

Proposition 17. *Upon completion of Algorithm 1 we have $\mathcal{B}_j = \mathcal{V}_j$ for $j = 1, \ldots, n$, and thus B_j is a basis for $\mathcal{V}_j$ over F.*

Proof. From Algorithm 1, we have

$$\mathcal{B}_j = \sum_{i=0}^{n-1} \sum_{\substack{|I|=i, \\ I \subseteq \{1,\ldots,n\}}} \mathbb{1}_{\{j \in \bar{I}\}} \bigcap_{t \in \bar{I}} \mathcal{V}_t, \tag{2.45}$$

so clearly $\mathcal{B}_j \subseteq \mathcal{V}_j$. Suppose that $v \in \mathcal{V}_j \backslash \mathcal{B}_j$, then there exists a subset $\bar{I} \subset \{1, \ldots, n\}$

with $1 \leqslant |\bar{I}| \leqslant n$ such that $j \in \bar{I}$ and that

$$v \notin \bigcap_{t \in \bar{I}} \mathcal{V}_t.$$

However, $\mathcal{B}_j \supseteq \bigcap_{t \in \bar{I}} \mathcal{V}_t$ for every $\bar{I}$ with $1 \leqslant |\bar{I}| \leqslant n$ such that $j \in \bar{I}$, which is a contradiction. Hence, $\mathcal{B}_j = \mathcal{V}_j$. □

Proposition 18. *Algorithm 1 returns a basis B for K over F.*

Proof. From Algorithm 1, for every $\bar{I} \subseteq \{1, \ldots, n\}$ with $1 \leqslant |\bar{I}| \leqslant n$ and for every $j \in \bar{I}$, the set B_j contains a basis of the subspace $\mathcal{U}_I = \bigcap_{t \in \bar{I}} \mathcal{V}_t$. It follows that for every $\bar{I} \subseteq \{1, \ldots, n\}$ with $1 \leqslant |\bar{I}| \leqslant n$, the set $\bigcap_{t \in \bar{I}} B_t$ is a basis for $\bigcap_{t \in \bar{I}} \mathcal{V}_t$.

Now by Proposition 17, B_1, B_2 are bases for $\mathcal{V}_1, \mathcal{V}_2$ over F, respectively. From the above, we have $B_1 \cap B_2$ is a basis of $\mathcal{V}_1 \cap \mathcal{V}_2$ over F. It follows that $\dim_F(\mathcal{V}_1 \cap \mathcal{V}_2) = |B_1 \cap B_2|$. Then

$$\begin{aligned} \dim_F(\mathcal{V}_1 + \mathcal{V}_2) &= \dim_F(\mathcal{V}_1) + \dim_F(\mathcal{V}_2) - \dim_F(\mathcal{V}_1 \cap \mathcal{V}_2) \\ &= |B_1| + |B_2| - |B_1 \cap B_2| \\ &= |B_1 \cup B_2|. \end{aligned}$$

By definition, $\mathrm{Span}_F(B_1 \cup B_2) = \mathcal{V}_1 + \mathcal{V}_2$, and so the set $B_1 \cup B_2$ is a basis of $\mathcal{V}_1 + \mathcal{V}_2$ over F. By a straightforward induction argument, we conclude that $\bigcup_{j=1}^{n} B_j$ is a basis for $\sum_{j=1}^{n} \mathcal{V}_j$ over F.

Since $\sum_{j=1}^{n} \mathcal{V}_j \subseteq K$, we have $|\bigcup_{j=1}^{n} B_j| \leqslant [K : F] = l$. It follows that we can extend the set $\bar{B} = \bigcup_{j=1}^{n} B_j$ to a basis B of K over F. □

Now we are ready to present a repair scheme for the code $\mathcal{C}$ with the optimal access property. Let $B = (b_m)$ be the basis of K over F constructed above and let $B^* = (b_m^*)$ be its dual basis. Given a codeword $c = (c_1, \ldots, c_n) \in \mathcal{C}$, we expand its coordinates in the basis B^*, writing

$$c_i = \sum_{m=0}^{l-1} c_{i,m} b_m^*. \tag{2.46}$$

Suppose that c_i is the erased coordinate of c (the "failed node"). The starting point, as above, is Eq. (2.31), and our first step is to choose l dual codewords $c_t^\perp, t = 1, \ldots, l$ that support the repair. Construct the $l \times n$ matrix $C^\perp = AH$ and take the rows of C to be the needed codewords $c_t^\perp$. Since $c_t^\perp \cdot c = 0$ for all t, we have $c_{t,i}^\perp c_i = -\sum_{\substack{j=1 \\ j \neq i}}^{n} c_{t,j}^\perp c_j$ for all $t = 1, \ldots, l$. Computing the trace $\mathrm{tr}_{K/F}$, we obtain

$$\begin{aligned} \mathrm{tr}_{K/F}(c_{t,i}^\perp c_i) &= -\sum_{j \neq i}^{n} \mathrm{tr}_{K/F}(c_{t,j}^\perp c_j) \\ &= -\sum_{j \neq i}^{n} \mathrm{tr}_{K/F}(c_{t,j}^\perp \sum_{m=0}^{l-1} c_{j,m} b_m^*) \\ &= -\sum_{j \neq i}^{n} \sum_{m=0}^{l-1} \mathrm{tr}_{K/F}(c_{t,j}^\perp b_m^*) c_{j,m}. \end{aligned} \tag{2.47}$$

Note that for each $j \in \{1, \ldots, n\} \backslash \{i\}$, we have

$$\mathrm{Span}_F(c_{1,j}^\perp, \ldots, c_{l,j}^\perp) = \mathrm{Span}_F(Ah_j) = \mathcal{V}_i, \tag{2.48}$$

where the last equality follows by the assumption of constant repair subspaces. By Proposition 17, the set $B_i \subseteq B$ is a basis for $\mathcal{V}_i$ over F. Therefore, $c_{t,j}^\perp$ can be linearly

generated by the set B_i for every $t = 1, \ldots, l$. More precisely, let $B_i = \{b_{i,u} \,|\, u = 1, \ldots, l/r\}$, then we have

$$c_{t,j}^{\perp} = \sum_{u=1}^{l/r} \gamma_{j,u} b_{i,u} \tag{2.49}$$

for some $\gamma_{j,u}, u = 1, \ldots, l/r$. Substituting into (2.47), we obtain the equality

$$\mathrm{tr}_{K/F}(c_{t,i}^{\perp} c_i) = -\sum_{j \neq i}^{n} \sum_{m=0}^{l-1} \sum_{u=1}^{l/r} \mathrm{tr}_{K/F}(b_{i,u} b_m^*) \gamma_{j,u} c_{j,m}. \tag{2.50}$$

It follows that to determine the left-hand side of (2.50), on each node $c_j, j \neq i$ the repair procedure needs to access the set of symbols $\{c_{j,m} \,|\, \mathrm{tr}_{K/F}(b_{i,u} b_m^*) = 1\}$. Since $B_i \subseteq B$ and B^* is the dual basis of B for K over F, the cardinality of this subset equals $|B_i| = l/r$, verifying that the repair can be accomplished with the minimum possible access cost.

2.5.3 Optimal-access repair for general scalar MSR codes

In this section we extend the above arguments for optimal repair schemes that do not necessarily have constant repair subspaces. This is done by a simple extension of Algorithm 1. We use the same notation as in Sec. 2.5.2.

2.5.3.1 Repair degree $d = n - 1$

Assume that the index of the failed node is $i \in \{1, \ldots, n\}$. By Proposition 14, for each $j \in \{1, \ldots, n\} \backslash \{i\}$, we have

$$\dim_F(\mathcal{A}_u) = \dim_F(Ah_j) = \frac{l}{r}, \quad u = 1, \ldots, r.$$

It follows that for $j \in J^c \backslash \{i\}$ we have

$$\mathcal{A}_1 h_{j,1} = \mathcal{A}_2 h_{j,2} = \cdots = \mathcal{A}_r h_{j,r}.$$

Let $J = (i_1, \ldots, i_r)$ be the set of parity nodes written in increasing order of their indices, and for $i_t \in J$ let $\sigma(i_t) = t$. Define

$$\mathcal{V}_i^{(j)} = \begin{cases} \mathcal{A}_1 h_{j,1} & j \in J^c \backslash \{i\}, \\ \mathcal{A}_{\sigma(j)} & j \in J. \end{cases} \tag{2.51}$$

Proposition 19. *When Algorithm 2 terminates, we have* $\mathcal{B}_i^{(j)} = \mathcal{V}_i^{(j)}$ *for* $i \in \{1, \ldots, n\}$ *and* $j \in \{1, \ldots, n\} \backslash \{i\}$, *and thus* $B_i^{(j)}$ *is a basis for* $\mathcal{V}_i^{(j)}$ *over* F.

Proposition 20. *Algorithm 2 returns a basis* B *for* K *over* F.

The proofs of Propositions 19 and 20 follow closely the proofs of Proposition 17 and 18 and will be omitted.

Now it is not difficult to see that we can repair the failed node c_i with optimal

Algorithm 2: Construction of an optimal basis; repair degree $d = n - 1$

Input: Subspaces $\mathcal{V}_i^{(j)}, i \in \{1, \ldots, n\}, j \in \{1, \ldots, n\} \backslash \{i\}$.
Output: A basis B for K over F.

for $i \leftarrow 1$ **to** n **do**
    foreach $j \in \{1, \ldots, n\} \backslash \{i\}$ **do**
        $B_i^{(j)} \leftarrow \varnothing$;
        $\mathcal{B}_i^{(j)} \leftarrow \{0\}$;
$\Omega \leftarrow \{1, \ldots, n\}^2 \backslash \{(i, i) \mid i = 1, \ldots, n\}$;
for $u \leftarrow 0$ **to** $n^2 - n - 1$ **do**
    foreach $I \subseteq \Omega$ *such that* $|I| = u$ **do**
        $\bar{I} \leftarrow \Omega \backslash I$;
        $\mathcal{U}_I \leftarrow \bigcap_{(i,j) \in \bar{I}} \mathcal{V}_i^{(j)}$;
        for $i \leftarrow 1$ **to** n **do**
            foreach $j \in \{1, \ldots, n\} \backslash \{i\}$ **do**
                if $(i, j) \in \bar{I}$ **then**
                    $\mathcal{B}_i^{(j)} \leftarrow \mathcal{B}_i^{(j)} + \mathcal{U}_I$;
                    Extend the set $B_i^{(j)}$ to be a basis of $\mathcal{B}_i^{(j)}$ over F;
$\bar{B} \leftarrow \bigcup_{i=1}^{n} \bigcup_{j \neq i}^{n} B_i^{(j)}$;
Extend the set $\bar{B}$ to be a basis B for K over F;

access cost relying on the basis B. Indeed, for each $j \in \{1, \ldots, n\} \backslash \{i\}$, we have

$$\mathrm{Span}_F(c_{1,j}^{\perp}, \ldots, c_{l,j}^{\perp}) = \mathrm{Span}_F(Ah_j) = \mathcal{V}_i^{(j)}. \tag{2.52}$$

By Algorithm 2 and Proposition 19, the set $B_i^{(j)} \subseteq B$ is a basis for $\mathcal{V}_i^{(j)}$ over F. Therefore, $c_{t,j}^{\perp}$ can be linearly generated by the set $B_i^{(j)}$ for every $t = 1, \ldots, l$. Let $B_i^{(j)} = \{b_{i,u}^{(j)} \mid u = 1, \ldots, l/r\}$. Then, similarly to (2.49) and (2.50), we have

$$c_{t,j}^{\perp} = \sum_{u=1}^{l/r} \gamma_{j,u} b_{i,u}^{(j)}, \tag{2.53}$$

$$\mathrm{tr}_{K/F}(c_{t,i}^{\perp} c_i) = -\sum_{j \neq i}^{n} \sum_{m=0}^{l-1} \sum_{u=1}^{l/r} \mathrm{tr}_{K/F}(b_{i,u}^{(j)} b_m^*) \gamma_{j,u} c_{j,m}. \tag{2.54}$$

Therefore, each node $c_j, j \neq i$ needs to access the set of symbols $\{c_{j,m} \mid \mathrm{tr}_{K/F}(b_{i,u}^{(j)} b_m^*) = 1\}$, whose cardinality is given by $|B_i^{(j)}| = l/r$. It follows that the repair scheme has the optimal access property.

2.5.3.2 Arbitrary repair degree

So far we assumed that the repair relies on all the surviving nodes except for the single failed node, i.e., $|\mathcal{R}| = n - 1$. In this section we derive the most general version of the result of this section, that any scalar MDS code can be repaired with optimal access from any subset of helper nodes $\mathcal{R}$ of size $d, k + 1 \leqslant d \leqslant n - 1$. Let $s := d - k + 1$.

Let $G = [g_1 | g_2 | \ldots | g_n]$ be a $k \times n$ generator matrix of $\mathcal{C}$, where g_i is a k-column over K. Let $i \in \{1, \ldots, n\}$ and let $\mathcal{R} \not\ni \{i\}$ be a subset of d helper nodes. Let

$\hat{\mathcal{R}} = \mathcal{R} \cup \{i\}$ and $G_{\hat{\mathcal{R}}}$ be the $k \times (d+1)$ submatrix formed by the columns $g_j, j \in \hat{\mathcal{R}}$. Clearly, $G_{\hat{\mathcal{R}}}$ defines a $(d+1, k)$ punctured code $\mathcal{C}_{\hat{\mathcal{R}}}$ of the code $\mathcal{C}$. Since $\mathcal{C}$ is MDS, the code $\mathcal{C}_{\hat{\mathcal{R}}}$ is itself MDS. Let $H^{\hat{\mathcal{R}}} = (h_i^{(\hat{\mathcal{R}})}, i = 1, \ldots, d+1)$ be an the $s \times (d+1)$ parity-check matrix of the code $\mathcal{C}_{\hat{\mathcal{R}}}$. Recalling Remark 1, the code generated by $H^{\hat{\mathcal{R}}}$ is a shortened code $(\mathcal{C}^{\perp})^{\hat{\mathcal{R}}}$, i.e., a subcode of $\mathcal{C}^{\perp}$ formed of the codewords with zeros in the coordinates in $\hat{\mathcal{R}}^c$.

Suppose that the code $\mathcal{C}$ can optimally repair any single failed node i from the coordinates in $\mathcal{R} = \hat{\mathcal{R}} \backslash \{i\}$. This means that the MDS code $\mathcal{C}_{\hat{\mathcal{R}}}$ can optimally repair any single failed node i from the helper nodes $\hat{\mathcal{R}} \backslash \{i\}$. Let $J \subseteq \hat{\mathcal{R}}, |J| = s$ and $i \notin J$ and assume without loss of generality that the submatrix $H_J^{\hat{\mathcal{R}}}$ is an $s \times s$ identity matrix. Now Proposition 14 applied for the code $\mathcal{C}_{\hat{\mathcal{R}}}$ guarantees that there exist vectors $a_u \in K^{l/s}, u = 1, \ldots, s$ such that the block-diagonal matrix $A = \mathrm{Diag}(a_1, \ldots, a_s)$ satisfies

$$\dim_F(A h_i^{(\hat{\mathcal{R}})}) = l, \tag{2.55}$$

$$\dim_F(A h_j^{(\hat{\mathcal{R}})}) = \frac{l}{s}, \quad j \in \hat{\mathcal{R}} \backslash \{i\}, \tag{2.56}$$

$$\dim_F(\mathcal{A}_u) = \frac{l}{s}, \quad u = 1, \ldots, s, \tag{2.57}$$

where $\mathcal{A}_u := \mathrm{Span}_F(a_u)$.

It follows from (2.56) and (2.57) that for $j \in \hat{\mathcal{R}} \backslash (J \cup \{i\})$, we have

$$\mathcal{A}_1 h_{j,1}^{(\hat{\mathcal{R}})} = \mathcal{A}_2 h_{j,2}^{(\hat{\mathcal{R}})} = \cdots = \mathcal{A}_s h_{j,s}^{(\hat{\mathcal{R}})}.$$

Let us define

$$\mathcal{V}_{\hat{\mathcal{R}},i}^{(j)} = \begin{cases} \mathcal{A}_1 h_{j,1}^{(\hat{\mathcal{R}})} & j \in \hat{\mathcal{R}} \backslash (J \cup \{i\}), \\ \\ \mathcal{A}_{\sigma(i)} & j \in J, \end{cases} \tag{2.58}$$

where σ is a bijection between J and $\{1, \ldots, s\}$ defined as before (2.51).

The procedure to construct a basis for optimal-access repair in this case is constructed as a modification of Algorithm 2, and is given in Algorithm 3.

Algorithm 3: Construction of an optimal basis; arbitrary repair degree

Input: Subspaces $\mathcal{V}_{\hat{\mathcal{R}},i}^{(j)}$ for each $\hat{\mathcal{R}} \subseteq \{1, \ldots, n\}$ such that $|\hat{\mathcal{R}}| = d+1$ and $i \in \hat{\mathcal{R}}, j \in \hat{\mathcal{R}} \backslash \{i\}$.

Output: A basis B for K over F.

foreach $\hat{\mathcal{R}} \subseteq \{1, \ldots, n\}$ *such that* $|\hat{\mathcal{R}}| = d+1$ **do**
  foreach $i \in \hat{\mathcal{R}}$ **do**
    foreach $j \in \hat{\mathcal{R}} \backslash \{i\}$ **do**
      $B_{\hat{\mathcal{R}},i}^{(j)} \leftarrow \varnothing$;
      $\mathcal{B}_{\hat{\mathcal{R}},i}^{(j)} \leftarrow \{0\}$;
$\Omega \leftarrow \{(\hat{\mathcal{R}}, i, j) \mid \hat{\mathcal{R}} \subseteq \{1, \ldots, n\}, i \in \hat{\mathcal{R}}, j \in \hat{\mathcal{R}} \backslash \{i\}\}$;
for $u \leftarrow 0$ **to** $\binom{n}{d+1}((d+1)^2 - (d+1)) - 1$ **do**
  foreach $I \subseteq \Omega$ *such that* $|I| = u$ **do**
    $\bar{I} \leftarrow \Omega \backslash I$;
    $\mathcal{U}_I \leftarrow \bigcap_{(\hat{\mathcal{R}},i,j) \in \bar{I}} \mathcal{V}_i^{(j)}$;
    foreach $\hat{\mathcal{R}} \subseteq \{1, \ldots, n\}$ *such that* $|\hat{\mathcal{R}}| = d+1$ **do**
      foreach $i \in \hat{\mathcal{R}}$ **do**
        foreach $j \in \hat{\mathcal{R}} \backslash \{i\}$ **do**
          if $(\hat{\mathcal{R}}, i, j) \in \bar{I}$ **then**
            $\mathcal{B}_{\hat{\mathcal{R}},i}^{(j)} \leftarrow \mathcal{B}_{\hat{\mathcal{R}},i}^{(j)} + \mathcal{U}_I$;
            Extend the set $B_{\hat{\mathcal{R}},i}^{(j)}$ to be a basis of $\mathcal{B}_{\hat{\mathcal{R}},i}^{(j)}$ over F;
$\bar{B} \leftarrow \bigcup_{\hat{\mathcal{R}} \subseteq \{1,\ldots,n\}, |\hat{\mathcal{R}}| = d+1} \bigcup_{i \in \hat{\mathcal{R}}} \bigcup_{j \neq \hat{\mathcal{R}} \backslash \{i\}} B_{\hat{\mathcal{R}},i}^{(j)}$;
Extend the set $\bar{B}$ to be a basis B of K over F;

Similarly to the previous sections, we have the following propositions, whose proofs are analogous to the proofs of Propositions 17 and 18.

Proposition 21. *When Algorithm 3 terminates, we have* $\mathcal{B}^{(j)}_{\hat{\mathcal{R}},i} = \mathcal{V}^{(j)}_{\hat{\mathcal{R}},i}$ *for* $\hat{\mathcal{R}} \subseteq \{1,\ldots,n\}$ *with* $|\hat{\mathcal{R}}| = d+1$, $i \in \hat{\mathcal{R}}$, *and* $j \in \hat{\mathcal{R}}\backslash\{i\}$, *and thus* $B^{(j)}_{\hat{\mathcal{R}},i}$ *is a basis of* $\mathcal{V}^{(j)}_{\hat{\mathcal{R}},i}$ *over* F.

Proposition 22. *Algorithm 3 returns a basis* B *of* K *over* F.

The basis of K over F constructed in the algorithm enables us to construct an optimal-access repair scheme for the code $\mathcal{C}$. Let $d \in \{k+1,\ldots,n-1\}$ be the repair degree. Let $(c_1,\ldots,c_n)$ be a codeword of the code $\mathcal{C}$ written on the storage nodes, and suppose that the failed node is i and that $\mathcal{R}$ be the set of d helper nodes. Let A be the block-diagonal matrix defined above, constructed with respect to i and $H^{\hat{\mathcal{R}}}$. Define the matrix $C^{\perp} = AH^{\hat{\mathcal{R}}}$ and note that its rows $c^{\perp}_t, t = 1,\ldots,l$ form codewords of the code dual to the punctured code $\mathcal{C}_{\hat{\mathcal{R}}}$. Letting $c^{\perp}_t = (c^{\perp}_{t,i})_{i\in\mathcal{R}}$, we can write

$$c^{\perp}_{t,i}c_i = -\sum_{j\in\mathcal{R}} c^{\perp}_{t,j}c_j. \tag{2.59}$$

Similarly to (2.47), we have

$$\mathrm{tr}_{K/F}(c^{\perp}_{t,i}c_i) = -\sum_{j\in\mathcal{R}}\sum_{m=0}^{l-1} \mathrm{tr}_{K/F}(c^{\perp}_{t,j}b^*_m)c_{j,m}, \tag{2.60}$$

where $B^* = (b^*)$ is the dual basis of the basis B. Note that for $j \in \mathcal{R}$ we have

$$\mathrm{Span}_F(c^{\perp}_{1,j},\ldots,c^{\perp}_{l,j}) = \mathrm{Span}_F(Ah_j) = \mathcal{V}^{(j)}_{\mathcal{R},i}. \tag{2.61}$$

By Algorithm 3 and Proposition 21, the set $B^{(j)}_{\mathcal{R},i} \subseteq B$ forms a basis for the subspace $\mathcal{V}^{(i)}_{\mathcal{R},j}$ over F. Therefore, the element $c^{\perp}_{t,j}$ can be linearly generated by the set $B^{(j)}_{\mathcal{R},i}$ for every $t = 1, \ldots, l$. Let $B^{(j)}_{\mathcal{R},i} = \{b^{(j)}_{\mathcal{R},i,u} \,|\, u = 1, \ldots, l/s\}$. Then, similarly to (2.49) and (2.50), we have

$$c^{\perp}_{t,j} = \sum_{u=1}^{l/s} \gamma_{j,u} b^{(j)}_{\mathcal{R},i,u}, \tag{2.62}$$

$$\mathrm{tr}_{K/F}(c^{\perp}_{t,i} c_i) = -\sum_{j \in \mathcal{R}} \sum_{m=0}^{l-1} \sum_{u=1}^{l/s} \mathrm{tr}_{K/F}(b^{(j)}_{\mathcal{R},i,u} b^*_m) \gamma_{j,u} c_{j,m}. \tag{2.63}$$

Therefore, each node $c_j, j \in \mathcal{R}$ needs to access the set of symbols $\{c_{j,m} \,|\, \mathrm{tr}_{K/F}(b^{(j)}_{\mathcal{R},i,u} b^*_m) = 1\}$, whose cardinality equals $|B^{(j)}_{\mathcal{R},i}| = l/s$. It follows that the constructed repair scheme has the optimal access property.

This completes the proof of Theorem 12.

Chapter 3: Explicit Constructions of MSR Codes for the Rack-aware Storage Model

3.1 Introduction

In this chapter we consider a model of storage that assumes that nodes are organized into equally sized groups, called racks, that within each group the nodes can communicate freely without taxing the system bandwidth, and that the only information transmission that counts is the one between the racks. This assumption implies that the nodes within each of the racks can collaborate before providing information to the failed node. The main emphasis of the chapter is on code construction for this storage model. We present an explicit family of MDS array codes that support recovery of a single failed node from any number of helper racks using the minimum possible amount of inter-rack communication (such codes are said to provide optimal repair). The codes are constructed over finite fields of size comparable to the code length.

We also derive a bound on the number of symbols accessed at helper nodes for the purposes of repair, and construct a code family that approaches this bound, while still maintaining the optimal repair property.

Finally, we present a construction of scalar Reed-Solomon codes that support optimal repair for the rack-oriented storage model. We also show how the RS code families and repair schemes presented in Chapter 2 can be modified to enable optimal error correction and low access for the rack-aware storage model.

3.1.1 Organization

We start with the problem statement of the rack-aware storage model and some structural lemmas for the model in Sec. 3.2, and then move on to present the first explicit construction of rack-aware MSR code for all admissible parameters in Sec. 3.3. In Sec. 3.4, we construct a family rack-aware MSR code with low access. In Sec. 3.5 we extend our approach of constructing rack-aware vector MSR codes to (scalar) RS codes.

3.2 Problem statement and structural lemmas

Assume that the data file of size M is divided into k blocks and encoded using an array code $\mathcal{C}$ of length n over some finite field F. Each symbol of the codeword is represented by an l-dimensional vector over F and is placed on a separate storage node. We assume that the code is MDS, i.e., the entire codeword can be recovered from any k of its coordinates (from the encoding stored on any k out of the n nodes). According to the cut-set bound of [20], the amount of information required for repair

of a single node from d helper nodes satisfies the inequality

$$\beta(d) \geqslant \frac{dl}{d-k+1}, \tag{3.1}$$

where $k \leqslant d \leqslant n-1$.

Suppose that information is encoded with an MDS array code $\mathcal{C}$ of length $n = \bar{n}u$ over a finite field F. If the size of the code is q^{kl}, we refer to it as a $\mathcal{C}(n,k,l)$ code. The set of nodes $[n] = \{1,2,\ldots,n\}$ is partitioned into $\bar{n}$ subsets (racks) of size u each. Accordingly, the coordinates of the codeword $c \in \mathcal{C}$ are partitioned into segments of length u, and we label them as $c_t, t = 1,\ldots,n$, where $t = (m-1)u + j, 1 \leqslant m \leqslant \bar{n},\ 1 \leqslant j \leqslant u$. We do not distinguish between the nodes and the coordinates of the codeword, and refer to both of them as nodes. Each node is an element in F^l, and when needed, we denote its entries as $c_{t,j}, j = 1,\ldots,l$.

Denote by $\mathcal{R} \subset \{1,\ldots,\bar{n}\}$ the set of $\bar{d}$ helper racks and let m^* be the index of the host rack. To repair the failed node, information is generated in the helper racks and is combined with the contents of the local nodes to perform the repair. This is modeled by computing a linear function of the contents of the nodes within each helper rack (the function depends on the contents of all the nodes in the rack, and can in principle also depend on the rack index), and sending this information to rack m^*.

Definition 8 (Repair scheme)**.** *Let $\mathcal{C}(n,k,l)$ be an array code. Suppose that node $c_{(m^*-1)u+j^*}$ is erased (has failed). To recover the lost data, we rely on the values of the symbols in coordinates c_{iu+j}, where $i \in \mathcal{R}$ and $j = 1,\ldots,u$. A* repair scheme $\mathcal{S}$

with repair degree $\bar{d} \leqslant \bar{n} - 1$ *is formed of* $\bar{d}$ *functions* $f_i : F^{ul} \to F^{\beta_i}, i \in \mathcal{R}$ *and a function* $g : F^{\sum_{i\in\mathcal{R}} \beta_i} \times F^{(u-1)l} \to F^l$. *For a given* $i \in \mathcal{R}$ *the function* f_i *maps* $c^{(i)}$ *(the nodes in rack* i*) to some* β_i *symbols of* F. *The function* g *accepts these symbols together with the available nodes in the host rack as arguments, and returns the value of the failed node:*

$$g(\{f_i(c_{(i-1)u+j}, 1 \leqslant j \leqslant u), i \in \mathcal{R}\}, \{c_{(m^*-1)u+j}, j \in \{1, \ldots, u\}\backslash\{j^*\}\}) = c_{(m^*-1)u+j^*}.$$

In general the function $f_i, i \in \mathcal{R}$ *depends on* i, m^* *and* j^*, *and the function* g *depends on* $\mathcal{R}, m^*, j^*$.

The quantity $\beta(\mathcal{R}, m^*, j^*) = \sum_{i\in\mathcal{R}} \beta_i$ *is called the repair bandwidth of the node* $c_{(m^*-1)u+j^*}$ *from the helper racks in* $\mathcal{R}$ *and from the available nodes in the host rack* m^*.

The repair scheme can be defined in a more general way: for instance, each of the functions f_i that form the information downloaded by the failed node could depend on the entire set $\mathcal{R}$ (and not just on the contents of the node i) and the function g could depend on the labels of the helper nodes in addition to the information downloaded from them. At the same time, all our results as well as all the results in the earlier literature are well described by this definition, which therefore suffices for our purposes. If the functions f_i, g are F-linear, the repair scheme itself is called *linear*. Only such schemes will be considered below.

Let

$$\beta_u(\bar{d}) := \min_{\mathcal{C}\subset F^{nl}} \max_{\mathcal{R},m^*,j^*} \beta(\mathcal{R}, m^*, j^*)$$

where the minimum is taken over all $(n, M = q^{kl})$ MDS array codes and the maximum over the index of the host rack, the failed node in the rack, and the choice of the set of the helper racks $\mathcal{R}$. To rule out the trivial case, we assume throughout that $k \geqslant u$.

3.2.1 Optimal repair

Suppose that $k = \bar{k}u + v$, where $0 \leqslant v \leqslant u - 1$. A necessary condition for successful repair of a single node is given by a version of the cut-set bound [32], [31] which states that for any (n, k, l) MDS array code, the (inter-rack) repair bandwidth is at least

$$\beta_u(\bar{d}) \geqslant \frac{\bar{d}l}{\bar{d} - \bar{k} + 1} \tag{3.2}$$

The code that attains this bound with equality is said to have the optimal repair property.

The arguments below are based on the following obvious (and well-known) observation.

Lemma 23. *Let $\mathcal{C}(n, k, l)$ be an MDS array code. Suppose that a failed node is repaired using a set $\mathcal{I}, |\mathcal{I}| = d$ of helper nodes. The number of symbols of F downloaded for the repair task from any subset $\mathcal{I}' \subset \mathcal{I}$ of size $|\mathcal{I}'| = d - k + 1$ is at least l.*

To prove this it suffices to observe that, because of the MDS property, no subset of $k - 1$ nodes carries any information about the value of any other node.

We note that this lemma applies to the rack model (i.e., allowing processing of the information obtained from the nodes in $\mathcal{I}$). It also applies if the count of downloaded symbols is replaced by the count of symbols *accessed* on the helper nodes.

The next statement, called the *uniform download property*, is well known for the case of homogeneous storage. Its proof for the rack-aware storage is not much different, and is given for completeness in Appendix B.1.

Proposition 24. *Let $\mathcal{C}$ be an MSR code and suppose that $\bar{k} > 1$. Let $\mathcal{R}$ be the set of helper racks used to repair a single failed node. Then $\beta_i = l/(\bar{d} - \bar{k} + 1), i \in \mathcal{R}$.*

We note that both the bound (3.2) and this proposition can be generalized to the case of $2 \leqslant h \leqslant r$ failed nodes located on the same rack without any difficulty; for instance, the bound takes the form $\beta \geqslant \frac{h\bar{d}l}{\bar{d}-\bar{k}+1}$.

Next, observe that if k is divisible by the rack size u, then any MSR code for the standard model will be optimal for the rack model, i.e., cooperation between the nodes within the rack does not help to reduce the repair bandwidth (this has been first observed in [31, Thm. 4]).

Proposition 25. *Let $k = \bar{k}u$, and let $\mathcal{C}$ be an MSR code of length $n = \bar{n}u$ with optimal repair of a single node for the homogeneous storage model. Then $\mathcal{C}$ attains the cut-set bound (3.2) for repair of any single node in the rack-aware model.*

Proof: Take an MSR code of length n and assume that $v = 0$. Suppose that the number of helper nodes is d, and this includes the $u - 1$ local nodes. By (3.1), the repair bandwidth necessary equals $\frac{d}{d-k+1}l$. In accordance with the model, take

$d = \bar{d}u + (u-1)$, then

$$\frac{d}{d-k+1}l = \Big(\frac{\bar{d}}{\bar{d}-\bar{k}+1} + \frac{u-1}{d-k+1}\Big)l \tag{3.3}$$

and this achieves the bound (3.2) if the second term is discounted (which is possible because of the uniform download property and because intra-rack communication is free). □

Note that in the case of $v \neq 0$, optimal codes for the rack model perform repair using a strictly smaller repair bandwidth than optimal codes for the homogeneous model. This also suggests that the number of symbols downloaded from a helper rack is strictly smaller than the number of accessed symbols, i.e., intra-rack processing is necessary for optimal repair (this will be made rigorous once we establish Prop. 26 below).

For reader's convenience, let us summarize the code parameters: We consider (n, k, l) array codes used in a system where the nodes are arranged in racks of size u. The codes are designed to repair a single node. We further assume that $n = \bar{n}u, k = \bar{k}u + v$, where $0 < v \leqslant u-1$, and the number of helper racks is $\bar{d}$, where $\bar{k} \leqslant \bar{d} \leqslant \bar{n} - 1$. We also use the notation $r = n - k, \bar{r} = \bar{n} - \bar{k}$ for the number of parity nodes and parity racks, respectively. Finally, to shorten the formulas we denote

$$s = d - k + 1, \quad \bar{s} = \bar{d} - \bar{k} + 1,$$

where d is the total number of helper nodes accessed for repair, and $\bar{d}$ is the *repair degree*, i.e., number of helper racks (not counting the host rack).

3.2.2 Optimal access

Some of the constructions of codes for the homogeneous case have the additional property that the information accessed on the helper nodes is the same as the information that is downloaded by the helper node (no processing is performed before downloading). This property, also called *repair by transfer*, reduces the implementation overhead, and is therefore desirable in the code construction. Structure and constructions of optimal access (OA) codes for the homogeneous case were addressed in [75, 81, 85] among others.

Definition 9. *Let $\mathcal{C}(n = \bar{n}u, k, l)$ be a code that supports optimal repair of a single failed node with repair degree $\bar{d}$. Suppose that each of the helper racks provides $l/\bar{s}$ field symbols and these symbols are generated by accessing the smallest possible number of symbols of the nodes in the rack. In this case we say that $\mathcal{C}$ has the OA property.*

To motivate this definition, we draw an analogy with the homogeneous case. In this case, on account of the bound (3.1) and the uniform download property, the system accesses l/s symbols at each of the helper nodes, and these symbols are downloaded to accomplish the repair. As a consequence, a group of $u > 1$ helper nodes provides ul/s symbols. This observation also extends to the rack-aware model in the case that $u|k$. Indeed, in this case the number of symbols downloaded from,

and accessed on, each rack equals $l/\bar{s} = ul/s$.

In the next proposition (proved in Appendix B.2) we derive a lower bound on the number of accessed symbols and establish the *uniform access condition*.

Proposition 26. *Let $\mathcal{C}$ be an (n, k, l) optimal-repair MDS array code for the rack model with repair degree $\bar{d} \geqslant \bar{k} + 1$ and $u \leqslant k$. The total number of symbols accessed on the helper racks for repair of a single node satisfies*

$$\alpha \geqslant \frac{\bar{d}ul}{s}. \tag{3.4}$$

Equality holds if and only if the number of symbols accessed on node e satisfies $\alpha_{m,e} = l/s$ for all $m \in \mathcal{R}$; $e = 1, \ldots, u$.

As noted above, if $u|k$, the symbols accessed on the helper nodes can be downloaded without processing, accounting for optimal repair. At the same time, if $u \nmid k$, and the code meets the bound (3.4), then processing is necessary because $\bar{d}ul/s$ is strictly greater than the optimal bandwidth in (3.2).

3.2.3 A lower bound on the sub-packetization of rack-aware optimal-access MSR codes

In this section we present a lower bound on the value of the node size in MSR codes for the rack model, which will be implicitly assumed throughout without further mention. Similarly to [3, 75], we limit ourselves to systematic codes and linear repair schemes. Let $\mathcal{C}$ be an $(n = \bar{n}u, k = \bar{k}u, l)$ systematic optimal-access

MSR array code over F. Let $A = (A_{ij})$ be the $((n-k)l \times kl)$ encoding matrix of $\mathcal{C}$; in other words, the parity symbols $c_{k+i}, i = 1 \ldots, r = n-k$ are obtained from the data symbols $c_j, j = 1, \ldots, k$ according to the relation

$$c_{k+i} = \sum_{j=1}^{k} A_{i,j} c_j, \tag{3.5}$$

where each $A_{i,j}$ is an $l \times l$ invertible matrix over F. Assume without loss of generality that the k systematic nodes are located on racks $1, \ldots, \bar{k}$, called systematic racks below. Racks $\bar{k}+1, \ldots, \bar{n}$ will be called parity racks. Let $\boldsymbol{c}_m = (c_{(m-1)u+1}, \ldots, c_{mu})^T$ be the data vector stored in the m-th rack, $1 \leqslant m \leqslant \bar{k}$, where each component is an l-vector over F. Suppose for definiteness that the failed node is located in rack m_1, where $1 \leqslant m_1 \leqslant \bar{k}$. Suppose further that the set of $\bar{d}$ helper racks is formed of the remaining $\bar{k}-1$ systematic racks and some $\bar{s} = \bar{d} - \bar{k} + 1$ parity racks.

We assume throughout that the repair scheme is independent of the index of the failed node in its rack.

The main result of this section is given in the following theorem, whose proof is modeled on the result of [3] and generalizes its main ideas to the case of $u \geqslant 2$.

Theorem 27. *Let $\mathcal{C}$ be an $(n = \bar{n}u, k = \bar{k}u, l)$ optimal-access MSR array code, $k \geqslant u$, and let $\bar{d}, \bar{k} \leqslant \bar{d} \leqslant \bar{n} - 1$ be the size of the helper set $\mathcal{R}$. Suppose further that there is a linear repair scheme that supports repair of a single failed node from any $\bar{d}$ helper racks.*

(a) Suppose that the repair scheme depends on the choice of the helper racks

as well as on the index of the host rack. Then

$$l \geqslant \min\{\bar{s}^{(\bar{n}-1)/s}, \bar{s}^{\bar{k}-1}\}, \tag{3.6}$$

where $\bar{s} = \bar{d} - \bar{k} + 1$ *and* $s = \bar{s}u$.

(b) Suppose that the repair scheme depends on the index of the host rack but not on the choice of the helper racks, then

$$l \geqslant \min\{\bar{s}^{\bar{n}/s}, \bar{s}^{\bar{k}-1}\}. \tag{3.7}$$

A proof of this theorem is given in the Appendix. Here let us make the following remark. The theorem is proved under the assumption that $u|k$, in which case any optimal-access MSR code for the homogeneous storage model supports optimal repair for the rack model. The smallest possible value of sub-packetization for such codes is $l = r^{\lceil \frac{n-1}{r} \rceil}$ [3, 85]. Thus, this theorem says that it is possible that there exist optimal-access rack codes that have smaller node size than OA codes for homogeneous storage *even in the case when* k *is a multiple of* u.

3.3 Rack-aware codes with optimal repair for all parameters

Let $\bar{s} = \bar{d} - \bar{k} + 1$ and let $F, |F| > \bar{s}n$ be a finite field. The code that we construct is formed as an F-linear array MDS code $\mathcal{C}$ of length n, dimension k, and sub-packetization $l = \bar{s}^{\bar{n}}$. We denote a codeword of $\mathcal{C}$ by $(c_1, c_2, \ldots, c_n)$, where $c_i = (c_{i,1}, \ldots, c_{i,l})$ for all $i = 1, \ldots, n$. Suppose that $\bar{s}n \,|\, (|F| - 1)$ and let $\lambda \in F$ be

an element of multiplicative order $\bar{s}n$. Finally, given $j \in \{0, 1, \ldots l-1\}$, consider the base $\bar{s}$ expansion $j = (j_{\bar{n}}, j_{\bar{n}-1}, \ldots, j_1)$ and let

$$j(p, a) := (j_{\bar{n}}, \ldots, j_{p+1}, a, j_{p-1}, \ldots, j_1), \tag{3.8}$$

where $0 \leqslant a \leqslant \bar{s} - 1$.

Construction 3.3.1. *Consider an* $(n, k, l = \bar{s}^{\bar{n}})$ *code* $\mathcal{C} = \{\boldsymbol{c} = (c_{i,j})_{1 \leqslant i \leqslant n; 0 \leqslant j \leqslant l-1}\}$ *defined by the following set of* rl *parity-check equations over* F*:*

$$\sum_{e=1}^{\bar{n}} \lambda^{t((e-1)\bar{s}+j_e)} \sum_{i=1}^{u} \lambda^{t(i-1)\bar{s}\bar{n}} c_{(e-1)u+i,j} = 0 \tag{3.9}$$

for all $t = 0, \ldots, r-1; j = 0, \ldots, l-1$.

We will show that the code defined in (3.9) is an MDS code that has the smallest possible repair bandwidth according to the bound (3.2). Before stating the main theorem that proves these claims let us comment on the origin as well as the new elements in this construction. The code is formed of two levels, the algebraic one, which accounts for the repair of a node in any *fixed* rack, say $p, 1 \leqslant p \leqslant \bar{n}$, and a stacking construction which makes the code universal (i.e., rack-independent). The second part is accomplished by representing the index j of the parity check equation as an $\bar{s}$-ary number (3.8). This expansion enables us to isolate the parities that are used to perform repair of any failed node in rack p, specifically, they are the equations in (3.9) whose label j is obtained by varying the value of the entry j_p in the expansion (3.8) and fixing all the remaining values.

The algebraic development represents the main part of the proof of Theorem 28 and accounts for the optimal-bandwidth repair scheme. The key new idea utilized in the proof is the choice of λ based on the multiplicative structure of F and using the evaluation points given by the powers of λ.

Theorem 28. *Let $\bar{k} \leqslant \bar{d} \leqslant \bar{n} - 1$. The $(n, k, l = \bar{s}^{\bar{n}})$ code $\mathcal{C}$ defined by the parity-check equations (3.9) is an MDS code that supports optimal repair of any single node from any $\bar{d}$ helper racks, under the rack-aware storage model.*

Proof. We begin with proving the part of the claim about the repair properties of the code $\mathcal{C}$. Suppose that the index of the rack that contains the failed node is $p \in \{1, \ldots, \bar{n}\}$. We have $\bar{r}u = r + v$ and since $0 \leqslant v \leqslant u - 1$, $(\bar{r} - 1)u \leqslant r - 1$. Rewriting (3.9), we have:

$$\lambda^{t((p-1)\bar{s}+j_p)} \sum_{i=1}^{u} \lambda^{t(i-1)\bar{s}\bar{n}} c_{(p-1)u+i,j} = - \sum_{\substack{e=1 \\ e \neq p}}^{\bar{n}} \lambda^{t((e-1)\bar{s}+j_e)} \sum_{i=1}^{u} \lambda^{t(i-1)\bar{s}\bar{n}} c_{(e-1)u+i,j}$$

for all $t = 0, \ldots, r-1; j = 0, \ldots, l-1$. We will use a subset of the parity-check equations with indices t of the form $t = wu$:

$$\lambda^{((p-1)\bar{s}+j_p)wu} \sum_{i=1}^{n} c_{(p-1)u+i,j} = - \sum_{e \neq p} \lambda^{((e-1)\bar{s}+j_e)wu} \sum_{i=1}^{u} c_{(e-1)u+i,j}$$

for all $j = 0, \ldots, l-1; w = 0, 1, \ldots, \bar{r}-1$, where we have used the fact that $\lambda^{\bar{s}n} = 1$. Denoting $\alpha = \lambda^u$ and summing these equations on $j_p = 0, 1, \ldots, \bar{s} - 1$, we obtain

the following set of conditions:

$$\sum_{j_p=0}^{\bar{s}-1} \alpha^{((p-1)\bar{s}+j_p)w} \sum_{i=1}^{u} c_{(p-1)u+i,j} = -\sum_{e\neq p} \alpha^{((e-1)\bar{s}+j_e)w} \sum_{j_p=0}^{\bar{s}-1} \sum_{i=1}^{u} c_{(e-1)u+i,j} \tag{3.10}$$

for all $w = 0, 1, \ldots, \bar{r}-1$ and all $j_{\bar{n}}, \ldots, j_{p+1}, j_{p-1}, \ldots, j_1$, where each of these values ranges over $\{0, 1, \ldots, \bar{s}-1\}$. Let $\mathcal{R} = \{q_1, \ldots, q_{\bar{d}}\}$ be the set of helper racks and let $[\bar{n}]\backslash\mathcal{R} = \{p, p_1, \ldots, p_{\bar{r}-\bar{s}}\}$. Then (3.10) can be written as follows:

$$\sum_{j_p=0}^{\bar{s}-1} \alpha^{((p-1)\bar{s}+j_p)w} \sum_{i=1}^{u} c_{(p-1)u+i,j} + \sum_{\substack{a\in[\bar{n}]\backslash\mathcal{R} \\ a\neq p}} \alpha^{((a-1)\bar{s}+j_a)w} \sum_{j_p=0}^{\bar{s}-1} \sum_{i=1}^{u} c_{(a-1)u+i,j}$$
$$= -\sum_{b\in\mathcal{R}} \alpha^{((b-1)\bar{s}+j_b)w} \sum_{j_p=0}^{\bar{s}-1} \sum_{i=1}^{u} c_{(b-1)u+i,j}. \tag{3.11}$$

In matrix form these equations are shown in (3.12) below, where

$$\sigma_{e,j(p,*)} := \sum_{j_p=0}^{\bar{s}-1} \sum_{i=1}^{u} c_{(e-1)u+i,j}, \quad e = 1, \ldots, \bar{n},$$

and j is as given above after (3.10).

$$\begin{bmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ \alpha^{\bar{s}(p-1)} & \cdots & \alpha^{\bar{s}(p-1)+\bar{s}-1} & \alpha^{\bar{s}(p_1-1)+j_{p_1}} & \cdots & \alpha^{\bar{s}(p_{\bar{r}-\bar{s}}-1)+j_{p_{\bar{r}-\bar{s}}}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (\alpha^{\bar{s}(p-1)})^{\bar{r}-1} & \cdots & (\alpha^{\bar{s}(p-1)+\bar{s}-1})^{\bar{r}-1} & (\alpha^{\bar{s}(p_1-1)+j_{p_1}})^{\bar{r}-1} & \cdots & (\alpha^{\bar{s}(p_{\bar{r}-\bar{s}}-1)+j_{p_{\bar{r}-\bar{s}}}})^{\bar{r}-1} \end{bmatrix} \begin{bmatrix} \sum_{i=1}^{u} c_{(p-1)u+i,j(p,0)} \\ \vdots \\ \sum_{i=1}^{u} c_{(p-1)u+i,j(p,\bar{s}-1)} \\ \sigma_{p_1,j(p,*)} \\ \vdots \\ \sigma_{p_{\bar{r}-\bar{s}},j(p,*)} \end{bmatrix}$$
$$= -\begin{bmatrix} 1 & \cdots & 1 \\ \alpha^{\bar{s}(q_1-1)+j_{q_1}} & \cdots & \alpha^{\bar{s}(q_{\bar{d}}-1)+j_{q_{\bar{d}}}} \\ \vdots & \vdots & \vdots \\ \alpha^{(\bar{s}(q_1-1)+j_{q_1})(\bar{r}-1)} & \cdots & \alpha^{(\bar{s}(q_{\bar{d}}-1)+j_{q_{\bar{d}}})(\bar{r}-1)} \end{bmatrix} \begin{bmatrix} \sigma_{q_1,j(p,*)} \\ \vdots \\ \sigma_{q_{\bar{d}},j(p,*)} \end{bmatrix} \tag{3.12}$$

We claim that Equations (3.12) suffice to recover one failed node in rack p.

Indeed, suppose that the $\bar{d}$-dimensional vector on the right-hand side of (3.12) is made available to the failed node by transmitting one symbol of F from each of the helper racks. Let us check that the matrix on the left-hand side is Vandermonde, i.e., that the defining elements in the second row are distinct. To see this, note that $\mathrm{ord}(\alpha) = \bar{s}\bar{n}$, and the maximum degree of α in the set $\{\alpha^{\bar{s}(e-1)+m}, m = 0, \ldots, \bar{s}-1; a = 1, \ldots, \bar{n}\}$ is

$$\bar{s}(\bar{n}-1) + \bar{s} - 1 < \bar{s}\bar{n}.$$

Moreover, each of the first $\bar{s}$ coordinates of the multiplier vector on the left-hand side of (3.12)

$$\Big(\sum_{i=1}^{u} c_{(p-1)u+i,j(p,0)}, \ldots, \sum_{i=1}^{u} c_{(p-1)u+i,j(p,\bar{s}-1)}\Big)^T$$

contains only one unknown term which corresponds to the failed node. Thus, if the values $c_{(p-1)u+i,j(p,*)}$ of all the functional local nodes are made available to the failed node (recall that this does not count toward the repair bandwidth), then system (3.12) can be solved to find the entries of the missing node. This calculation is repeated $\bar{s}^{\bar{n}-1}$ times for each assignment of the values $j_{\bar{n}}, \ldots, j_{p+1}, j_{p-1}, \ldots, j_1$, thereby completing the repair procedure.

Let us compute the inter-rack repair bandwidth of the described procedure. To repair the entries of the single failed node in the pth rack with indices in the subset $\{j(p,a), a = 0, 1, \ldots, \bar{s}-1\}$ we download one symbol of F from each of the $\bar{d}$ helper racks. There are $\bar{s}^{\bar{n}-1}$ subsets of the above form, and thus the total repair bandwidth is

$$\bar{d}\bar{s}^{\bar{n}-1} = \frac{\bar{d}\bar{l}}{\bar{s}},$$

proving the optimality claim of the code according to (3.2).

Finally let us prove that the code $\mathcal{C}$ is MDS. This is immediate upon observing that each subset of parity-check equations isolated by fixing the value of $j = 0, 1, \ldots, l-1$ defines an MDS code. To check this, observe that the set of rows of the parity-check matrix of $\mathcal{C}$ for a fixed value $j = (j_{\bar{n}}, \ldots, j_1)$ forms a set of parities of a generalized Reed-Solomon codes (i.e., each column is a set of powers of an element of F), and the defining row of this set of parities is shown below,

$$|\lambda^{j_1}, \lambda^{j_1+\bar{s}\bar{n}}, \ldots, \lambda^{j_1+(u-1)\bar{s}\bar{n}}|\lambda^{j_2+\bar{s}}, \lambda^{j_2+\bar{s}(1+\bar{n})}, \ldots, \lambda^{j_2+\bar{s}(1+(u-1)\bar{n})}|\ldots$$
$$|\lambda^{j_{\bar{n}}+\bar{s}(\bar{n}-1)}, \lambda^{j_{\bar{n}}+\bar{s}(2\bar{n}-1)}, \ldots, \lambda^{j_{\bar{n}}+\bar{s}(\bar{n}-1+(u-1)\bar{n})}| \quad (3.13)$$

where each group between the vertical bars corresponds to a fixed value of $s = 1, \ldots, \bar{n}$ in (3.9). It suffices to show that all these elements are distinct or that these groups do not overlap. Note that the largest power in (3.13) is

$$j_{\bar{n}} + \bar{s}(\bar{n} - 1 + (u-1)\bar{n}) \leqslant \bar{s} - 1 + u\bar{n}\bar{s} - \bar{s} < \bar{s}n = \mathrm{ord}(\lambda). \quad (3.14)$$

Now consider two groups and let their numbers be a and b, where $1 \leqslant b < a \leqslant \bar{n}$. Then the difference between the exponents of the first elements in the two groups is

$$(a-b)\bar{s} + (j_a - j_b) \geqslant 1$$

so the first elements are obviously distinct. Further, the exponents of the elements

in each of the groups are obtained by adding a multiple of $\bar{s}\bar{n}$ to the exponent of the first element, which together with (3.14) implies that the groups are disjoint. This shows that the code $\mathcal{C}$ is MDS, and the proof is complete. □

We remark that the repair procedure relies on a subset of the parity-check equations of the code $\mathcal{C}$. Namely, the only rows of the parity-check matrix that we use are the rows whose numbers are integer multiples of the size of the rack u. It suffices to use only these parities because the assumptions of the rack model are relaxed compared to the standard definition of regenerating codes. The remaining parities support the MDS property of the code $\mathcal{C}$ and do not contribute to the repair procedure.

In Sec. 3.4.2 we construct codes with somewhat better parameters than the codes given by Construction 3.3.1. Specifically, the smallest field size required for the code family in Sec. 3.4.2 is $n + \bar{s} - 1$ (as opposed to $\bar{s}n$), and the repair procedure accesses fewer symbols on the helper nodes than the procedure presented in the above proof. At the same time, the codes presented in this section have the *optimal update property*. Namely, a codeword of the code $\mathcal{C}$ can be viewed as an $l \times n$ array, and for a given row index $j \in \{1, \ldots, l-1\}$ the n symbols are encoded with a generalized RS code independently of the other rows. Thus, if some k symbols are taken as information symbols, then the change of one symbol in the data requires to change r parity symbols, which is also the smallest possible number [75]. At the same time, the codes in the family of Sec. 3.4.2 do not have optimal update, and are in this respect inferior to the present construction.

3.4 Low-access codes for the rack model

This section aims at constructing an optimal-repair MSR code for the rack model that accesses a reduced number of symbols on the nodes in the helper racks. Our presentation is formed of two parts. In the first part we construct an optimal-access MSR code for arbitrary repair degree $k \leqslant d \leqslant n-1$ *without assuming the rack model* of storage. The code has subpacketization $l = (d-k+1)^n$.

In the second part we present a modification of this construction for the rack model, attaining subpacketization $l = \bar{s}^{\bar{n}}$. Note that this value is smaller than the smallest node size of known constructions of OA codes for the homogeneous model, which is s^n [85].

3.4.1 Optimal-access MSR codes with arbitrary repair degree for homogeneous storage

In this section we present a family of OA codes for any repair degree $k \leqslant d \leqslant n-1$. Let $s = d-k+1$ and let $F, |F| \geqslant n+s-1$ be a finite field. Let $\lambda_0, \ldots, \lambda_{n-1}, \mu_1, \ldots, \mu_{s-1}$ be $n+s-1$ distinct elements of F. Let $i = (i_{n-1}, \ldots, i_0)$ be the s-ary representation of $i = 0, \ldots, l-1$ and (as before) let

$$i(a,b) = (i_{n-1}, \ldots, i_{a+1}, b, i_{a-1}, \ldots, i_0)$$

for $0 \leqslant a \leqslant n-1$ and $0 \leqslant b \leqslant s-1$. For brevity below we use the notation

$$\delta(i) := \mathbb{1}_{\{i=0\}}.$$

Construction 3.4.1. *Define an* $(n, k = n-r, l = s^n)$ *array code* $\mathcal{C} = \{\boldsymbol{c} = (c_{j,i})_{0\leqslant j\leqslant n-1; 0\leqslant i\leqslant l-1}\}$, *where the codeword* $\boldsymbol{c}$ *satisfies the following parity check equations over* F*:*

$$\sum_{j=0}^{n-1} \lambda_j^t c_{j,i} + \sum_{j=0}^{n-1} \delta(i_j) \sum_{p=1}^{s-1} \mu_p^t c_{j,i(j,p)} = 0, \quad i = 0, \ldots, l-1;\ t = 0, \ldots, r-1. \tag{3.15}$$

Since later in this section we rely on multiplicative structure of F*, we label the nodes* $0, \ldots, n-1$ *and not* $1, \ldots, n$ *as in Construction 3.3.1. In the next subsection we will also label the racks from* 0 *to* $\bar{n}-1$ *for the same reason.*

Theorem 29. *The code* $\mathcal{C}$ *defined in* (3.15) *is an optimal-access MDS array code.*

The proof will be omitted because in principle it can be obtained from the proof of Theorem 30 below upon taking the size of the rack $u = 1$. This is however not entirely immediate, and interested readers can consult the arXiv posting of a preprint of this chapter (arXiv:1901.04419, January 2019) which contains a complete and independent proof of Theorem 29.

3.4.2 Rack-aware MSR codes with low access

In this section we adapt the code family constructed in Sec. 3.4.1 for the rack-aware storage model. This result is obtained by adjusting the sub-packetization and

by carefully choosing the elements $\lambda_0, \ldots, \lambda_{n-1}$.

We aim to construct an (n, k, l) MDS array code over F, where $n = \bar{n}u$, and u is the size of the rack. Recall that $\bar{s} = \bar{d} - \bar{k} + 1$ where $\bar{k} \leqslant \bar{d} \leqslant \bar{n} - 1$, and $\bar{k} = \lfloor k/u \rfloor$. Let $|F| \geqslant n + \bar{s} - 1$ and $n|(|F| - 1)$. Let $\lambda \in F$ be an element of multiplicative order n, and let $\mu_1, \ldots, \mu_{\bar{s}-1}$ be $\bar{s} - 1$ distinct elements in $F \backslash \{\lambda^i \mid i = 0, \ldots, n-1\}$. For $j = 0, \ldots, n-1$, let us write $j = eu + g$ where $0 \leqslant e < \bar{n}$ and $0 \leqslant g < u$.

We construct an rack-aware low-access MSR code over F that can repair any single node from any $\bar{d}$ helper racks.

Construction 3.4.2. *Define an* $(n, k = n - r, l = \bar{s}^{\bar{n}})$ *array code* $\mathcal{C} = \{\boldsymbol{c} = (c_{j,i})_{0 \leqslant j \leqslant n-1; 0 \leqslant i \leqslant l-1}\}$ *by the following parity-check equations over* F*:*

$$\sum_{j=0}^{n-1} \lambda_j^t c_{j,i} + \sum_{j=0}^{n-1} \delta(i_e) \sum_{p=1}^{\bar{s}-1} \mu_p^t c_{j,i(e,p)} = 0, \tag{3.16}$$

where $\lambda_j = \lambda^{e+g\bar{n}}$, $i = 0, \ldots, l-1$ *and* $t = 0, \ldots, r-1$.

We will show that this code family supports optimal repair while accessing $l/\bar{s}$ symbols on each of the nodes in the helper racks, which is by a factor of $s/\bar{s} \approx u$ greater than the bound in Prop. 26. While these codes stop short of attaining the bound (3.4), they have lower access requirement than the codes given by Construction 3.3.1, which access all symbols of the helper nodes, i.e., $\bar{s}$ times more symbols than the current construction.

Theorem 30. *The code* $\mathcal{C}$ *defined in* (3.16) *is an optimal-repair MDS array code. The repair procedure accesses* $l/\bar{s}$ *symbols on each of the nodes in* $\bar{d}$ *helper racks.*

The repair scheme does not depend on the choice of the subset of $\bar{d}$ helper racks.

Proof. I. Optimal-access property. Suppose c_{j_1} is the failed node, where $j_1 = e_1 u + g_1$. Let $\mathcal{R}$ be the set of helper racks and let $\mathcal{J} = \{0, \ldots, \bar{n}-1\} \backslash \mathcal{R}$. We write this set as $\mathcal{J} = \{e_1, e_2, \ldots, e_{\bar{n}-\bar{d}}\}$. For a given $a, 1 \leqslant a \leqslant \bar{n} - \bar{d}$ we will need a-subsets of $\mathcal{J}$, which we denote by $\mathcal{J}_a$. We always assume that $e_1 \in \mathcal{J}_a$. As before, let $\mathcal{I} \subset \{0, 1, \ldots, l-1\}$ be the subset of indices such that $i_{e_1} = 0$; let

$$\mathcal{I}_1 = \{i = (i_{\bar{n}-1}, \ldots, i_0) \in \{0, \ldots, l-1\} \mid i_{e_1} = 0; i_e \neq 0, e \in \mathcal{J} \backslash \mathcal{J}_1\}$$

and define

$$\mathcal{I}_a = \bigcup_{\mathcal{J}_a \subseteq \mathcal{J}} \mathcal{I}(\mathcal{J}_a), \quad a = 2, \ldots, \bar{n} - \bar{d},$$

where

$$\mathcal{I}(\mathcal{J}_a) = \{i = (i_{\bar{n}-1}, \ldots, i_0) \in \{0, \ldots, l-1\} \mid i_e = 0, e \in \mathcal{J}_a; i_e \neq 0, e \in \mathcal{J} \backslash \mathcal{J}_a\}.$$

Recall that $\bar{r} = \bar{n} - \bar{k}$. We will use the parity check equations corresponding to $i \in \mathcal{I}$ and all powers $t = uw, w = 0, \ldots, \bar{r} - 1$ to repair c_{j_1}. To show that the repair is possible, we argue by induction on $a = 1, \ldots, \bar{n} - \bar{d}$.

To prove the induction basis, we show that it is possible to recover the values $\{c_{j_1, i(e_1, p)} \mid p = 0, \ldots, \bar{s} - 1\}$ and $\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid e \in \mathcal{J} \backslash \mathcal{J}_1\}$ for every $i \in \mathcal{I}_1$ from the

helper racks $\mathcal{R}$. From (3.16), for $i \in \mathcal{I}_1$, we have

$$\sum_{e\in\mathcal{J}}\sum_{g=0}^{u-1}\lambda^t_{eu+g}c_{eu+g,i} + \sum_{g=0}^{u-1}\sum_{p=1}^{\bar{s}-1}\mu^t_p c_{e_1u+g,i(e_1,p)} \\ = -\sum_{e\in\mathcal{R}}\sum_{g=0}^{u-1}\Big(\lambda^t_{eu+g}c_{eu+g,i} + \delta(i_e)\sum_{p=1}^{\bar{s}-1}\mu^t_p c_{eu+g,i(e,p)}\Big).$$

Using $t = uw$, $\lambda_{eu+g} = \lambda^{e+g\bar{n}}$, and $\lambda^{\bar{n}u} = 1$, we obtain

$$\sum_{e\in\mathcal{J}}\lambda^{euw}\sum_{g=0}^{u-1}c_{eu+g,i} + \sum_{p=1}^{\bar{s}-1}\mu^{uw}_p\sum_{g=0}^{u-1}c_{e_1u+g,i(e_1,p)} \\ = -\sum_{e\in\mathcal{R}}\Big(\lambda^{euw}\sum_{g=0}^{u-1}c_{eu+g,i} + \delta(i_e)\sum_{p=1}^{\bar{s}-1}\mu^{uw}_p\sum_{g=0}^{u-1}c_{eu+g,i(e,p)}\Big), \quad (3.17)$$

$i \in \mathcal{I}_1, w = 0,\dots,\bar{r}-1$. To shorten our notation, denote the right-hand side of (3.17) by $\sigma_{i,w}(\mathcal{J}_1)$ and let

$$\pi_{i,e} := \sum_{g=0}^{u-1} c_{eu+g,i}.$$

Note that the value of $\sigma_{i,w}(\mathcal{J}_1)$ only depends on the helper racks. For $i = 1,\dots,\bar{n}-\bar{d}$ define $\alpha_i := \lambda^{e_iu}$. Let us write equations (3.17) for all $w = 0,\dots,\bar{r}-1$ in matrix

form:

$$\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 & \cdots & 1 \\ \alpha_1 & \mu_1 & \cdots & \mu_{\bar{s}-1} & \alpha_2 & \cdots & \alpha_{\bar{n}-\bar{d}} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{\bar{r}-1} & \mu_1^{\bar{r}-1} & \cdots & \mu_{\bar{s}-1}^{\bar{r}-1} & \alpha_2^{\bar{r}-1} & \cdots & \alpha_{\bar{n}-\bar{d}}^{\bar{r}-1} \end{bmatrix} \begin{bmatrix} \pi_{i,e_1} \\ \pi_{i(e_1,1),e_1} \\ \vdots \\ \pi_{i(e_1,\bar{s}-1),e_1} \\ \pi_{i,e_2} \\ \vdots \\ \pi_{i,e_{\bar{n}-\bar{d}}} \end{bmatrix} = \begin{bmatrix} \sigma_{i,0}(\mathcal{J}_1) \\ \sigma_{i,1}(\mathcal{J}_1) \\ \vdots \\ \sigma_{i,\bar{r}-1}(\mathcal{J}_1) \end{bmatrix}. \quad (3.18)$$

Observe that the matrix on the left-hand side of (3.18) is invertible. Therefore, the values $\{c_{j_1,i(j_1,p)} \mid p = 0,\dots,\bar{s}-1\}$ and $\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid e \in \mathcal{J}\backslash\mathcal{J}_1\}$ can be found from the values $\{\sigma_{i,w}(\mathcal{J}_1) \mid w = 0,\dots,\bar{r}-1\}$ and the local nodes $\{c_{e_1u+g} \mid g \neq g_1\}$ for every $i \in \mathcal{I}_1$. This completes the proof of the induction basis.

Now let us fix $a \in \{2,\dots,\bar{n}-\bar{d}\}$ and suppose that we have recovered the values $\{c_{j_1,i(e_1,p)} \mid p = 0,\dots,\bar{s}-1\}$ and $\left\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid e \in \mathcal{J}\backslash\mathcal{J}_1\right\}$, $i \in \mathcal{I}_{a'}$; $1 \leqslant a' \leqslant a-1$ from the information downloaded from the helper racks $\mathcal{R}$.

Fix a subset $\mathcal{J}_a, |\mathcal{J}_a| = a$, and let $i \in \mathcal{I}(\mathcal{J}_a)$. From (3.16), we have

$$\begin{aligned} \sum_{e\in\mathcal{J}}\sum_{g=0}^{u-1} \lambda_{eu+g}^t c_{eu+g,i} + \sum_{e\in\mathcal{J}_a}\sum_{g=0}^{u-1}\sum_{p=1}^{\bar{s}-1} \mu_p^t c_{eu+g,i(e,p)} \\ = \sum_{e\in\mathcal{J}}\sum_{g=0}^{u-1} \lambda_{eu+g}^t c_{eu+g,i} + \sum_{p=1}^{\bar{s}-1} \mu_p^t \sum_{e\in\mathcal{J}_a}\sum_{g=0}^{u-1} c_{eu+g,i(e,p)} \\ = -\sum_{e\in\mathcal{R}} \Big(\sum_{g=0}^{u-1} \lambda_{eu+g}^t c_{eu+g,i} + \delta(i_e) \sum_{p=1}^{\bar{s}-1} \mu_p^t \sum_{g=0}^{u-1} c_{eu+g,i(e,p)} \Big). \end{aligned} \quad (3.19)$$

Using $t = uw$, $\lambda_{eu+g} = \lambda^{e+g\bar{n}}$, and $\lambda^{\bar{n}u} = 1$, we obtain

$$\sum_{e\in\mathcal{J}} \lambda^{euw} \sum_{g=0}^{u-1} c_{eu+g,i} + \sum_{p=1}^{\bar{s}-1} \mu_p^{uw} \sum_{e\in\mathcal{J}_a} \sum_{g=0}^{u-1} c_{e_1 u+g,i(e_1,p)}$$
$$= -\sum_{e\in\mathcal{R}} \left(\lambda^{euw} \sum_{g=0}^{u-1} c_{eu+g,i} + \delta(i_e) \sum_{p=1}^{\bar{s}-1} \mu_p^{uw} \sum_{g=0}^{u-1} c_{eu+g,i(e,p)} \right). \quad (3.20)$$

Again for notational convenience denote the right-hand side of (3.20) by $\sigma_{i,w}(\mathcal{J}_a)$ and let

$$\rho_{i,p} := \sum_{e\in\mathcal{J}_a} \sum_{g=0}^{u-1} c_{eu+g,i(e,p)}.$$

Note that the value of $\sigma_{i,w}(\mathcal{J}_a)$ depends only on the information in the helper racks.

Let us write Equations (3.20) for all $w = 0, \ldots, \bar{r}-1$ in matrix form:

$$\begin{bmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ \mu_1 & \cdots & \mu_{\bar{s}-1} & \alpha_1 & \cdots & \alpha_{\bar{n}-\bar{d}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mu_1^{\bar{r}-1} & \cdots & \mu_{\bar{s}-1}^{\bar{r}-1} & \alpha_1^{\bar{r}-1} & \cdots & \alpha_{\bar{n}-\bar{d}}^{\bar{r}-1} \end{bmatrix} \begin{bmatrix} \rho_{i,1} \\ \vdots \\ \rho_{i,\bar{s}-1} \\ \pi_{i,e_1} \\ \vdots \\ \pi_{i,e_{\bar{n}-\bar{d}}} \end{bmatrix} = \begin{bmatrix} \sigma_{i,0}(\mathcal{J}_a) \\ \sigma_{i,1}(\mathcal{J}_a) \\ \vdots \\ \sigma_{i,\bar{r}-1}(\mathcal{J}_a) \end{bmatrix}. \quad (3.21)$$

Therefore, for any $\mathcal{J}_a \subseteq \mathcal{J}$ and every $i \in \mathcal{I}(\mathcal{J}_a)$, the values $\{\rho_{i,p} \mid p = 1, \ldots, \bar{s}-1\}$ and $\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid e \in \mathcal{J}\}$ can be found from the values $\{\sigma_{i,w}(\mathcal{J}_a) \mid w = 0, \ldots, \bar{r}-1\}$. It follows that we can recover the values $\{\rho_{i,p} \mid p = 1, \ldots, \bar{s}-1\}$ and $\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid e \in \mathcal{J}\}$ for all $i \in \mathcal{I}_a$.

Note that for $i \in \mathcal{I}(\mathcal{J}_a)$, $e \in \mathcal{J}_a \backslash \mathcal{J}_1$, and for $p \neq 0$, we have $i(e,p) \in \mathcal{I}_{a-1}$. By the induction hypothesis, we have recovered the values $\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid i \in \mathcal{I}_{a-1}; e \in \mathcal{J} \backslash \mathcal{J}_1\}$, and therefore, we know the values $\{\sum_{g=0}^{u-1} c_{eu+g,i(e,p)} \mid j \in \mathcal{J}_a \backslash \mathcal{J}_1, p \neq 0\}$ for each $i \in \mathcal{I}_a$. With these values and $\{\rho(i,p) \mid i \in \mathcal{I}_a, p = 1, \ldots, s-1\}$, we can obtain the values $\{\sum_{g=0}^{u-1} c_{e_1 u+g,i} \mid p = 1, \ldots, s-1\}$. Since the values of local nodes $\{c_{e_1 u+g,i} \mid g \neq g_1\}$ are available, we can further recover the value $c_{j_1,i}$.

Thus, we can obtain the values $\{c_{j_1,i(e_1,p)} \mid p = 0, \ldots, \bar{s}-1\}$ and $\{\sum_{g=0}^{u-1} c_{e_1 u+g,i} \mid e \in \mathcal{J} \backslash \mathcal{J}_1\}$ for every $i \in \mathcal{I}_a$. It follows that we can recover these values for every $i \in \mathcal{I}_a$ and $1 \leqslant a \leqslant \bar{n} - \bar{d}$ from the helper racks $\mathcal{R}$. In conclusion, we can recover the values $\{c_{j_1,i(e_1,p)} \mid i \in \mathcal{I}, p = 0, \ldots, \bar{s}-1\} = \{c_{j_1,i} \mid i = 0, \ldots, l-1\}$ from the information obtained from the helper racks in $\mathcal{R}$.

Now let us count the number of symbols we access in each helper rack. It is clear from the definition of $\sigma_{i,w}(\mathcal{J}_a)$ (see (3.20)) that we need to access the symbols $\{c_{eu+g,i} \mid 0 \leqslant g < u, i \in \mathcal{I}\}$ for each $e \in \mathcal{R}$. In other words, we need to access $\bar{s}^{\bar{n}-1} = l/\bar{s}$ symbols on each node in the helper racks; thus, the total number of accessed symbols equals $\bar{d}ul/\bar{s}$. Moreover, the set of symbols we access in each helper rack depends on index of the host rack but not the index of the helper rack.

Note also that the symbols downloaded to the rack e_1 from any helper rack $e \in \mathcal{R}$ form the subset $\{\sum_{g=0}^{u-1} c_{eu+g,i} \mid i \in \mathcal{I}\}$. Thus, the total amount of information downloaded for the purposes of repair equals

$$\bar{d}|\mathcal{I}| = \bar{d}\bar{s}^{\bar{n}-1} = \frac{\bar{d}l}{\bar{d} - \bar{k} + 1}.$$

This is the smallest possible number according to the bound (3.2), and thus the codes support optimal repair.

II. MDS PROPERTY. We will show that the contents of any $n-r$ nodes suffices to find the values of the remaining r nodes.

Let $\mathcal{K} = \{j_1, \ldots, j_r\} \subseteq \{0, \ldots, n-1\}$ be the set of r nodes to be recovered from the set of $n-r$ nodes in $[0, n-1]\backslash\mathcal{K}$. Let us write $j_b = e_b u + g_b$ where $0 \leqslant g_b < u-1$ for $b = 1, \ldots, r$.

Let $\mathcal{J}$ be the set of distinct $e_b, b = 1 \ldots, r$. For $1 \leqslant a \leqslant |\mathcal{J}|$, let $\mathcal{J}_a \subseteq \mathcal{J}$ be such that $|\mathcal{J}_a| = a$.

Let $\mathcal{I}_0 = \{i = (i_{\bar{n}-1}, \ldots, i_0) \in \{0, \ldots, l-1\} \mid i_e \neq 0, e \in \mathcal{J}\}$. For $1 \leqslant a \leqslant |\mathcal{J}|$ and $\mathcal{J}_a \subseteq \mathcal{J}$, let $\mathcal{I}(\mathcal{J}_a) = \{i = (i_{\bar{n}-1}, \ldots, i_0) \in \{0, \ldots, l-1\} \mid i_e = 0, e \in \mathcal{J}_a; i_{e'} \neq 0, e' \in \mathcal{J}\backslash\mathcal{J}_a\}$. Let $\mathcal{I}(a) = \bigcup_{\mathcal{J}_a \subseteq \mathcal{J}} \mathcal{I}(\mathcal{J}_a)$ where $1 \leqslant a \leqslant |\mathcal{J}|$. Observe that the sets $I_a, 0 \leqslant a \leqslant |\mathcal{J}|$ partition the set $\{0, 1, \ldots, l-1\}$.

We will prove by induction that we can recover the nodes in $\mathcal{J}$ from the nodes in $\{0, 1, \ldots, n-1\}\backslash\mathcal{J}$. First, let us establish the induction basis, i.e., we can recover the values $\{c_{j,i} \mid j \in \mathcal{J}\}$ for every $i \in \mathcal{I}_0$ from the nodes $\{c_j \mid j \in \mathcal{J}^c\}$. From (3.16), for $i \in \mathcal{I}_0$, we have

$$\sum_{j \in \mathcal{J}} \lambda_j^t c_{j,i} = -\sum_{j \in \mathcal{J}^c} \left(\lambda_j^t c_{j,i} + \delta(i_e) \sum_{p=1}^{\bar{s}-1} \mu_p^t c_{j,i(e,p)} \right). \tag{3.22}$$

To simplify notation, denote the right-hand side of (3.22) by $\sigma_{i,t} = \sigma_{i,t}(\varnothing)$. Note that the value of $\sigma_{i,t}$ only depends on the nodes $\{c_j \mid j \in \mathcal{J}^c\}$. Writing (3.22)

in matrix form, we have

$$\begin{bmatrix} 1 & \cdots & 1 \\ \lambda_{j_1} & \cdots & \lambda_{j_r} \\ \vdots & \ddots & \vdots \\ \lambda_{j_1}^{r-1} & \cdots & \lambda_{j_r}^{r-1} \end{bmatrix} \begin{bmatrix} c_{j_1,i} \\ c_{j_2,i} \\ \vdots \\ c_{j_r,i} \end{bmatrix} = \begin{bmatrix} \sigma_{i,0} \\ \sigma_{i,1} \\ \vdots \\ \sigma_{i,r-1} \end{bmatrix}. \tag{3.23}$$

Therefore, the values $\{c_{j,i} \mid j \in \mathcal{J}\}$ can be calculated from the values $\{\sigma_{i,t} \mid t = 0, \ldots, r-1\}$ for every $i \in \mathcal{I}_0$.

Now let us establish the induction step. Suppose we recover the values $\{c_{j,i} \mid j \in \mathcal{J}\}$ for every $i \in \mathcal{I}_{a'}$ and $0 \leqslant a' \leqslant a-1$ from the nodes $\{c_j \mid j \in \mathcal{J}^c\}$, where $1 \leqslant a \leqslant |\mathcal{J}|$.

Now let us fix a set $\mathcal{J}_a \subseteq \mathcal{J}$ and let $i \in \mathcal{I}(\mathcal{J}_a)$. From (3.16), we have

$$\begin{aligned} \sum_{j \in \mathcal{J}} \lambda_j^t c_{j,i} &= -\sum_{p=1}^{\bar{s}-1} \mu_p^t \sum_{j \in \mathcal{J} \colon e \in \mathcal{J}_a} c_{j,i(e,p)} - \sum_{j \in \mathcal{J}^c} \left(\lambda_j^t c_{j,i} + \delta(i_e) \sum_{p=1}^{\bar{s}-1} \mu_p^t c_{j,i(e,p)} \right) \\ &=: -\rho'_{i,t} - \sigma_{i,t}(\mathcal{J}_a), \end{aligned} \tag{3.24}$$

where the last line serves to introduce the shorthand notation. Note that we know the values $\{\sigma_{i,t}(\mathcal{J}_a) \mid t = 0, \ldots, r-1\}$ since the value $\sigma_{i,t}(\mathcal{J}_a)$ only depends on the nodes $\{c_j \mid j \in \mathcal{J}^c\}$. Furthermore, we also know the values $\{\rho'_{i,t} \mid t = 0, \ldots, r-1\}$. Indeed, for $i \in \mathcal{I}(\mathcal{J}_a)$, $e \in \mathcal{J}_a$, and $p \neq 0$, we have $i(e,p) \in \mathcal{I}_{a-1}$. By the induction hypothesis, we have recovered the values $\{c_{j,i} \mid i \in \mathcal{I}_{a-1}, j \in \mathcal{J}\}$, and therefore, we know the values $\{c_{j,i(e,p)} \mid j \in \mathcal{J} \colon e \in \mathcal{J}_a, p \neq 0\}$ for each $i \in \mathcal{I}_a$. It follows that we

know the values $\{\rho'_{i,t} \mid t = 0, \ldots, r-1\}$. Writing (3.24) in matrix form, we have

$$\begin{bmatrix} 1 & \cdots & 1 \\ \lambda_{j_1} & \cdots & \lambda_{j_r} \\ \vdots & \ddots & \vdots \\ \lambda_{j_1}^{r-1} & \cdots & \lambda_{j_r}^{r-1} \end{bmatrix} \begin{bmatrix} c_{j_1,i} \\ c_{j_2,i} \\ \vdots \\ c_{j_r,i} \end{bmatrix} = \begin{bmatrix} \rho'_{i,0} + \sigma_{i,0}(\mathcal{J}_a) \\ \rho'_{i,1} + \sigma_{i,1}(\mathcal{J}_a) \\ \vdots \\ \rho'_{i,r-1} + \sigma_{i,r-1}(\mathcal{J}_a) \end{bmatrix}. \tag{3.25}$$

Therefore, the values $\{c_{j,i} \mid j \in \mathcal{J}\}$ can be recovered for every $i \in \mathcal{I}(\mathcal{J}_a)$ and $\mathcal{J}_a \subseteq \mathcal{J}$. It follows that we can recover the values $\{c_{j,i} \mid j \in \mathcal{J}\}$ for every $i \in \mathcal{I}_a$. Thus, all the values $\{c_{j,i} \mid j \in \mathcal{J}, i \in \mathcal{I}_a, 0 \leqslant a \leqslant |\mathcal{J}|\} = \{c_{j,i} \mid j \in \mathcal{J}, i \in \{0, \ldots, l-1\}$ can be recovered from the nodes $\{c_j \mid j \in \mathcal{J}^c\}$.

Since $\mathcal{J}$ is arbitrary, we conclude that any $n - r$ nodes can recover the entire codeword, i.e., the code is MDS. □

3.5 A construction of Reed-Solomon codes with optimal repair

In this section we present a family of scalar MDS codes that support optimal repair of a single node from an arbitrary subset of $\bar{d}$ helper racks. We still use the same notation as in the previous parts of this chapter. As noted earlier, the construction is a modification of the RS code family in [77]. The new element of the construction is the idea of coupling the code family of [77] and the multiplicative structure that matches the grouping of the nodes into racks. This latter part is similar to the idea of Sec. 3.3.

3.5.1 Rack-aware RS codes with optimal repair

Let q be a power of a prime, let u be the size of the rack, and suppose that $u|(q-1)$. Let $k = \bar{k}u + v, 0 \leqslant v \leqslant u-1$, $\bar{s} = \bar{d} - \bar{k} + 1$. Let $p_i, i = 1, \ldots, \bar{n}$ be distinct primes such that $p_i \equiv 1 \bmod \bar{s}$ and $p_i > u$ for $i = 1, \ldots, \bar{n}$; for instance, we can take the *smallest* $\bar{n}$ primes with these properties. For $i = 1, \ldots, \bar{n}$ let λ_i be an element of degree p_i over $\mathbb{F}_q$. Let

$$F_i := \mathbb{F}_q(\lambda_j, j \in \{1, \ldots, \bar{n}\} \backslash \{i\}),\ i = 1, \ldots, \bar{n}$$

$$\mathbb{F} := \mathbb{F}_q(\lambda_1, \ldots, \lambda_{\bar{n}}).$$

Let $\mathbb{K}$ be an extension of $\mathbb{F}$ of degree $\bar{s}$ and let $\mu \in \mathbb{K}$ be a generating element of $\mathbb{K}$ over $\mathbb{F}$. Thus, for any $i = 1, \ldots, \bar{n}$ we have the chain of inclusions

$$\mathbb{F}_q \subset F_i \subset \mathbb{K};$$

so $\mathbb{K}$ is the l-th degree extension of $\mathbb{F}_q$, where $l = [\mathbb{K} : \mathbb{F}_q] = \bar{s} \prod_{m=1}^{\bar{n}} p_m$.

Further, let $\lambda \in \mathbb{F}_q$ be an element of multiplicative order u. Consider the set of elements

$$\lambda_{ij} = \lambda_i \lambda^{j-1}, i = 1, \ldots, \bar{n}; j = 1, \ldots, u.$$

Consider an RS code $\mathcal{C} = RS_{\mathbb{K}}(n, k, \Omega)$ where the set of evaluation points Ω is as follows:

$$\Omega = \bigcup_{i=1}^{\bar{n}} \Omega_i, \text{ where } \Omega_i = \{\lambda_{ij}, j = 1, \ldots, u\}.$$

A codeword of $\mathcal{C}$ has the form $c = (c_1, c_2, \ldots, c_n)$, where the coordinate $c_m, m = (i-1)u + j, 1 \leqslant i \leqslant \bar{n}; 1 \leqslant j \leqslant u$ corresponds to the evaluation point λ_{ij}.

To describe the repair procedure, we will need the following easy modification of Lemma 1 of [77].

Lemma 31. *For $i \in \{1, \ldots, \bar{n}\}$, there exists subspace S_i of $\mathbb{K}$ such that*

$$\dim_{F_i} S_i = p_i, \quad S_i + S_i\lambda_i^u + \cdots + S_i\lambda_i^{u(\bar{s}-1)} = \mathbb{K} \tag{3.26}$$

where $S_i\beta = \{\gamma\beta, \gamma \in S_i\}$ and the operation $+$ is the Minkowski sum of sets, $T_1 + T_2 := \{\gamma_1 + \gamma_2 : \gamma_1 \in T_1, \gamma_2 \in T_2\}$.

Proof. The space S_i is constructed as follows. Define the following vector spaces over F_i:

$$S_i^{(1)} = \mathrm{Span}_{F_i}(\mu^t \lambda_i^{t+e\bar{s}}, t = 0, 1, \ldots, \bar{s}-1; e = 0, 1, \ldots, \tfrac{p_i-1}{\bar{s}} - 1)$$

$$S_i^{(2)} = \mathrm{Span}_{F_i}\Big(\sum_{t=0}^{\bar{s}-1} \mu^t \lambda_i^{p_i-1}\Big)$$

and take

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

Now the proof of [77, Lemma 1] can be followed step by step, using the fact that $\{1, \lambda_i^u, \ldots, (\lambda_i^u)^{p_i-1}\}$ forms a basis for $\mathbb{F}$ over F_i, and we do not repeat it here. □

The main result of this section is given in the following proposition.

Proposition 32. *The code $\mathcal{C}$ supports optimal repair of a single failed node in any*

rack from any $\bar{d}$ helper racks.

The proof follows the scheme in [77] which is itself an implementation of the framework for repair of RS codes proposed in [28].

Proof. Let

$$c = ((c_{(i-1)u+j})_{1 \leqslant i \leqslant \bar{n}; 1 \leqslant j \leqslant u})$$

be a codeword of $\mathcal{C}$. Suppose that $c_{(i^*-1)u+j^*}$ is the failed node, i.e., the index of the host rack is $i^*, 1 \leqslant i^* \leqslant n$, and the index of the failed node in this rack is $j^*, 1 \leqslant j^* \leqslant u$. Denote by $\mathcal{R} \subseteq \{1, \ldots, \bar{n}\} \backslash \{i^*\}, |\mathcal{R}| = \bar{d}$ the set of helper racks. The repair relies on the information downloaded from all the nodes in $\mathcal{R}$ and the functional nodes in the host rack. Define the annihilator polynomial of the set of locators of all the nodes in $\mathcal{R}$:

$$h(x) = \prod_{\substack{i \in \{1,\ldots,\bar{n}\} \backslash (\mathcal{R} \cup \{i^*\}), \\ 1 \leqslant j \leqslant u}} (x - \lambda_{ij}). \tag{3.27}$$

Let $t = uw$, where $w = 0, \ldots, \bar{s} - 1$. Since

$$\deg x^t h(x) \leqslant (\bar{s} - 1)u + (\bar{n} - \bar{d} - 1)u = (\bar{r} - 1)u < \bar{r}u - v = n - k \tag{3.28}$$

evaluations of the polynomials $x^t h(x)$ are contained in the dual code $\mathcal{C}^\perp$.

Since $\mathcal{C}^\perp$ itself is a (generalized) RS code, there is a vector $a = (a_1, \ldots, a_n) \in (\mathbb{K}^*)^n$ such that any codeword of $\mathcal{C}^\perp$ has the form $(a_{ij} f(\lambda_{ij}))_{1 \leqslant i \leqslant \bar{n}; 1 \leqslant j \leqslant u}$, where $f \in \mathbb{K}[x]$ is a polynomial of degree $\leqslant r - 1$. Thus, by (3.28), we have the vec-

tor $(a_1\lambda_{11}^t h(\lambda_{11}), \ldots, a_n\lambda_{\bar{n},u}^t h(\lambda_{\bar{n},u})) \in \mathcal{C}^\perp$, so the inner product of this vector and the codeword c is zero. In other words, we have

$$\sum_{j=1}^{u} a_{(i^*-1)u+j}\lambda_{i^*j}^t h(\lambda_{i^*j})c_{(i^*-1)u+j} = -\sum_{\substack{i=1,\\ i\neq i^*}}^{\bar{n}}\sum_{j=1}^{u} a_{(i-1)u+j}\lambda_{ij}^t h(\lambda_{ij})c_{(i-1)u+j}.$$

Let S_{i^*} be the subspace defined in Lemma 31 and let $\{e_1, \ldots, e_{p_{i^*}}\}$ be a basis of S_{i^*} over F_{i^*}. Then for $m = 1, \ldots, p_{i^*}$, we have

$$\sum_{j=1}^{u} e_m a_{(i^*-1)u+j}\lambda_{i^*j}^t h(\lambda_{i^*j})c_{(i^*-1)u+j} = -\sum_{\substack{i=1,\\ i\neq i^*}}^{\bar{n}}\sum_{j=1}^{u} e_m a_{(i-1)u+j}\lambda_{ij}^t h(\lambda_{ij})c_{(i-1)u+j}.$$

Evaluating the trace from $\mathbb{K}$ to F_{i^*} on both sides of the last equation, we obtain

$$\begin{aligned}
&\mathrm{tr}_{\mathbb{K}/F_{i^*}}\Big(\sum_{j=1}^{u} e_m a_{(i^*-1)u+j}\lambda_{i^*j}^t h(\lambda_{i^*j})c_{(i^*-1)u+j}\Big)\\
&\qquad = -\sum_{\substack{i=1,\\ i\neq i^*}}^{\bar{n}}\sum_{j=1}^{u} \lambda_{ij}^t h(\lambda_{ij})\mathrm{tr}_{\mathbb{K}/F_{i^*}}\left(e_m a_{(i-1)u+j}c_{(i-1)u+j}\right)\\
&\qquad = -\sum_{i\in\mathcal{R}}\sum_{j=1}^{u} \lambda_{ij}^t h(\lambda_{ij})\mathrm{tr}_{\mathbb{K}/F_{i^*}}\left(e_m a_{(i-1)u+j}c_{(i-1)u+j}\right), \qquad (3.29)
\end{aligned}$$

where we used (3.27), the fact that $\lambda_{ij} \in F_{i^*}$ for all $i \neq i^*$, and where $t = uw, w = 0, \ldots, \bar{s}-1$ and $m = 1, \ldots, p_{i^*}$.

Recall that $\lambda_{ij} = \lambda_i\lambda^{j-1}$ and $\lambda^u = 1$. From (3.29) we have

$$\begin{aligned}
&\mathrm{tr}_{\mathbb{K}/F_{i^*}}\Big(e_m\lambda_{i^*}^{uw}\sum_{j=1}^{u} a_{(i^*-1)u+j}h(\lambda_{i^*j})c_{(i^*-1)u+j}\Big)\\
&\qquad = -\sum_{i\in\mathcal{R}}\lambda_i^{uw}\sum_{j=1}^{u} h(\lambda_{ij})\mathrm{tr}_{\mathbb{K}/F_{i^*}}\left(e_m a_{(i-1)u+j}c_{(i-1)u+j}\right), \qquad (3.30)
\end{aligned}$$

where the parameters t, m are as above. By (3.26) in Lemma 31 and the definition of the set $\{e_m\}$, the set $\{e_m\lambda_{i*}^{uw} \mid 1 \leqslant m \leqslant p_{i*}, 0 \leqslant w \leqslant \bar{s}-1\}$ forms a basis for $\mathbb{K}$ over F_{i*}. Therefore, the mapping

$$\beta \mapsto \mathrm{tr}_{\mathbb{K}/F_{i*}}(e_m\lambda_{i*}^{uw}\beta), 1 \leqslant m \leqslant p_{i*}, 0 \leqslant w \leqslant \bar{s}-1 \tag{3.31}$$

is a bijection.

The repair procedure is accomplished as follows. For every $i \in \mathcal{R}$, the elements

$$\sum_{j=1}^{u} h(\lambda_{ij})\mathrm{tr}_{\mathbb{K}/F_{i*}}(e_m a_{(i-1)u+j}c_{(i-1)u+j}), m = 1, \ldots, p_{i*} \tag{3.32}$$

are downloaded from helper rack i. By (3.30) this enables us to find the elements

$$\mathrm{tr}_{\mathbb{K}/F_{i*}}\left(e_m\lambda_{i*}^{uw}\sum_{j=1}^{u} a_{(i^*-1)u+j}h(\lambda_{i^*j})c_{(i^*-1)u+j}\right),$$

$m = 1, \ldots, p_{i^*}$. Next, on account of (3.31) we can find the value of the sum

$$\sum_{j=1}^{u} a_{(i^*-1)u+j}h(\lambda_{i^*j})c_{(i^*-1)u+j}.$$

Finally, since the values of the coordinates $c_{(i^*-1)u+j}, j \neq j^*$ stored on the functional nodes in the host rack i^* are available and the entires of the vector a are nonzero, we can find $c_{(i^*-1)u+j^*}$, completing the repair.

The number of field symbols of F_{i*} (3.32) transmitted from the helper racks to the host rack equals $\bar{d}p_{i*}$. Therefore, we conclude that the repair bandwidth of

$\mathcal{C}$ is

$$\frac{\bar{d}p_{i*}}{[\mathbb{K} : F_{i*}]}l = \frac{\bar{d}l}{\bar{s}}. \tag{3.33}$$

This meets the bound (3.2) with equality, and proves the claim of optimal repair. □

3.5.2 Rack-aware RS codes with optimal error correction and low access

The repair schemes and constructions of the RS code families with optimal error correction and optimal access for the homogeneous storage model in Chapter 2 can also be modified by incorporating the ideas of constructing rack-aware MSR codes we presented in this chapter, resulting in rack-aware RS codes with optimal error correction and low access. Below we briefly sketch the basic ideas.

First, we need a bound for the repair bandwidth when there are unreliable helper racks that provide erroneous information. Suppose there are $\bar{e} \geqslant 0$ unreliable helper racks such that $\bar{d} \geqslant \bar{k} + 2\bar{e}$. Let $\beta_u(\bar{d}, \bar{e})$ be the minimum number of symbols needed to repair a failed node in the host rack from $\bar{d}$ helper racks (and from the local nodes in the host rack), of which $\bar{e}$ racks provide incorrect information. The bound (3.2) can be generalized by extending the argument in Appendix B.1 to show that for $\bar{d} \geqslant \bar{k} + 2\bar{e}$,

$$\beta_u(\bar{d}, \bar{e}) \geqslant \frac{\bar{d}l}{\bar{d} - 2\bar{e} - \bar{k} + 1}. \tag{3.34}$$

Next, we present a repair scheme that enables error correction for the rack-

aware RS code construction we saw in this chapter and that achieves the above lower bound for the repair bandwidth. Indeed, with a moment of retrospection, this goal can be accomplished as follows. Instead of using n minimal polynomials for each of the n evaluation points of the code as in Sec 2.3, under the setting of the rack-aware storage model, one only needs $\bar{n}$ minimal polynomials for the evaluation points $\lambda_i, i = 1, \ldots, \bar{n}$. Then one can establish a proposition analogous to Prop. 4 by repeating arguments similar to those in Sec. 2.3 and applying the techniques we used in this chapter. More precisely, one can show that the information provided by the helper racks can be transformed by linear maps into vectors contained in certain $(\bar{n} - 1, \bar{d} - 2\bar{e})$ MDS code. Thus, as long as there are no more than $\bar{e}$ helper racks that provide erroneous information, one can repair the failed node with the smallest possible bandwidth given by (3.34).

Lastly, we can also construct a family of rack-aware RS codes that support low access cost (and optimal error correction). Similarly to Sec. 2.4, we need additional structures of the underlying finite field to reduce the access cost. This can be done by a sequence of algebraic extensions of $\mathbb{F}_q$ similar to (2.29) and (2.30). In short, let μ_i an element of degree $\bar{s}^i$ over $\mathbb{F}_q$ for $i = 1, \ldots, \bar{n}$ and define $\mathbb{E} := \mathbb{F}_q(\lambda_1, \ldots, \lambda_{\bar{n}}, \mu_1, \ldots, \mu_{\bar{n}})$. The low-access rack-aware RS code is then defined over $\mathbb{E}$ with evaluation points $\lambda_{ij}, i = 1, \ldots, \bar{n}, j = 1, \ldots, u$. The access cost of this code can be shown to equal $\bar{d}ul/\bar{s}$. Moreover, it is possible to adopt the approach of Sec. 2.4.2 to demonstrate a repair scheme for this code that supports both optimal error correction and low access cost but we leave it to interested readers.

Chapter 4: Cyclic and Convolutional Codes with Locality

4.1 Introduction

LRC codes and their variants have been extensively studied in recent years. In this chapter we focus on cyclic constructions of LRC codes and derive conditions on the zeros of the code that support the property of hierarchical locality. As a result, we obtain a general family of hierarchical LRC codes for a new range of code parameters. We also observe that our approach enables one to represent an LRC code in quasicyclic form, and use this representation to construct tail-biting convolutional LRC codes with locality. Among other results, we extend the general approach to cyclic codes with locality to multidimensional cyclic codes, yielding new families of LRC codes with availability, and construct a family of q-ary cyclic hierarchical LRC codes of unbounded length.

4.1.1 Organization

We begin with a characterization of the structure of the zeros of cyclic codes in Sec. 4.2, which plays a fundamental role in the results of this chapter. Then we present our results on cyclic H-LRC codes in Sec. 4.3, including families of cyclic H-

LRC codes and conditions for which the cyclic H-LRC codes are strong optimal. In Sec. 4.3, we also address the problem of maximum length of optimal H-LRC codes, wherein the main question is constructing such codes of length larger than the size of the code alphabet.

Next, we derive an upper bound on the column distance of convolutional codes with locality and present a family of tailbiting convolutional codes with row locality in Sec. 4.4. The construction of the code family is done by exploiting a classic link between quasicyclic codes and convolutional codes [70], whose parameters are controlled by the set of zeros of the underlying quasicyclic code.

In Sec. 4.5 we study another variant of codes with locality, namely, LRC codes with availability. Specifically, building upon the characterization of zeros of cyclic codes and the technique of code concatenation, we construct a family of bi-cyclic codes with availability whose rate is higher than that of product codes of LRC codes with the same distance guarantee.

4.2 The structure of zeros of cyclic codes with locality

In this section, we characterize the structure of zeros of cyclic codes with locality, which relates to the local dimension, the local distance, and the (global) distance of the code.

4.2.1 Optimal cyclic LRC codes

Let $\mathcal{C}$ be a cyclic code of length n with generator polynomial $g(x)$ and check polynomial $h(x) = \frac{x^n-1}{g(x)}$. The dual code $\mathcal{C}'$ has generator polynomial $g_{\mathcal{C}'}(x) = x^{\deg(h(x))}h(x^{-1})$, and the code $\overleftarrow{\mathcal{C}}'$ generated by $h(x)$ is obtained from $\mathcal{C}'$ by inverting the order of the coordinates. A codeword $a(x) \in \mathcal{C}'$ of weight $r+1$ defines a repair group of the code $\mathcal{C}$, and so does the reversed codeword $\overleftarrow{a}(x) \in \overleftarrow{\mathcal{C}}'$. For this reason below in this section we argue about the code $\overleftarrow{\mathcal{C}}'$ rather than $\mathcal{C}'$, which makes the writing more compact without affecting the conclusions.

Let us recall a connection between cyclic codes and LRC codes of [73], which we present in the form close to the earlier works [6, 50]. The following lemma underlies constructions of cyclic LRC codes in this chapter and elsewhere, and it represents a mild extension of Lemma 3.3 in [73]. In the statement as well as elsewhere in the chapter we do not distinguish between zeros of the code and their exponents in terms of some fixed primitive nth root of unity in $\mathbb{F}_q$.

Lemma 33. *Let $\mathcal{C}$ be a cyclic code over $\mathbb{F}_q$ of length $n|(q-1)$ and let α be a primitive nth root of unity in $\mathbb{F}_q$. Suppose that $n = \nu m$ for some integers ν, m. Then the code $\overleftarrow{\mathcal{C}}'$ contains a vector*

$$b(x) = \sum_{i=0}^{m-1} x^{i\nu}\alpha^{(m-1-i)\nu u}, \quad u \in \{0, \ldots, m-1\} \tag{4.1}$$

if and only if the set $\mathcal{L} = \{u + im, i = 0, 1, \ldots, \nu - 1\}$ is among the zeros of $\mathcal{C}$.

Proof. Notice that the polynomial $b(x)$ can be equivalently written as $b(x) = \frac{x^n-1}{x^\nu-\alpha^{\nu u}}$,

where

$$x^{\nu} - \alpha^{\nu u} = \prod_{i=0}^{\nu-1}(x - \alpha^{u+im}) \tag{4.2}$$

is the annihilator polynomial of the set $\mathcal{L}$. Thus, $b(\alpha^t) \neq 0$ for all $t \in \mathcal{L}$. If $b(x) \in \overleftarrow{\mathcal{C}'}$, this implies that $\mathcal{L}$ is a subset of the set of zeros of $\mathcal{C}$.

Conversely, let $g(x) = (x^{\nu} - \alpha^{\nu u})p(x)$ be the generator polynomial of $\mathcal{C}$. Then

$$\Big(\frac{x^n - 1}{g(x)}\Big)p(x) = \frac{x^n - 1}{x^{\nu} - \alpha^{\nu u}} = b(x) \in \overleftarrow{\mathcal{C}'}.$$

□

This lemma immediately yields the cyclic codes from [73] (the cyclic case of the codes from [71]).

Theorem 34 ([73])**.** *Let $(r+1)|n, r|k, n|(q-1)$. Let $\alpha \in \mathbb{F}_q$ be a primitive n-th root of unity, and let $\mathcal{C}$ be an (n,k) cyclic code with zeros $\alpha^i, i \in \mathcal{Z} := \mathcal{L} \cup \mathcal{D}$, where*

$$\begin{aligned} \mathcal{L} &= \{1 + l(r+1), l = 0, \ldots, \tfrac{n}{r+1} - 1\} \\ \mathcal{D} &= \{1, 2, \ldots, n - \frac{k}{r}(r+1) + 1\}. \end{aligned} \tag{4.3}$$

Then $\mathcal{C}$ is an (n,k,r) optimal LRC code.

Proof. Since $\mathcal{L} \subset \mathcal{Z}$, Lemma 33 implies that

$$b(x) = \sum_{i=0}^{r} \alpha^{i\frac{n}{r+1}} x^{(r-i)\frac{n}{r+1}}$$

is a codeword in $\overleftarrow{\mathcal{C}'}$. This codeword is of weight $r+1$, and its cyclic shifts give $\frac{n}{r+1}$

disjoint repair groups, supporting the locality claim. At the same time, the BCH bound implies that $d(\mathcal{C}) \geqslant n - \frac{k}{r}(r+1) + 2$, so the code is optimal by (1.4) once one observes $|\mathcal{Z}| = n - k$ and $\dim(\mathcal{C}) = n - |\mathcal{Z}| = k$. □

This construction extends without difficulty to codes with (r, δ) locality for any $\delta \geqslant 2$. A family of optimal codes in the sense of the bound (1.6) was constructed in [71], Construction 8 (see also [11]). The codes in this family are constructed as certain subcodes of Reed-Solomon codes that rely on piecewise-constant polynomials over $\mathbb{F}_q$. In the particular case that the code length n divides $q-1$ it is possible to represent these codes in cyclic form. For this, we assume that $r|k$, take $m = r+\delta-1$, and take the zeros of the code to be

$$\begin{aligned} \mathcal{L} &= \{i + lm \mid l = 0, \ldots, \nu - 1, i = 1, \ldots, \delta - 1\} \\ \mathcal{D} &= \{1, 2, \ldots, n - k - ((k/r) - 1)(\delta - 1)\}. \end{aligned} \tag{4.4}$$

As will be apparent from the proof of Lemma 35, the condition about zeros given by the set $\mathcal{L}$ translates into conditions on the dual code that support the locality claim. The distance of the code $\mathcal{C}$ clearly meets the bound (1.6) with equality.

4.2.2 Cyclic codes with locality

In the next lemma we present a slightly more general view of the method in Theorem 34 that will be instrumental in the code constructions below in this work. The main element of the construction is code $\mathcal{C}^{(0)}$ defined in (4.5), which isolates a repair group in the code $\mathcal{C}$ and supports local correction of several erasures.

Lemma 35. *Let $n|(q-1), n = \nu m$. Let α be a primitive n-th root of unity in $\mathbb{F}_q$, and fix $\delta \in \{2, \ldots, m\}$. Let $\mathcal{Z}$ be a subset of size Z such that*

$$\{1, \ldots, \delta - 1\} \subset \mathcal{Z} \subset \{0, \ldots, m-1\}$$

and let $\mathcal{L} = \bigcup_{s=0}^{\nu-1}(\mathcal{Z} + sm)$. Consider a cyclic code $\mathcal{C} = \langle g(x) \rangle$ of length n, where

$$g(x) = p(x) \prod_{t \in \mathcal{L}} (x - \alpha^t),$$

and $p(x) \in \mathbb{F}_q[x]$ is some polynomial. Let

$$\mathcal{C}^{(0)} = \{(c_0, c_\nu, \ldots, c_{(m-1)\nu}) \mid (c_0, \ldots, c_{n-1}) \in \mathcal{C}\}. \tag{4.5}$$

Then $\dim(\mathcal{C}^{(0)}) \leqslant m - Z, \quad d(\mathcal{C}^{(0)}) \geqslant \delta$, and thus, the code $\mathcal{C}$ is an LRC code with $(m - Z, \delta)$ locality.

Further, if for every $u \in \{0, \ldots, m-1\} \backslash \mathcal{Z}$ there exists $s \in \{0, 1, \ldots, \nu - 1\}$ such that $g(\alpha^{u+sm}) \neq 0$, then $\dim(\mathcal{C}^{(0)}) = m - Z$.

Proof. We proceed similarly to Theorem 34. Let

$$l(x) = \prod_{t \in \mathcal{L}} (x - \alpha^t) = \prod_{s=0}^{\nu-1} \prod_{i \in \mathcal{Z}} (x - \alpha^{i+sm}) = \prod_{i \in \mathcal{Z}} l_i(x)$$

where $l_i(x) := x^\nu - \alpha^{\nu i}$. Let $h(x) = \frac{x^n - 1}{g(x)}$ and consider a subset of Z codewords of

$\overleftarrow{\mathcal{C}}'$ given by

$$b_i(x) := h(x)p(x) \prod_{j \in \mathcal{Z} \setminus \{i\}} l_j(x), \quad i \in \mathcal{Z}.$$

A codeword b_i has the form

$$b_i(x) = \frac{x^n - 1}{l_i(x)} = \sum_{j=0}^{m-1} \alpha^{(m-1-j)i\nu} x^{j\nu},$$

Hamming weight m, and contains $\nu - 1$ zero coordinates after every nonzero entry.

To prove the statement about locality, let us form a $Z \times m$ matrix H obtained by inverting the order of coordinates in the codewords $b_i, i \in \mathcal{Z}$, writing the resulting vectors as rows, and discarding all the zero columns. By construction, every row of H is a parity-check equation of the code $\mathcal{C}^{(0)}$. Any submatrix of $\delta - 1$ columns of H has rank $\delta - 1$ (its first $\delta - 1$ rows form a Vandermonde determinant), and thus, $d(\mathcal{C}^{(0)}) \geqslant \delta$. Since the rows of H give Z independent parity-check equations for the code $\mathcal{C}^{(0)}$, we also have $\dim(\mathcal{C}^{(0)}) \leqslant m - Z$. This argument exhibits a local code in the coordinates that are integer multiples of ν, and by cyclic shifts we can partition the set $\{0, 1, \ldots, n-1\}$ into supports of disjoint local codes of length m and distance at least δ. Furthermore, we note that the punctured code $\mathcal{C}^{(0)}$ is itself a cyclic code of length $m|n$. Let $g_0(x)$ be its generator polynomial. Since each row of H is a parity-check equation for the code $\mathcal{C}^{(0)}$, we have $g_0(\alpha^{i\nu}) = 0$ for every $i \in \mathcal{Z}$ and thus $\deg(g_0(x)) \geqslant Z$. Together these arguments prove the claim about $(m - Z, \delta)$ locality of the code $\mathcal{C}$.

Next we show that if $\deg(g_0(x)) > Z$, then necessarily there exists $u \in \{0, \ldots, m-$

$1\}\backslash\mathcal{Z}$ such that $g(\alpha^{u+sm}) = 0$ for all $s = 0, \ldots, \nu - 1$. Suppose that there exists $u \in \{0, \ldots, m-1\}\backslash\mathcal{Z}$ such that $g_0(\alpha^{u\nu}) = 0$. By Eq. (4.2) in the proof Lemma 33 there exists a codeword $b_u \in \overleftarrow{\mathcal{C}}'$ given by

$$b_u(x) = \sum_{j=0}^{m-1} \alpha^{ju\nu} x^{(m-1-j)\nu} = \frac{x^n - 1}{x^\nu - \alpha^{\nu u}}.$$

On the other hand, $\overleftarrow{\mathcal{C}}' = \langle h(x) \rangle$, so $h(x)|b_u(x)$ and therefore $(x^\nu - \alpha^{\nu u})|g(x)$. Noticing that $x^\nu - \alpha^{\nu u} = \prod_{s=0}^{\nu-1}(x - \alpha^{u+sm})$ (cf. (4.2)), we conclude that $g(x)$ is divisible by $x - \alpha^{u+sm}$ for all $s = 0, \ldots, \nu - 1$. Hence, if for every $u \in \{0, \ldots, m-1\}\backslash\mathcal{Z}$ there exists $0 \leqslant s \leqslant \nu - 1$ such that $(x - \alpha^{u+sm}) \nmid g(x)$, i.e., $g(\alpha^{u+sm}) \neq 0$, then $\deg(g_0(x)) = Z$, and thus $\dim(\mathcal{C}^{(0)}) = m - Z$. □

4.3 Codes with hierarchical locality

4.3.1 Optimal cyclic codes with hierarchy

In this section we construct a family of H-LRC cyclic codes with $h \geqslant 1$ levels of hierarchy and derive sufficient conditions for their optimality. Suppose that h is fixed and we are given the local dimension r_1 (the dimension of the first, innermost local code), and the designed local distances $1 = \delta_0 \leqslant \delta_1 \leqslant \ldots \leqslant \delta_h \leqslant \delta_{h+1}$, where δ_{h+1} is the designed distance of the overall code $\mathcal{C}$.

We will assume that the first local code is MDS and thus its length is $n_1 = r_1 + \delta_1 - 1$. For $1 \leqslant i \leqslant h$, let $n_{i+1} = \nu_i n_i$ be the length of the code in the $(i+1)$st level of hierarchy, where $\nu_i > 1$ is an integer. Let $\mathbb{F}_q$ be a finite field and suppose

that $n_{h+1}|(q-1)$.

We construct a cyclic H-LRC code $\mathcal{C}$ over $\mathbb{F}_q$ of length $n = n_{h+1}$ and designed (local) distances $\delta_1, \ldots, \delta_{h+1}$ as follows. Let $\alpha \in \mathbb{F}_q$ be a primitive n-th root of unity. The code $\mathcal{C}$ will be given by its defining set of zeros $\mathcal{Z}$ which we specify via a recursive procedure. Consider the set of exponents $\mathcal{D}_1 = \{1, \ldots, \delta_1 - 1\}$ of the primitive element α. Further, let $\mathcal{L}_1 = \varnothing$ and

$$\mathcal{Z}_1 = \mathcal{L}_1 \cup \mathcal{D}_1. \tag{4.6}$$

Having (4.3) in mind, for $1 \leqslant i \leqslant h$ let

$$\begin{gathered} \mathcal{L}_{i+1} = \bigcup_{s=0}^{\nu_i - 1} (\mathcal{Z}_i + s n_i), \quad \mathcal{D}_{i+1} = \{1, \ldots, \delta_{i+1} - 1\}, \\ \mathcal{Z}_{i+1} = \mathcal{L}_{i+1} \cup \mathcal{D}_{i+1}. \end{gathered} \tag{4.7}$$

Finally, put $\mathcal{Z} = \mathcal{Z}_{h+1}$.

The generator polynomial of the cyclic code $\mathcal{C}$ of length n is given by

$$g(x) = \prod_{t \in \mathcal{Z}} (x - \alpha^t). \tag{4.8}$$

The parameters of the code $\mathcal{C}$ are estimated as follows.

Proposition 36. *(i) The dimension of the code $\mathcal{C}$ is $n - |\mathcal{Z}|$ and the distance $d \geqslant \delta_{h+1}$. (ii) The code $\mathcal{C}$ is an h-level H-LRC code with locality $(n_i - |\mathcal{Z}_i|, \delta_i), i = 1, \ldots, h$.*

Proof. (i) The value of the dimension is clear from the construction, and the estimate for the distance comes from the BCH bound.

(ii) The statement about the locality parameters follows by Lemma 35 once we observe that $g(x)$ is divisible by $\prod_{t\in\bigcup_{s=0}^{n/n_i-1}(\mathcal{Z}_i+sn_i)}(x-\alpha^t)$ for every $i=1,\ldots,h$. $\square$

Next we examine the conditions that suffice for the distance of $\mathcal{C}$ to meet the bound (1.5) with equality. We build up the optimality of our code in an incremental manner in the sense that we first ensure that the first local codes are optimal (i.e., MDS codes), and relying on these optimal local codes we make sure that the second local codes are optimal (i.e., optimal LRC codes), and so forth until we reach the outermost code.

Let $r_1 < r_2 < \ldots < r_h < r_{h+1} = \dim(\mathcal{C})$ be the chosen values of the dimensions of the local codes. As before, we set $n_1 = r_1+\delta_1-\delta_0$ and let $n_{i+1} = \nu_i n_i$ for $1 \leqslant i \leqslant h$ where the integer number ν_i satisfies $\nu_i \geqslant \lceil r_{i+1}/r_i \rceil$. Note that this assumption does not entail a loss of generality since, assuming the ith and $(i+1)$st local codes are optimal, by (4.7) we have $|\mathcal{L}_{i+1}| = (n_i - r_i)\nu_i$, and $r_{i+1} = n_{i+1} - |\mathcal{Z}_{i+1}| \leqslant n_{i+1} - |\mathcal{L}_{i+1}| = r_i\nu_i$.

To show optimality of the code, we connect the target values of the local distances $\delta_2, \delta_3, \ldots, \delta_{h+1}$ with the dimension values through several auxiliary parameters. For $1 \leqslant i \leqslant h$, let us write $r_{i+1} = a_i r_i - b_i$ where $0 \leqslant b_i < r_i$. Further, let $b_i^{(i)} = b_i$ and for $j = i, i-1, \ldots, 2$, let

$$b_j^{(i)} = u_{j-1}^{(i)} r_{j-1} + b_{j-1}^{(i)}, \quad 0 \leqslant b_{j-1}^{(i)} < r_{j-1}.$$

Put $b_0^{(0)} = 0$ and for $1 \leqslant i \leqslant h$ let

$$b_0^{(i)} = (b_1^{(i)} + b_0^{(i-1)}) \bmod r_1, \quad u_0^{(i)} = \left\lfloor \frac{b_1^{(i)} + b_0^{(i-1)}}{r_1} \right\rfloor. \tag{4.9}$$

Finally, for $1 \leqslant i \leqslant h$, let

$$\delta_{i+1} = (\nu_i - a_i)n_i + \delta_i + \sum_{j=1}^{i-1} u_j^{(i)} n_j + u_0^{(i)} n_1 + b_0^{(i)} - b_0^{(i-1)}. \tag{4.10}$$

The high-level ideas behind these parameters are as follows. By Lemma 35, the quantities $\delta_1, \ldots, \delta_{h+1}$ control the distances via the BCH bound and we would like to make these quantities as large as possible given the target dimensions. We need to make sure that the ith local code has a run of consecutive zeros of length $\delta_i - 1$, and our budget of creating such a run is limited by the dimension. Therefore, we seek to rely on the already present runs of zeros of the jth local codes, $j < i$, and spend the budget frugally on the way to optimality. The quantities $b_j^{(i)}$ serve the purpose bridging the "distance gap" (the gaps between runs of zeros) between the local codes in levels j and $j + 1$ on the way to ensure the distance of the ith code.

As for the dimensions of the local codes, by Lemma 35 they are determined by $\mathcal{Z}_i, 1 \leqslant i \leqslant h + 1$. The cardinality of $\mathcal{Z}_i$ is established in the next claim.

Proposition 37. *For $1 \leqslant i \leqslant h + 1$, we have $|\mathcal{Z}_i| = n_i - r_i$.*

The proof of this proposition proceeds by induction and is given in Appendix C.1. An examination of the proof also gives a better understanding of the parameters a_i, b_i introduced above.

On account of Proposition 37, the locality parameters of the code $\mathcal{C}$ are $(r_i, \delta_i), 1 \leqslant i \leqslant h$. Furthermore, $\dim(\mathcal{C}) = r_{h+1}$, and by the BCH bound $d(\mathcal{C}) \geqslant \delta_{h+1}$.

Sufficient conditions for optimality of the code $\mathcal{C}$ are given in the following lemma whose proof is given in Appendix C.2.

Lemma 38. *Suppose that for $i \geqslant 2$ and $2 \leqslant s \leqslant i$ the following conditions are satisfied:*

$$\begin{aligned} \left\lceil \frac{r_{s+1}}{r_s} \right\rceil \left\lceil \frac{r_s}{r_1} \right\rceil - \left\lceil \frac{r_{s+1}}{r_1} \right\rceil &= u_0^{(s)} + u_1^{(s)} + \sum_{j=2}^{s-1} u_j^{(s)} \left\lceil \frac{r_j}{r_1} \right\rceil \\ \left\lceil \frac{r_{s+1}}{r_s} \right\rceil \left\lceil \frac{r_s}{r_l} \right\rceil - \left\lceil \frac{r_{s+1}}{r_l} \right\rceil &= u_l^{(s)} + \sum_{j=l+1}^{s-1} u_j^{(s)} \left\lceil \frac{r_j}{r_l} \right\rceil, \end{aligned} \tag{4.11}$$

where the first condition holds for $s \geqslant 2$ and the second for $2 \leqslant l \leqslant s-1$. Then for $1 \leqslant i \leqslant h$, we have

$$\delta_{i+1} = n_{i+1} - r_{i+1} + \delta_i - \sum_{l=1}^{i} \left\lceil \frac{r_{i+1}}{r_l} \right\rceil (\delta_l - \delta_{l-1}).$$

It follows from Lemma 35, Lemma 38, and the bound (1.5) that the code $\mathcal{C}$ is an $(n = n_{h+1}, k = r_{h+1})$ optimal H-LRC code with local parameters $(r_i, \delta_i), 1 \leqslant i \leqslant h$. Clearly, when $h = 0$ our construction gives an (n_1, r_1) MDS code and when $h = 1$, it gives an (n_2, r_2) optimal LRC code of [73]. For $h = 2$, conditions (4.11) take a simpler form:

$$\left\lceil \frac{r_3}{r_2} \right\rceil \left\lceil \frac{r_2}{r_1} \right\rceil - \left\lceil \frac{r_3}{r_1} \right\rceil = \left\lfloor \frac{b_1 + b_2}{r_1} \right\rfloor. \tag{4.12}$$

We note that the condition of [63, Theorem 2.6] is easily seen to be equivalent to

(4.12). Another known case of optimality, the divisibility conditions $r_i | r_{i+1}, i = 1, \ldots, h$, is also covered by Lemma 38 (in this case both the left-hand sides and the right-hand sides of (4.11) are zero).

Let us give a general example of the choice of parameters that ensures optimality. Suppose that $r_1 \geqslant 2^h$ and $r_{i+1} = 2r_i - 1$ for $1 \leqslant i \leqslant h$. Then conditions (4.11) are satisfied. Indeed, we have $r_i = 2^{i-1}(r_1 - 1) + 1$ for $1 \leqslant i \leqslant h + 1$. Therefore, for $1 \leqslant j \leqslant i \leqslant h + 1$ we have

$$\left\lceil \frac{r_i}{r_j} \right\rceil = \left\lceil 2^{i-j} - \frac{2^{i-j} - 1}{2^j(r_1 - 1) + 1} \right\rceil = 2^{i-j},$$

where the last equality follows because $r_1 \geqslant 2^h$. It follows that the left-hand sides of conditions (4.11) are zero. On the other hand, we have $b_i = 1$ for all $1 \leqslant i \leqslant h$. Since $r_1 \geqslant 2^h$, we have $u_l^{(s)} = 0$ for all $1 \leqslant s \leqslant h$ and $0 \leqslant l \leqslant s - 1$, and thus, the right-hand sides are also zero, which confirms the optimality claim.

Proposition 39. *Suppose that the conditions* (4.11) *are satisfied, then the code* $\mathcal{C}$ *is a strongly optimal H-LRC code in the sense of Sec. 1.2.5.*

Proof. It suffices to show that the dimension of the i-th local code *equals* r_i for all i.

By assumption, we have $r_{i+1} > r_i$ and thus $a_i \geqslant 2$ for $1 \leqslant i \leqslant h$. It is not difficult to verify from (4.10) that $\delta_{i+1} \leqslant (\nu_i - a_i + 1)n_i \leqslant n_{i+1} - n_i$ for all $1 \leqslant i \leqslant h$. We claim that $g(\alpha^{n-n_i+t_i}) \neq 0$ for every $t_i \in \mathcal{T}_i, 1 \leqslant i \leqslant h$ where

$$\mathcal{T}_i = \{n\} \cup (\{1, \ldots, n_i - 1\} \backslash \mathcal{Z}_i + n - n_i).$$

Then by the second part of Lemma 35 the dimension of the i-th local code equals r_i and the strong optimality follows.

Now let us show $n-n_i+t_i \notin \mathcal{Z}$. Observe that the set $\mathcal{Z}$ contains n/n_i copies of $\mathcal{Z}_i$ and the set $\mathcal{T}_i$ is the complement to the last one of those copies with respect to $\{1,\dots,n_i\}$. Indeed, we have $n-n_i+t_i = t_i+(n/n_i-1)n_i \notin \{n\}\cup(\mathcal{Z}_i+(n/n_i-1)n_i)$. Now consider the last copy of $\mathcal{Z}_{i+1}$ contained in $\mathcal{Z}$. Obviously, it contains the last copy of $\mathcal{Z}_i$. To establish $n-n_i+t_i \notin \mathcal{Z}$, it remains to show $n-n_i+t_i$ is not in the last copy of $\mathcal{D}_{i+1}$, namely, $n-n_i+t_i \notin \mathcal{D}_{i+1}+n-n_{i+1}$. Since $\delta_{i+1} \leqslant n_{i+1}-n_i$ as we observed above and $n_{i+1}-n_i+t_i \geqslant n_{i+1}-n_i+\delta_i$, we have $n_{i+1}-n_i+t_i \notin \mathcal{D}_{i+1}$. It follows that $n-n_i+t_i \notin \mathcal{D}_{i+1}+n-n_{i+1}$. Therefore, $n-n_i+t_i \notin \mathcal{Z}$ and $g(\alpha^{n-n_i+t_i}) \neq 0$ for every $t_i \in \mathcal{T}_i, 1 \leqslant i \leqslant h$.

In the case that $r_{i+1}=r_i$ for some $1 \leqslant i \leqslant h$ (although we rule out this trivial case in Definition 3), the code $\mathcal{C}$ is still strongly optimal if the optimality conditions are satisfied. In fact, if $r_{i+1}=r_i$ then from (4.10) we have $\delta_{i+1}=(\nu_i-a_i)n_i+\delta_i = n_{i+1}-n_i+\delta_i \notin \mathcal{D}_{i+1}$. By similar arguments as above, we have $n-n_i+t_i \notin \mathcal{Z}$ for every $t_i \in \mathcal{T}_i, 1 \leqslant i \leqslant h$, and thus establish the strong optimality of the code. □

We conclude with a numerical example that shows that the assumptions on the parameters can be simultaneously satisfied for moderate values of the length and alphabet size.

Example 1. *Consider the case $h=3$. Let $r_1=2$ and $\delta_1=2$. Then $n_1=3$. Let $(n_2,r_2)=(9,3)$, $(n_3,r_3)=(27,5)$, and $(n_4,r_4)=(81,7)$. Then our construction (with designed distances found from (4.10)) gives rise to a strongly optimal H-LRC*

code of length $n = n_4$ *and dimension* $k = r_4$ *with local distances* $\delta_2 = 6, \delta_3 = 17$ *and distance* $d = \delta_4 = 53$ *over a finite field* $\mathbb{F}_q$ *where* $81|(q-1)$ *(for example, we can take* $q = 163$*).*

4.3.2 Hierarchical cyclic codes of unbounded length

In this section we construct a family of H-LRC codes with distance $d = \delta_h + 1, h \geqslant 1$ and unbounded length. The construction combines the idea of [50] with H-LRC codes of the previous section.

Let $1 < r_1 < r_2 < \ldots < r_h$ be integers. Let $1 = \delta_0 < \delta_1$ and let $\delta_2, \delta_3, \ldots, \delta_h$ be as in (4.10). Again, we put $n_1 = r_1 + \delta_1 - \delta_0$ and let $n_{i+1} = \nu_i n_i$ for $1 \leqslant i \leqslant h-1$, where $\nu_i \geqslant \lceil r_{i+1}/r_i \rceil$ is an integer. Let $\mathbb{F}_{q^m}, m \geqslant 1$ be a finite field and let $n_h|(q-1)$. Let $n = q^m - 1$ and observe that $n_h|n$. Let $\alpha \in \mathbb{F}_{q^m}$ be a primitive n-th root of unity. Let $\mathcal{Z}_h$ be constructed by the procedure in (4.6) and (4.7). Finally, define

$$\mathcal{L} = \bigcup_{s=0}^{n/n_h-1} (\mathcal{Z}_h + sn_h), \quad \mathcal{Z} = \mathcal{L} \cup \{0\}. \tag{4.13}$$

Consider a cyclic code $\mathcal{C}$ with generator polynomial

$$g(x) = \prod_{t \in \mathcal{Z}} (x - \alpha^t). \tag{4.14}$$

As is easily seen, $g(x) \in \mathbb{F}_q[x]$. Indeed,

$$g(x) = (x-1) \prod_{t \in \mathcal{L}} (x - \alpha^t)$$

$$= (x-1)\prod_{s=0}^{n/n_h-1}\prod_{t\in\mathcal{Z}_h}(x-\alpha^{sn_h+t})$$

$$= (x-1)\prod_{t\in\mathcal{Z}_h}(x^{n/n_h}-\alpha^{nt/n_h}).$$

For the last equality we note that $x^{n/n_h}-\alpha^{nt/n_h} = \prod_{s=0}^{n/n_h-1}(x-\alpha^{sn_h+t})$. Observe that for $t\in\mathcal{Z}_h$ we have $(\alpha^{nt/n_h})^{q-1}=1$ since $n_h|(q-1)$. It follows that $\alpha^{nt/n_h}\in\mathbb{F}_q$ for $t\in\mathcal{Z}_h$ and thus $g(x)\in\mathbb{F}_q[x]$.

Proposition 40. *Let $\mathcal{C}=\langle g(x)\rangle\in\mathbb{F}_q[x]/(x^n-1)$ be a cyclic code. Then $\dim(\mathcal{C}) = nr_n/n_h-1$, $d(\mathcal{C})\geqslant\delta_h+1$, and the locality parameters are (r_i,δ_i) for $1\leqslant i\leqslant h$.*

Proof. The dimension of $\mathcal{C}$ is found as

$$k = n-\deg(g(x)) = n-1-\frac{|\mathcal{Z}_h|n}{n_h} = \frac{nr_h}{n_h}-1,$$

where the last equality follows by Proposition 37. The distance of the code $\mathcal{C}$ is $d\geqslant\delta_h+1$ since $g(x)$ has consecutive roots $\alpha^t, t=0,\ldots,\delta_h-1$.

The locality parameters of the code $\mathcal{C}$ follow immediately by Lemma 35 and Proposition 37. □

The next lemma provides the conditions when the code $\mathcal{C}$ is optimal. Its proof amounts to a calculation based on Lemma 38 and Proposition 40.

Lemma 41. *Suppose that for $1\leqslant i\leqslant h-1$, conditions (4.11) are satisfied. Further, suppose that*

$$\frac{n}{n_h}\left\lceil\frac{r_h}{r_l}\right\rceil = \left\lceil\frac{k}{r_l}\right\rceil, \quad 1\leqslant l\leqslant h-1. \tag{4.15}$$

Then the code $\mathcal{C}$ is optimal.

Proof. By Lemma 38, we have

$$\delta_h = n_h - r_h + \delta_{h-1} - \sum_{l=1}^{h-1} \left\lceil \frac{r_h}{r_l} \right\rceil (\delta_l - \delta_{l-1}). \tag{4.16}$$

By Proposition 40, we have $k = nr_h/n_h - 1$. Using the bound (1.5), the distance of the code cannot exceed

$$\begin{aligned} n - k + \delta_h - \sum_{l=1}^{h} \left\lceil \frac{k}{r_l} \right\rceil (\delta_l - \delta_{l-1}) & \\ = n - \frac{nr_h}{n_h} + 1 + \delta_h - \frac{n}{n_h}(\delta_h - \delta_{h-1}) - \sum_{l=1}^{h-1} \left\lceil \frac{k}{r_l} \right\rceil (\delta_l - \delta_{l-1}) & \quad (4.17) \\ = 1 + \delta_h + \frac{n}{n_h} \sum_{l=1}^{h-1} \left\lceil \frac{r_h}{r_l} \right\rceil (\delta_l - \delta_{l-1}) - \sum_{l=1}^{h-1} \left\lceil \frac{k}{r_l} \right\rceil (\delta_l - \delta_{l-1}) & \quad (4.18) \\ = 1 + \delta_h, & \quad (4.19) \end{aligned}$$

where (4.17) follows since $r_h > 1$ implies $\lceil k/r_h \rceil = n/n_h$, in (4.18) we used (4.16), and (4.19) follows by (4.15). Hence, the code $\mathcal{C}$ has the largest possible distance $d = \delta_h + 1$. □

In particular, the conditions in Lemma 41 are satisfied when $r_i | r_{i+1}, i = 1, \ldots, h-1$ and $r_h | k$.

As in Sec. 4.3.1, the code $\mathcal{C}$ constructed above in this section has the strong optimality property if the optimality conditions in Lemma 41 are satisfied. Specifically, the main difference between the construction in this section and the one in the previous section is in the final step of constructing the defining set $\mathcal{Z}$, which also

includes element 0 (i.e., α^0). By an argument similar to Sec. 4.3.1, one can show $n - n_i + t_i \notin \mathcal{Z}$ for every $t_i \in (\{0, \ldots, n_i - 1\} \backslash \mathcal{Z}_i) + (n - n_i), 1 \leqslant i \leqslant h$ and so strong optimality follows.

Example 2. *Consider the case $h = 3$. Let $r_1 = 2$ and $\delta_1 = 2$. Then $n_1 = 3$. Let $(n_2, r_2) = (9, 3)$ and $(n_3, r_3) = (27, 5)$. Let $m \geqslant 1$ be an arbitrary integer and $q = 163$. Then our construction (with designed local distances given by (4.10)) gives rise to a strongly optimal H-LRC code of length $n = 163^m - 1$ and dimension $k = 2(163^m - 1)/9$ with local distances $\delta_2 = 6, \delta_3 = 17$ and distance $d = 18$ over $\mathbb{F}_q$.*

Recall that [6] shows that the length of an optimal LRC code in the general case cannot be greater than a certain power of the alphabet size q. Using similar arguments, it might be possible to derive upper bounds on the length of optimal H-LRC codes in the general case; however already in the case of (r, δ) locality addressed in [10] (with just a single level of hierarchy), following this route requires cumbersome calculations.

4.4 Convolutional codes with locality

It has been recognized a long while ago that quasi-cyclic codes can be encoded convolutionally, and multiple papers constructed families of convolutional codes from their quasicyclic counterparts [21, 38, 78]. In this section, we present a family of convolutional codes with locality by relying on the tailbiting version of convolutional codes [70]. We single out this approach because it enables us to establish the locality properties of convolutional codes based on the properties of cyclic H-LRC codes

constructed above in this chapter.

We begin with a brief reminder of the basic notions for convolutional codes [37]. Let D be an indeterminate and let $\mathbb{F}_q(D)$ be the field of rational functions of one variable over $\mathbb{F}_q$. A q-ary (n, k) convolutional code $\mathcal{C}$ is a linear k-dimensional subspace of $\mathbb{F}_q(D)^n$. A generator matrix $G(D) = (g_{ij}(D))$ of the code $\mathcal{C}$ is a $k \times n$ matrix with entries in $\mathbb{F}_q(D)$ whose rows form a basis of $\mathcal{C}$. Thus, the code $\mathcal{C}$ is a linear space $\{u(D)G(D) \mid u(D) \in \mathbb{F}_q(D)^k\}$. The matrix $G(D)$ can be transformed to the polynomial form by multiplying every element by the least common denominator of its entries. The transformed matrix generates the same code $\mathcal{C}$, and so in the sequel we will consider only *polynomial generator matrices*. Below we will assume that the generator matrix $G(D)$ is a $k \times n$ matrix with entries in $\mathbb{F}_q[D]$, where $\mathbb{F}_q[D]$ is the ring of polynomials over $\mathbb{F}_q$.

For $1 \leqslant i \leqslant k$, the *degree m_i of the i-th row of $G(D)$* is the maximum degree of the entries in row i, namely, $m_i = \max_{1 \leqslant j \leqslant n} \deg(g_{i,j}(D))$. As with linear block codes, the encoding of a convolutional code depends on the choice of a generator matrix. The maximum degree $M := \max_{1 \leqslant i \leqslant k} m_i$ is called the *memory* of the encoder. The generator matrix of the code $\mathcal{C}$ can also be written in the form

$$G = \begin{bmatrix} G_0 & G_1 & \dots & G_M & & \\ & G_0 & G_1 & \dots & G_M & \\ & & \ddots & \ddots & & \ddots \end{bmatrix}, \tag{4.20}$$

where each G_i is a $k \times n$ matrix over $\mathbb{F}_q$. The codeword of the code $\mathcal{C}$ is obtained as

a product uG, where u is a semi-infinite input sequence of symbols of $\mathbb{F}_q$.

With a given convolutional code $\mathcal{C}$ one can associate a multitude of distance measures. In direct analogy with block codes, one defines the *free distance* of the code $\mathcal{C}$ as the minimum Hamming weight of the Laurent expansions of the nonzero codewords.

Another distance measure of interest is the so-called *column distance* of the code [37, p.162]. To define it, let $\mathcal{C}_{[0,j]}$ be the truncated code of $\mathcal{C}$ at the j-th time instant, $j \geqslant 0$, namely,

$$\mathcal{C}_{[0,j]} = \{c_{[0,j]}(D) = \sum_{i=0}^{j} c_i D^i \mid c(D) = \sum_{i \geqslant 0} c_i D^i \in \mathcal{C}\}.$$

This is a linear block code of length $n(j+1)$, and by (4.20) its generator matrix can be written in the form

$$G_j^c = \begin{bmatrix} G_0 & G_1 & \dots & G_j \\ & G_0 & \dots & G_{j-1} \\ & & \ddots & \vdots \\ & & & G_0 \end{bmatrix} \tag{4.21}$$

where we put $G_l = 0$ for $l > M$. Clearly, the code $\mathcal{C}_{[0,j]}$ is obtained by truncating the code $\mathcal{C}$ to its first $j+1$ entries. A codeword of $\mathcal{C}_{[0,j]}$ has the form $(c_0, c_1, \dots, c_j)$, where for $j \leqslant M$ and each $l = 0, 1, \dots, j$

$$c_l = \sum_{i=0}^{l} u_i G_{l-i}, \tag{4.22}$$

where $c_l = (c_l^{(1)}, \ldots, c_l^{(n)})$ for each l.

We assume throughout that G_0 has full rank, so the mapping $\mathbb{F}_q^k \xrightarrow{G} \mathbb{F}_q^n$ given by $u_0 G_0 \mapsto c_0$ is injective.

Definition 10. *For $j \geqslant 0$ the j-th column distance of $\mathcal{C}$ is given by*

$$d_j^c = \min\{\mathrm{wt}(c_{[0,j]}(D)) \mid c_{[0,j]}(D) \in \mathcal{C}_{[0,j]}, c_0 \neq 0\}.$$

Clearly, the value of d_j^c is at least the minimum distance of the truncated code $\mathcal{C}_{[0,j]}$. This follows because for the column distance we seek the minimum of pairwise distances of codewords that differ in the first coordinate, while the standard minimum distance computation does not involve this assumption. In many cases the column distance is in fact strictly greater. This remark is important for the sliding window repair which enables one to correct more erasures than would be possible for block codes relying on their minimum distance.

Convolutional codes support several forms of erasure repair. One of them, called the sliding window repair [52, 80], is based on the column distance and is used to correct erasures in streaming applications [80]. We illustrate the idea of sliding window repair in Fig. 4.1, representing a code sequence of the code $\mathcal{C}$ as a semi-infinite matrix whose columns are length n vectors $c_l, l \geqslant 0$, and whose row $c^{(i)}, i = 1, \ldots, n$ represents the stream formed by the ith coordinates of the symbols $c_l, l = 0, 1, \ldots$. We begin with fixing j based on the value of the column distance d_j^c of the code. The box in the figure shown with dashed lines represents the window of length $j + 1$ that contains the truncated code at time $l \geqslant j$. The erasures within the

sliding window can clearly be repaired as long as their number at no point exceeds $d_j^c - 1$.

Having in mind streaming applications, one may argue that a more efficient way of repairing erasures is to rely either on the symbols at a fixed time instant, or on a small group of symbols contained within the same stream i. Accordingly, in the next two subsections we define two types of locality for convolutional codes, calling them the column and row localities.

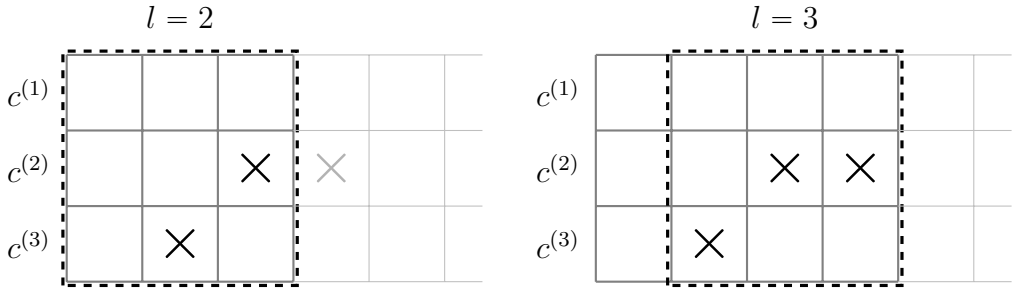

Figure 4.1: Sliding window repair. Suppose $\mathcal{C}$ is an $(3, 2)$ convolutional code with $d_2^c = 4$. Let $(c^{(1)}, c^{(2)}, c^{(3)}) \in \mathcal{C}$ be a codeword, where the crosses denote erasures. At time instant $l = 2$, there are two erasures in the window of length three (dashed box in the left figure), which is less than the 2nd column distance. However, neither of the two erasures are in the first column of the window, and thus their recovery is postponed until later. At time $l = 3$ (right figure), the sliding window contains 3 erasures, of which one is in the first column. This erasure can be recovered from the other symbols in the window. The remaining erasures are corrected in the next steps as long as the number of erasures in the window does not exceed $d_2^c - 1$.

4.4.1 Convolutional codes with column locality

Column locality was introduced in [52]. First let us define the *ith column code* $\mathcal{C}_i, i \geqslant 0$ of a convolutional code $\mathcal{C}$ as a block code of length n given by

$$\mathcal{C}_i = \{c_i \mid c(D) \in \mathcal{C}\}.$$

We say that a convolutional code $\mathcal{C}$ has (r, δ) column locality if for all i the codes $\mathcal{C}_i$ have the (r, δ) locality property.

The results in [52] are based on a version of this definition that requires that *only the code* $\mathcal{C}_M$ support (r, δ) locality. This restriction may seem too narrow until one realizes that if locality is present in the code $\mathcal{C}_M$, then every code $\mathcal{C}_i, i \geqslant 0$ has the (r, δ) locality property. This follows immediately from (4.22) and the definition of $\mathcal{C}_i$ because $\mathcal{C}_i = \mathcal{C}_M$ for $i > M$ and $\mathcal{C}_i$ forms a linear subcode of $\mathcal{C}_M$ otherwise. The only difference between this definition and the one given above is that under the approach of [52], every code $\mathcal{C}_i$ has similarly aligned repair groups which are propagated from the repair groups of $\mathcal{C}_M$, while our definition allows differently aligned repair groups for different values of i.

To enable local repair, we simply assume that every column of the codeword forms a block code with (r, δ) locality. An example is given in Fig 4.2, demonstrating sliding window repair combined with column locality.

4.4.2 Convolutional codes with row locality

In this section we introduce and study another notion of locality for convolutional codes. Given a convolutional code $\mathcal{C}$, define the *ith row code* $\mathcal{C}^{(i)}_{[0,j]}, 1 \leqslant i \leqslant n$ truncated at jth time instant, $j \geqslant 0$, as follows:

$$\mathcal{C}^{(i)}_{[0,j]} = \{c^{(i)}_{[0,j]} = (c^{(i)}_0, \ldots, c^{(i)}_j) \mid c \in \mathcal{C}\}. \tag{4.23}$$

Definition 11. *We say that* $\mathcal{C}$ *has* (r, δ) *row locality if for all* $t \geqslant 0$ *the codes*

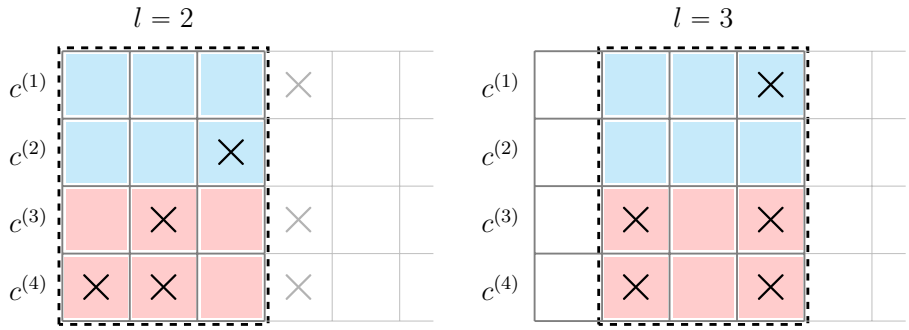

Figure 4.2: SLIDING WINDOW REPAIR WITH COLUMN LOCALITY. Suppose $\mathcal{C}$ is a $(4, 2)$ convolutional code with $(1, 2)$ column locality and $d_2^c = 6$. Let $(c^{(1)}, c^{(2)}, c^{(3)}, c^{(4)}) \in \mathcal{C}$ be a codeword, where the crosses denote erasures, and different repair groups in the columns are shown in different colors. At time $l = 2$, the window of length $j + 1 = 3$ contains 4 erasures. Of these, the symbols $c_0^{(4)}, c_2^{(2)}$ can be recovered within their repair groups. Then for $l = 3$ the window contains 5 erasures, of which two in the first column can be repaired from the other symbols in the window, while the symbol $c_3^{(1)}$ can be recovered locally.

$\mathcal{C}_{[t,t+j]}^{(i)}, 1 \leqslant i \leqslant n$ *have the* (r, δ) *locality property, where* $j \geqslant 0$ *is fixed.*

In the case of tailbiting codes, it is more convenient to give this definition in the following form, which will also be used in our constructions below.

Definition 12. *Let* $j \geqslant 0$ *be fixed. A convolutional code* $\mathcal{C}$ *has* (r, δ) *row locality at time* j *if every code* $\mathcal{C}_{[0,j]}^{(i)}, 1 \leqslant i \leqslant n$ *has* (r, δ) *locality.*

We give two examples of repair with row locality. Namely, Figure 4.3 illustrates Def. 11 while Figure 4.4 applies to the case of tailbiting codes and Def. 12. Let us stress that whenever local repair by rows is not possible, we fall back on sliding window repair relying on the column distance of the truncated code.

The problem that we address is to construct convolutional codes with locality and large column distance. This is similar to the problem studied in [52] and also to the case of block codes with locality and large minimum distance. We begin with deriving an upper bound on the column distance of the truncated code with either

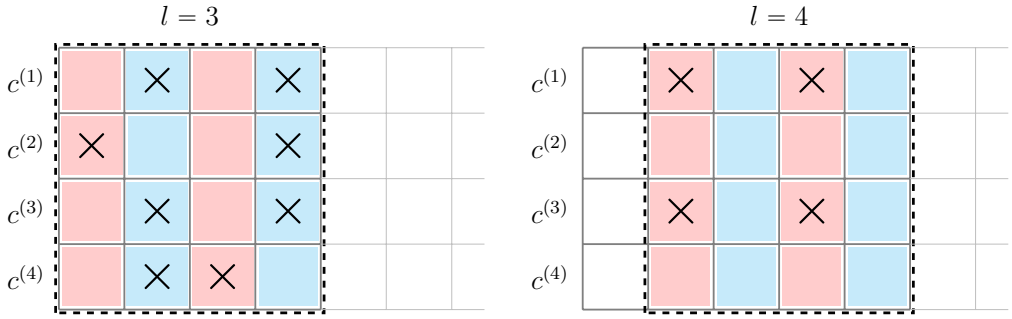

Figure 4.3: Sliding window repair with row locality. Suppose $\mathcal{C}$ is a $(4, 2)$ convolutional code with $(1, 2)$ row locality $d_3^c = 8$. Let $(c^{(1)}, c^{(2)}, c^{(3)}, c^{(4)}) \in \mathcal{C}$ be a codeword, where the crosses denote erasures, and different repair groups in the rows are shown in different colors. At time $l = 3$, by row locality, the symbols $c_0^{(2)}, c_3^{(2)}, c_1^{(4)}, c_2^{(4)}$ can be recovered from the other symbols in their respective repair groups. At time $l = 4$ there are four erasures in the window of length four, which is smaller than d_3^c, so they are recoverable. Thereafter the two remaining erasures can be recovered relying on row locality.

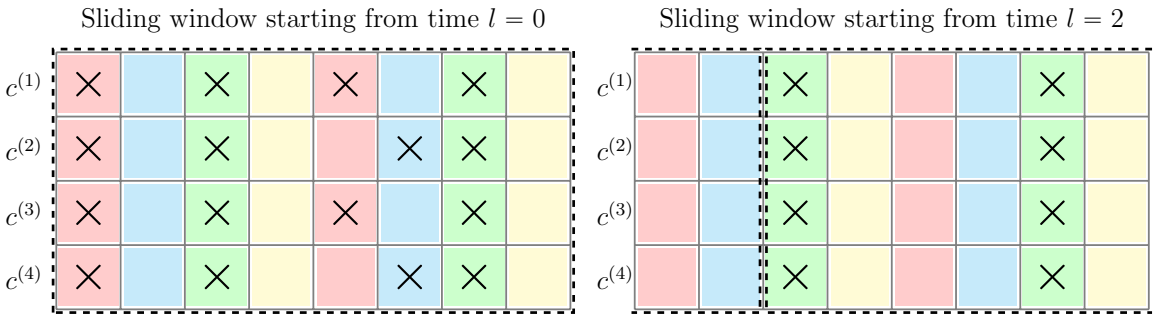

Figure 4.4: Sliding window repair with row locality for tailbiting codes. Suppose $\mathcal{C}$ is a $(4, 2)$ unit memory tailbiting convolutional code with $(1, 2)$ row locality and $d_7^c = 16$. Let $(c^{(1)}, c^{(2)}, c^{(3)}, c^{(4)}) \in \mathcal{C}$ be a codeword, where the crosses denote erasures, and different repair groups in the rows are shown in different colors. There are a total of 16 erasures. First we engage row locality to repair symbols $c_0^{(2)}, c_5^{(2)}, c_0^{(4)}, c_5^{(4)}$, whereupon 12 erasures remain. In the sliding window starting from time $l = 0$, the remaining two erasures in the first column can be recovered. After that, $c_4^{(1)}, c_4^{(3)}$ can be recovered locally.

Next, we move the sliding window to start at time $l = 2$. (Note that the window is wrapped around.) The erasures in the first column of this window can be recovered since the number of erasures is smaller than the column distance, and after that the remaining erasures can be recovered relying on row locality.

column or row locality property.

Proposition 42. *(a) Let $\mathcal{C}$ be an (n, k) convolutional code with (r, δ) (column or row) locality. Then for any $j \geqslant 0$, the j-th column distance satisfies*

$$d_j^c \leqslant (n-k)(j+1) + \delta - \left\lceil \frac{k}{r} \right\rceil (\delta - 1). \tag{4.24}$$

(b) Equality in (4.24) implies that for all $i \leqslant j$, the i-th column distance satisfies

$$d_i^c = (n-k)(i+1) + \delta - \left\lceil \frac{k}{r} \right\rceil (\delta - 1). \tag{4.25}$$

The proof of Proposition 42 is given in Appendix C.3.

Part (b) of Proposition 42 is similar to the propagation of the column distance optimality property in the case of general convolutional codes proved in [24]. Namely, the Singleton bound implies that the column distance for all j satisfies

$$d_j^c \leqslant (n-k)(j+1) + 1, \tag{4.26}$$

and equality for a given j implies that all the other distances $d_i^c, i \leqslant j$ also attain their versions of the Singleton bound with equality.

4.4.3 Convolutional codes and quasicyclic codes

A transformation between these two code families was constructed in [70], and it has led to a broader family of convolutional codes and trellises now known as *tailbiting codes* (tailbiting trellises) [37]. An $(n(m+1), k(m+1))$ quasicyclic code can be defined by a generator matrix

$$G = (G_{ij}),\ i = 0, \ldots, k-1, j = 0, \ldots, n-1$$

where each G_{ij} is an $(m+1) \times (m+1)$ circulant matrix. With a given matrix G_{ij} we associate a polynomial $g_{ij}(D) = \sum_{l=0}^{m} g_l D^l$, where $(g_0, \ldots, g_m)$ is the first row of the matrix. Then the $k \times n$ generator matrix $G(D) = (g_{ij}(D))$ defines an (n, k) convolutional code. The authors of [70] showed that if one takes the input sequences of the convolutional code in the form

$$u(D) = \sum_{l=-M}^{m} u_l D^l \text{ such that } u_{-s} = u_{m+1-s} \text{ for } s = 1, \ldots, M, \tag{4.27}$$

then the convolutional code is equivalent to the quasicyclic code defined above. In other words, the quasicyclic code can be encoded convolutionally, and the convolutional code with "symmetric" input sequences as in (4.27) is exactly the quasicyclic code.

4.4.4 A family of tailbiting convolutional codes with row locality

We come to the main result of this section, which is a construction of a family of convolutional codes with (r, δ) row locality. This family of codes has the largest possible minimum distance for the truncated code $\mathcal{C}_{[0,j]}$, however we stop short of showing that the j-th column distance attains (4.24) with equality. The construction is achieved by exploiting a connection between quasi-cyclic codes and convolutional codes discussed above. In high-level terms, our plan is to construct a cyclic code from its set of zeros chosen according to the procedure in Sec. 4.3, writing it in a quasicyclic form (via a circulant generator matrix) and to construct a convolutional code using the technique discussed above. We note that the lower bounds on the column distance of the codes constructed here and in [52] coincide. At the same time, the field size needed for our construction is linear in the output length n of the code at each time instant whereas the construction in [52] requires exponentially sized alphabet.

Let us first construct an $(n(j+1), k(j+1))$ cyclic LRC codes with (r, δ) locality. We proceed similarly to Sec. 4.3.1. We will need a few assumptions regarding the parameters of the code. Let $j \geqslant 0$ be such that $k \leqslant j+1 \leqslant n$ and that $j+1 = (r+\delta-1)\nu$ where $\nu \geqslant 1$. Let $\mathbb{F}_q$ be a finite field such that $n(j+1) \mid (q-1)$ and let $\alpha \in \mathbb{F}_q$ be a primitive root of unity of order $n(j+1)$ in $\mathbb{F}_q$.

The set of zeros of the code is obtained as follows. let $\mathcal{Z}_1 = \{1, \ldots, \delta-1\}$.

Using (4.7) and setting $\mathcal{D}_2 = \mathcal{Z}_1$, we have

$$\mathcal{Z}_2 = \bigcup_{l=0}^{\nu-1} (\mathcal{Z}_1 + l(r+\delta-1)). \tag{4.28}$$

Further, let $\mathcal{L}_3 = \bigcup_{l=0}^{n-1} (\mathcal{Z}_2 + l(j+1))$ and let $\mathcal{D}_3 = \{1, \ldots, \delta_3 - 1\}$, where

$$\delta_3 = (n-k)(j+1) + \delta - \left\lceil \frac{k(j+1)}{r} \right\rceil (\delta - 1). \tag{4.29}$$

Finally, put $\mathcal{Z} = \mathcal{L}_3 \cup \mathcal{D}_3$ and let $\mathcal{B}$ be the cyclic code with generator polynomial $g_{\mathcal{B}}(x) = \prod_{t \in \mathcal{Z}} (x - \alpha^t)$. Note for future use that the complement of the set $\mathcal{D}_3$ in the set of exponents of α has cardinality

$$|\bar{\mathcal{D}}_3| = n(j+1) - (\delta_3 - 1) = k(j+1) + \left(\left\lceil \frac{k(j+1)}{r} \right\rceil - 1 \right) (\delta - 1). \tag{4.30}$$

In the next theorem we give an explicit representation of the code $\mathcal{B}$ in quasicyclic form, and we also specify its locality properties.

Theorem 43. *(a) The code $\mathcal{B}$ is an $(n(j+1), k(j+1))$ optimal LRC code with (r, δ) locality, and the punctured codes*

$$\mathcal{B}_l = \{(c_l, c_{l+n}, \ldots, c_{l+nj}) \mid (c_0, \ldots, c_{n(j+1)-1}) \in \mathcal{B}\}, \quad l = 0, \ldots, n-1,$$

are LRC codes of length $j+1$ with (r, δ) locality.

(b) Furthermore, if $k|n$, then the code $\mathcal{B}$ is equivalent to a code with generator

matrix G given by

$$G = (G_{il}), i = 0, \ldots, k-1, l = 0, \ldots n-1,$$

where every G_{il} is a $(j+1) \times (j+1)$ circulant matrix. For every $i = 0, 1, \ldots, k-1$ the matrices G_{il} satisfy

$$G_{il} = \begin{cases} I_{j+1} & \text{if } l = in/k \\ 0 & \text{if } l \in \{0, n/k, \ldots, n-n/k\} \backslash \{in/k\}. \end{cases}$$

Proof. (a) We note that the set of zeros of the code $\mathcal{B}$ is partitioned into segments of length $r+\delta-1$, i.e., has the structure of the set $\mathcal{L}$ in (4.4). In other words, the generator polynomial of $\mathcal{B}$ satisfies

$$\prod_{t \in \mathcal{L}} (x - \alpha^t) | g_{\mathcal{B}}(x), \text{ where } \mathcal{L} = \bigcup_{s=0}^{n\nu-1} (\mathcal{Z}_1 + s(r+\delta-1)).$$

Therefore, Lemma 35 implies that the code $\mathcal{B}$ has (r, δ) locality. To compute the dimension of the code $\mathcal{B}$ we count the number of its nonzeros. They are all located in $\bar{\mathcal{D}}_3$. This is a consecutive segment of exponents, and by (4.28), within each whole subsegment of length $r+\delta-1$ in it there are r nonzeros. Once all such segments are accounted for, there may be an incomplete segment left, which contains $\min(|\bar{\mathcal{D}}_3| - \lfloor \frac{|\bar{\mathcal{D}}_3|}{r+\delta-1} \rfloor (r+\delta-1), r)$ zeros. As easily checked, the total number of nonzeros in either case is $k(j+1)$, which is therefore the dimension of the code $\mathcal{B}$. The distance of $\mathcal{B}$ is at least δ_3, and the bound (1.6) implies that the code $\mathcal{B}$ has the largest possible

distance for the chosen locality parameters.

Examining the structure of zeros of the punctured codes $\mathcal{B}_l$, we observe that they satisfy the assumptions of Lemma 35, and thus the punctured codes $\mathcal{B}_l$ also have (r, δ) locality. Indeed, let $l = 0$. By Lemma 35, Eq. (4.5), the code $\mathcal{B}^{(0)}$ given by

$$\mathcal{B}^{(0)} = \{(c_0, c_{n\nu}, \ldots, c_{(r+\delta-2)n\nu}) \mid (c_0, \ldots, c_{n(j+1)-1}) \in \mathcal{B}\}$$

has dimension at most r and minimum distance at least δ. This implies that the coordinates that are multiples of $n\nu$ isolate a repair group of the code $\mathcal{B}_0$. By shifting this set of coordinates to the right by n positions, we obtain another repair group of $\mathcal{B}_0$, which is disjoint from the first one. After several more shifts we will reach the set of coordinates $\{n(\nu-1), n(\nu-1)+n\nu, \ldots, n(\nu-1)+(r+\delta-2)n\nu\}$. The collection of the sets constructed along the way forms a partition of the support of $\mathcal{B}_0$ into disjoint repair groups. The same argument works for every code $\mathcal{B}_l, 1 \leqslant l \leqslant n-1$ whose repair groups are formed by shifting the repair groups of $\mathcal{B}_0$ to the right by l positions. This concludes the proof of Part (a).

Let us prove Part (b). Recalling the discussion in the beginning of Sec. 4.2.1, it is possible to represent the generator matrix G of the code to have rows of the form

$$((\alpha^t)^{n(j+1)-1}, (\alpha^t)^{n(j+1)-2}, \ldots, 1), \quad t \in \{0, \ldots, n(j+1)-1\} \backslash \mathcal{Z}, \tag{4.31}$$

(note the inverse order of the exponents). Let

$$\mathcal{I} = \{0, n/k, \ldots, n(j+1) - n/k\}$$

be a subset of coordinates. As before, we label the columns of G by the exponents of α from 0 to $n(j+1)-1$ and consider a square $k(j+1) \times k(j+1)$ submatrix $G_{\mathcal{I}}$ formed of the columns with indices in $\mathcal{I}$. We claim that $G_{\mathcal{I}}$ is invertible. Indeed, the rows of $G_{\mathcal{I}}$ have the form

$$\alpha^{-t}((\alpha^{tn/k})^{k(j+1)}, (\alpha^{tn/k})^{k(j+1)-1}, \ldots, \alpha^{tn/k}), \quad t \in \{0, \ldots, n(j+1)-1\} \backslash \mathcal{Z},$$

and thus it forms a Vandermonde matrix generated by the set $(\alpha^{tn/k})$ for all t outside the set of zeros $\mathcal{Z}$. We will assume that the submatrix $G_{\mathcal{I}} = I_{k(j+1)}$ (the identity matrix) and continue to use the notation G for the resulting generator matrix of the code.

Let $g_{i,0}, g_{i,1}, \ldots, g_{i,n(j+1)-1}$ be the ith row of G, where $i \in \{0, \ldots, k-1\}$. With an outlook of constructing convolutional codes later in this section, define the polynomials

$$g_i(D) = \sum_{s=0}^{n(j+1)-1} g_{i,s} D^s$$

$$g_{i,l}(D) = \sum_{s=0}^{j} g_{i,ns+i} D^i, \quad l = 0, 1, \ldots, n-1. \tag{4.32}$$

Then we have

$$g_i(D) = \sum_{l=0}^{n-1} D^l \sum_{s=0}^{j} g_{i,ns+l} D^{ns} = \sum_{l=0}^{n-1} D^l g_{i,l}(D^n),$$

Since $G_{\mathcal{I}}$ is the identity matrix, we have $g_{i,in/k}(D) = 1$ and $g_{i,l}(D) = 0$ for $l \in \{0, n/k, \ldots, n - n/k\} \backslash \{in/k\}$.

To write the generator matrix in the circulant form given in the statement, we need to form the matrices G_{il}. This is accomplished by writing the coefficients of $g_{i,j}(D)$ as the first row of G_{il} and filling the rest of this matrix by consecutive cyclic shifts to the right. This yields the following $(j+1) \times n(j+1)$ matrix

$$\begin{pmatrix} G_{i,0} & G_{i,1} & \cdots & G_{i,n-1} \end{pmatrix}. \tag{4.33}$$

Note that each row in this matrix is a codeword of the code (equivalent to) $\mathcal{B}$. Finally, the matrix

$$G = \begin{pmatrix} G_{0,0} & G_{0,1} & \cdots & G_{0,n-1} \\ G_{1,0} & G_{1,1} & \cdots & G_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ G_{k-1,0} & G_{k-1,1} & \cdots & G_{k-1,n-1} \end{pmatrix}. \tag{4.34}$$

formed of the rows (4.33) for $i = 0, \ldots, k-1$ generates a code equivalent to the code $\mathcal{B}$. □

This theorem gives an explicit representation of $\mathcal{B}$ as a quasicyclic code, and we

can use this representation to construct a convolutional code following the method in Sec. 4.4.3. Let $G(D) = (g_{i,l}(D))$ be a $k \times n$ generator matrix where $g_{i,l}(D)$ is defined in (4.32). Since $\deg(g_{i,l}(D)) \leqslant j$, we conclude that the memory of the generator matrix $G(D)$ is $M \leqslant j$. Having (4.27) in mind, define an (n, k) tailbiting convolutional code over $\mathcal{C} = \mathcal{C}_{[0,j]} \in \mathbb{F}_q[D]$ as a set of sequences

$$\mathcal{C} = \left\{ c(D) \mid c(D) = u(D)G(D); u(D) = \sum_{s=-M}^{j} u_s D^s; u_{-s} = u_{j+1-s}, s = 1, \ldots, M \right\}.$$

Here $j, k-1 \leqslant j \leqslant n-1$ is any integer such that $(r+\delta-1)|(j+1)$ and $k|n$.

The next theorem states the main properties our construction.

Theorem 44. *The code $\mathcal{C}$ has (r, δ) row locality. When viewed as a block code, the minimum distance of $\mathcal{C}$ attains the bound (1.6).*

Proof. For $l = 0, \ldots, n-1$, we have

$$c^{(l)}(D) = \sum_{i=0}^{k-1} u^{(i)}(D) g_{i,l}(D).$$

Furthermore, since $u_{-s} = u_{j+1-s}$ for $s = 1, \ldots, M$, we have the following relation

$$c^{(l)}(D) = \sum_{i=0}^{k-1} u^{(i)}(D) g_{i,l}(D) \mod (D^{j+1} - 1).$$

In other words, we have

$$\left(c^{(0)} c^{(1)} \ldots c^{(n-1)} \right) = \left(u^{(0)} u^{(1)} \ldots u^{(k-1)} \right) G,$$

where the matrix G is defined in (4.34). This implies that the codes $\mathcal{C}^{(l)}$ are exactly the codes $\mathcal{B}_l$ defined in Theorem 43(a), viz., $\mathcal{C}^{(l)} = \mathcal{B}_l$ for $l = 0, \ldots, n-1$. Since the code $\mathcal{B}_l$ has (r, δ) locality for $l = 0, \ldots, n-1$, we conclude that the code $\mathcal{C}$ has (r, δ) row locality. Concluding, we have established that the convolutional code $\mathcal{C}$ has (r, δ) row locality.

As a block code, $\mathcal{C}$ is equivalent to the code $\mathcal{B}$, which proves the last claim of the theorem. □

The large minimum distance of the code $\mathcal{C}$ is related to the performance of the (hard decision) Viterbi decoding of the code $\mathcal{C}$, and is therefore of interest in applications.

As remarked earlier, the constructed codes stop short of attaining the bound (4.24), and thus cannot be claimed to be optimal. Of course, as observed after Def. 10, the jth column distance of the code $\mathcal{C}$ is at least the minimum distance of the code $\mathcal{B}$, given by (4.29), but a more precise estimate remains an open question. Nevertheless, we believe that extension of the basic construction of LRC codes to the case of convolutional codes carries potential for future research into their structure. In particular, the algebraic machinery of quasicyclic codes of [42, 48] could lead to new constructions, and it may also be possible to further extend these studies to codes over ring alphabets [49].

4.5 Bi-cyclic codes with availability

The H-LRC codes constructed in Sec. 4.3 rely on h embedded recovering sets for each code coordinate, which are not disjoint. In this section we consider LRC codes with t *disjoint* recovering sets for each code coordinate, i.e., LRC codes with availability t (here we do not pursue a hierarchy of locality). This arises when the data is simultaneously requested by a large number of users, which suggests that the erased coordinate afford recovery from several nonoverlaping recovering sets in order to increase data availability.

Below we limit ourselves to the case $t = 2$. We say that two partitions $\mathcal{P}_1, \mathcal{P}_2$ of the coordinates of the code are *orthogonal* (or transversal) if $|P_1 \cap P_2| \leqslant 1$ for any $P_1 \in \mathcal{P}_1, P_2 \in \mathcal{P}_2$ and every coordinate is contained in a pair of subsets $X \in \mathcal{P}_1, Y \in \mathcal{P}_2$. Orthogonal partitions enable multiple disjoint recovering sets and were used in [71] to construct codes with availability. A simple observation made in [71] is that product codes naturally yield orthogonal partitions, and it is possible to use products of one-dimensional cyclic codes to support this structure. A drawback of this approach is that product codes result in rather poor parameters of LRC codes with availability, in particular the rate of the resulting codes is low (although the alphabet is small compared to the code length [34, 71]). It is well known that the rate of product codes can be increased with no loss to the distance by passing to generalized concatenations of codes [7]. In this section we use a particular case of this construction given in [62] and sometimes called *hyperbolic codes*. The resulting LRC codes with availability have the same distance guarantee as simple product

codes while having a much higher rate. As above, our starting point is the general method of Theorem 34, and we proceed similarly to Eq. (4.7).

We start with a finite field $\mathbb{F}_q$ and assume that the code length n divides $q-1$. We further choose the size of the repair groups to be r_1, r_2 and suppose that $0 < r_1 \leqslant r_2$ and $(r_1+1)|n$ and $(r_2+1)|n$. Further, let $\nu_1 = n/(r_1+1)$ and $\nu_2 = n/(r_2+1)$. Let $\alpha \in \mathbb{F}_q$ be a primitive n-th root of unity. To simplify the expressions below, we will construct codes with $\delta_1 = \delta_2 = 2$. To construct the defining set $\mathcal{Z}$ of our code, let

$$\mathcal{L}_1 = \varnothing, \quad \mathcal{D}_1 = \{(0,0)\}, \quad \mathcal{Z}_1 = \mathcal{L}_1 \cup \mathcal{D}_1. \tag{4.35}$$

Let us fix the designed distance of the code $\mathcal{C}$ to be $\delta \geqslant 2$. Define

$$\begin{aligned}
\mathcal{L}_{2,1} &= \bigcup_{l_1=0}^{\nu_1-1} \bigcup_{j=0}^{n-1} (\mathcal{Z}_1 + (l_1(r_1+1), j)), \\
\mathcal{L}_{2,2} &= \bigcup_{i=0}^{n-1} \bigcup_{l_2=0}^{\nu_2-1} (\mathcal{Z}_1 + (i, l_2(r_2+1))), \ \mathcal{L}_2 = \mathcal{L}_{2,1} \cup \mathcal{L}_{2,2}, \\
\mathcal{D}_2 &= \{(i,j) \mid (i+1)(j+1) < \delta\}, \ \mathcal{Z}_2 = \mathcal{L}_2 \cup \mathcal{D}_2.
\end{aligned}$$

Note that zeros are now indexed by pairs of exponents, and pairs are added element-wise. Finally, put $\mathcal{Z} = \mathcal{Z}_2$.

Consider a two-dimensional cyclic code $\mathcal{C} = \langle g(x,y) \rangle$ of length n^2, where

$$g(x,y) = \prod_{(i,j) \in \mathcal{Z}} (x - \alpha^i)(y - \alpha^j). \tag{4.36}$$

Lemma 45. *The code $\mathcal{C}$ has two disjoint recovering sets of size r_1 and r_2 for every coordinate.*

Proof. The proof relies on Lemma 35. For $c \in \mathcal{C}$, let us write $c = (c_{i,j})$ where $0 \leqslant i \leqslant n-1, 0 \leqslant j \leqslant n-1$. Fix j and let $\mathcal{C}^{(1)} = \{(c_{0,j}, c_{\nu_1,j}, \ldots, c_{r_1\nu_1,j}) \mid c \in \mathcal{C}\}$ and let $\mathcal{L}_{2,1}^{j} = \bigcup_{l_1=0}^{\nu_1-1}(\mathcal{Z}_1 + (l_1(r_1+1), j))$. The generator polynomial for the punctured code $\{(c_{0,j}, c_{1,j}, \ldots, c_{n-1,j}) \mid c \in \mathcal{C}\}$ is given by $y^j \prod_{(i,j)\in\mathcal{Z}}(x-\alpha^i)$, which is divisible by $\prod_{(i,j)\in\mathcal{L}_{2,1}^{j}}(x-\alpha^i)$. Then, by arguments similar to Lemma 35 we conclude that the code $\mathcal{C}^{(1)}$ has dimension at most r_1 and distance at least 2. Since the code is cyclic in both dimensions, we also claim that every symbol of the code $\mathcal{C}$ has a recovering set of size at most r_1 for one erasure.

Repeating the above arguments for a fixed index i, we isolate another recovering of size r_2 for every coordinate. Furthermore, the two recovering sets are disjoint by construction. □

To estimate the distance, recall the following result about bicyclic codes.

Lemma 46 (Hyperbolic bound [62])**.** *Suppose that the defining set of zeros of an $n \times n$ bicyclic code contains a subset given by $\mathcal{Z} = \{(i,j) : (i+1)(j+1) < d\}$. Then the distance of the code is at least d.*

Thus, the distance of the code $\mathcal{C}$ constructed above is at least δ, and its dimension $\dim(\mathcal{C}) = n^2 - |\mathcal{Z}|$. Let us estimate the dimension from below. We have

$$|\mathcal{D}_2| = \sum_{j=0}^{\delta-2} \left\lfloor \frac{\delta}{j+1} \right\rfloor \leqslant \delta \sum_{j=1}^{\delta-1} \frac{1}{j}$$

$$\leqslant \delta \left(1 + \int_1^{\delta-1} \ln(x-1)dx\right)$$
$$\leqslant \delta(1 + \ln(\delta - 1)).$$

Therefore, we have

$$\begin{aligned} n^2 - |\mathcal{Z}| &\geqslant n^2 - |\mathcal{L}_2| - |\mathcal{D}_2| \\ &= n^2 - (r_1 + r_2 + 1)\nu_1\nu_2 - \delta(1 + \ln(\delta - 1)) \\ &= \frac{n^2 r_1 r_2}{(r_1+1)(r_2+1)} - \delta(1 + \ln(\delta - 1)) \end{aligned} \tag{4.37}$$

To compare this estimate of the dimension with product codes, let $\mathcal{C}'$ be a direct product of two cyclic LRC codes of length n with locality r and distance $\sqrt{\delta}$. The defining set of this code are given by $\mathcal{Z}' = \mathcal{L}' \cup \mathcal{D}'$, where

$$\mathcal{L}' = \{(i,j) \mid i,j = 1 \bmod (r+1)\},$$
$$\mathcal{D}' = \{(i,j) \mid i,j = 1, \ldots, \sqrt{\delta} - 1\}.$$

Choosing $r_1 = r_2 = r$ in our construction, we have $\mathcal{L}_2 + (1,1) = \mathcal{L}'$. It is also easy to see that $|\mathcal{Z}| \leqslant |\mathcal{Z}'|$ and thus, $\dim(C) \geqslant \dim(\mathcal{C}')$.

Let us write out an estimate for the rate of the constructed codes. Putting $r_1 = r_2 = r$ in (4.37), we obtain for the rate of the code $\mathcal{C}$ the following estimate in terms of the relative distance $\theta = \delta/n^2$:

$$R \geqslant 1 - \frac{2}{r+1} + \frac{1}{(r+1)^2} - \theta(1 + \ln(\delta - 1)).$$

The best known upper bound on the rate of LRC codes with locality r, availability t, and relative distance θ [41] has the form

$$R \leqslant \frac{r-1}{r}(1-\theta) - o(1).$$

The gap between this bound and the lower estimate implied by our construction is roughly $O(\frac{1}{r} + \theta \ln \delta)$.

Example 3. *Let $r_1 = 2$, $r_2 = 6$, and $\delta = 9$. Our construction gives rise to an LRC code with availability two, of length $n^2 = 441$, dimension $k = 246$, and distance $d \geqslant 9$, over a finite field $\mathbb{F}_q$ such that $21|(q-1)$ (for example, we can take $q = 64$). To compare these codes with the product construction, let us choose the column codes and row codes of length 21 and distance 3, and let us take the maximum dimension of codes as given by the bound (1.4) for locality 2 and 6. This gives $k_1 = 13, k_2 = 17$ and the overall dimension $k = 221$, lower than above.*

An extension of the construction in this section to the case of $t \geqslant 2$ can be easily obtained via t-dimensional cyclic codes and the general hyperbolic bound (the generalized concatenation of codes). Furthermore, using procedures similar to those used in (4.6) and (4.7), our construction can be generalized to $h \geqslant 2$ levels of hierarchy such that the local codes in each of the h levels have availability $t \geqslant 2$.

Chapter 5: Conclusion

We conclude the dissertation with some open problems. Specifically, Sec. 5.1 mentions a few open questions for the repair problem of RS codes and Sec. 5.2 points out some future directions for rack-aware MSR codes. In Sec. 5.3, we discuss the problem of convolutional codes with locality, which remains largely open.

5.1 RS codes with optimal repair

In Chapter 2 we showed that error correction is feasible in the original code family of [77] without the increase of the extension degree of the symbol field of the code (the node size). Namely, codes from [77] use extension degree $l = (d-k+1)L$, where L is the product of the first n distinct primes in an arithmetic progression,

$$L = \left(\prod_{\substack{i=1 \\ p_i \equiv 1 \bmod (d-k+1)}}^{n} p_i \right).$$

The lower bound on l from [77], necessary for repair of a single node, has the form $l \geqslant \prod_{i=1}^{k-1} p_i$, where p_i is the i-th smallest prime. Asymptotically for fixed $d-k$ and growing n we obtain the following bounds on the node size: $\Omega(k^k) \leqslant l \leqslant O(n^n)$. Essentially the same node size is used in this chapter for repair with error correction.

At the same time, the explicit RS code family with optimal access that we construct comes at the expense of larger node size, namely $l = (d-k+1)^n L$. Since there is an optimal-access repair scheme for every scalar MSR code, this leaves a gap between what is known explicitly and what is shown to be possible, which represents a remaining open question related to the task of optimal repair of RS codes.

Another direction of interest is to extend our approach to optimal error correction and optimal access to the problem of repairing RS codes with multiple erasures.

5.2 Rack-aware MSR codes

In Chapter 3 we presented various families of rack-aware MSR codes, including both vector codes, scalar codes, and low-access codes. Nevertheless, a few problems remain open.

Following the discussion in Sec. 3.5.2, one open problem is to extend vector MSR code constructions in the literature that support optimal error correction for the homogeneous model to the rack-aware storage model or to come up with novel constructions that afford optimal error correction for the rack model.

More importantly, it would be interesting to close the gap of the access cost between the constructions and the bound for the rack-aware storage model, in particular, for the case when the rack size u does not divide the code dimension k.

Another direction worth further investigating is to understand the case of multiple erasures (node failures). When there are two or more failed nodes in the rack model, one has to distinguish cases between the failed nodes located in a single

rack and dispersed among multiple racks, which would give rise to a delicate bound for the repair bandwidth. However, we believe that our ideas presented in this chapter will serve a starting point even for constructing rack-aware MSR codes for multiple erasures.

5.3 Codes with locality

In Chapter 4 we addressed several variants of the problem of codes with locality. In particular, we constructed a family of tailbiting convolutional codes with row locality.

Apart from understanding the optimal trade-off among the parameters for codes with availability, convolutional codes with locality form an interesting direction for future work. In particular, it would be of interest to understand if the upper bound (4.24) for the column distance is tight or not for convolutional code with locality. It would be of great interest even if one could only prove existence of convolutional codes with locality attaining the column distance in (4.24).

Moreover, in addition to column and row locality, can one construct convolutional codes with a general type of locality? Further, as there are various types of distance measures for convolutional codes for different motivations (see [37] for more details), can one derive bounds for other types of distance measures and construct codes to meet these bounds? Would the connection between quasicyclic codes and convolutional codes we utilized herein be helpful for constructing convolutional codes with locality for other types of distance measures?

Appendix A: Omitted Proofs in Chapter 2

A.1 Proof of Proposition 7

First we present the proof for the case $h = 0$ (strictly speaking, we do not have to isolate it, but it makes understanding the general case much easier). In this case, definition (2.18), (2.19) simplifies as follows. Let $f_0(x) = x^{p_1} - f(x)$. Write f_0 as

$$\begin{aligned} f_0(x) &= a_0 + a_1 x + a_2 x^2 + \cdots + a_{p_1-1} x^{p_1-1} \\ &= \sum_{q=0}^{(p_1-1)/s-1} x^{qs} f_{0,q}(x), \end{aligned}$$

where

$$\begin{aligned} f_{0,0}(x) &= a_0 + a_1 x + \cdots + a_{s-1} x^{s-1} \\ f_{0,1}(x) &= a_s + a_{s+1} x + \cdots + a_{2s-1} x^{s-1} \\ &\cdots \\ f_{0,(p_1-1)/s-1}(x) &= a_{p_1-1-s} + a_{p_1-s} x + \cdots + a_{p_1-1} x^s, \end{aligned} \tag{A.1}$$

so that the degree of the last polynomial is $\leqslant s$ and the degrees of the remaining ones are $\leqslant s-1$. Obviously, we have

$$\alpha_1^{p_1} = f_0(\alpha_1) \tag{A.2}$$

$$= \sum_{q=0}^{(p_1-1)/s-1} \alpha_1^{qs} f_{0,q}(\alpha_1). \tag{A.3}$$

As before, we start with (2.7), which implies that for any polynomial $g \in F_1[x]$ of degree $\deg g \leqslant n-k-1$, we have

$$\operatorname{tr}(e_i v_1 g(\alpha_1) c_1) = -\sum_{j=2}^{n} g(\alpha_j) \operatorname{tr}(e_i v_j c_j). \tag{A.4}$$

Take $e_i = \alpha_1^{qs}$ and $g(x) = x^t f_{0,q}(\alpha_1)$ and sum on q on the left, then from (A.3) we obtain $\operatorname{tr}(v_1 \alpha_1^t f_0(\alpha_1) c_1)$. Summing on q on the right of (A.4) and using (A.2), we conclude that

$$\operatorname{tr}(v_1 \alpha_1^{p_1+t} c_1) = -\sum_{q=0}^{(p_1-1)/s-1} \sum_{j=2}^{n} \alpha_j^t f_{0,q}(\alpha_j) \operatorname{tr}(\alpha_1^{qs} v_j c_j) \tag{A.5}$$

for all $t = 0, 1, \ldots, n-k-s-1$, Note that the constraint $t \leqslant n-k-s-1$ is implied by the condition $\deg(g) = \deg(x^t f_{0,q}(x)) \leqslant n-k-1$ needed in order to use (A.4) (and (2.7)). Change the variable $t \mapsto (t-1)$ to write the last equation as

$$\operatorname{tr}(v_1 \alpha_1^{p_1-1+t} c_1) = -\sum_{q=0}^{(p_1-1)/s-1} \sum_{j=2}^{n} \alpha_j^{t-1} f_{0,q}(\alpha_j) \operatorname{tr}(\alpha_1^{qs} v_j c_j), \quad t = 1, 2, \ldots, n-k-s. \tag{A.6}$$

From (2.14) and the fact that

$$\bigcap_{u=1}^{s-1}\{u-s, u-s+1, \dots, u-s+n-k-1\} = \{-1, 0, 1, \dots, n-k-s\},$$

we obtain

$$\operatorname{tr}(\beta^u \alpha_1^{p_1-1+t} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^{t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j),$$

$$-1 \leqslant t \leqslant n-k-s, \quad 1 \leqslant u \leqslant s-1.$$

Summing these equations on $u = 1, 2, \dots, s-1$, we obtain the relation

$$\operatorname{tr}\Big(\sum_{u=1}^{s-1} \beta^u \alpha_1^{p_1-1+t} v_1 c_1\Big) = -\sum_{j=2}^{n}\sum_{u=1}^{s-1} \alpha_j^{t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j), \ -1 \leqslant t \leqslant n-k-s.$$

For each $t = 1, 2, \dots, n-k-s$ let us add this equation and (A.6). This gives $n-k-s$ relations of the form

$$\operatorname{tr}\Big(\sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1+t} v_1 c_1\Big) = -\sum_{j=2}^{n}\Big(\sum_{u=1}^{s-1} \alpha_j^{t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j) + \sum_{q=0}^{(p_1-1)/s-1} \alpha_j^{t-1} f_{0,q}(\alpha_j) \operatorname{tr}(\alpha_1^{qs} v_j c_j)\Big).$$

Observe that the left-hand side of this equation is the same as the left-hand side of (2.17). Therefore,

$$\sum_{j=2}^{n} \alpha_j^t \operatorname{tr}\Big(\sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1} v_j c_j\Big)$$

$$= \sum_{j=2}^{n} \Big(\sum_{u=1}^{s-1} \alpha_j^{t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j) + \sum_{q=0}^{(p_1-1)/s-1} \alpha_j^{t-1} f_{0,q}(\alpha_j) \operatorname{tr}(\alpha_1^{qs} v_j c_j) \Big),$$

for $1 \leqslant t \leqslant n-k-s$. Replacing $t-1$ with t in this equation, we obtain that

$$\sum_{j=2}^{n} \alpha_j^t \Big(\sum_{u=1}^{s-1} \alpha_j^{s-u+1} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j) + \sum_{q=0}^{(p_1-1)/s-1} f_{0,q}(\alpha_j) \operatorname{tr}(\alpha_1^{qs} v_j c_j) - \alpha_j \operatorname{tr}\Big(\sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1} v_j c_j \Big) \Big) = 0, \quad 0 \leqslant t \leqslant n-k-s-1.$$

By Proposition 1, the vector

$$\Big(\sum_{u=1}^{s-1} \alpha_j^{s-u+1} \operatorname{tr}\Big(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j\Big) + \sum_{q=0}^{(p_1-1)/s-1} f_{0,q}(\alpha_j) \operatorname{tr}(\alpha_1^{qs} v_j c_j) - \alpha_j \operatorname{tr}\Big(\sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1} v_j c_j \Big), j = 2, \ldots, n \Big) \quad \text{(A.7)}$$

is contained in a GRS code of length $n-1$ and dimension $s+k-1$. This proves the case $h=0$ of the proposition.

Now let us consider the general case $0 \leqslant h \leqslant s-1$. From (2.18) and (2.19) we obtain

$$\alpha_1^{p_1+h} = f_h(\alpha_1) = \sum_{q=0}^{(p_1-1)/s-1} \alpha_1^{qs} f_{h,q}(\alpha_1). \quad \text{(A.8)}$$

This relation enables us to use the argument that yielded (A.5) above: Take $e_i = \alpha_1^{u+qs}\beta^u$ and $g(x) = x^t f_{h,q}(x)$ in (A.4) and sum on $q = 0, 1, \ldots, (p_1-1)/s-1$. We obtain for $h = 0, \ldots, s-1$ and $u = 0, \ldots, s-1-h$

$$\operatorname{tr}(\alpha_1^{p_1+h+u+t} \beta^u v_1 c_1) = \sum_{q=0}^{(p_1-1)/s-1} \operatorname{tr}(\alpha_1^{qs+u+t} \beta^u f_{h,q}(\alpha_1) v_1 c_1)$$

$$= - \sum_{q=0}^{(p_1-1)/s-1} \sum_{j=2}^{n} \alpha_j^t f_{h,q}(\alpha_j) \operatorname{tr}(\alpha_1^{u+qs} \beta^u v_j c_j),$$

$$t = 0, 1, \ldots, n-k-s-1.$$

The restriction $t \leqslant n-k-s-1$ is imposed in the same way as in (A.5) (namely, it is necessary that $\deg(x^t f_{h,q}(x)) \leqslant n-k-1$). Replacing $h+u$ with h in the last equation, we obtain that

$$\operatorname{tr}(\alpha_1^{p_1+h+t} \beta^u v_1 c_1) = - \sum_{q=0}^{(p_1-1)/s-1} \sum_{j=2}^{n} \alpha_j^t f_{h-u,q}(\alpha_j) \operatorname{tr}(\alpha_1^{u+qs} \beta^u v_j c_j),$$

$$0 \leqslant h \leqslant s-1, \quad 0 \leqslant u \leqslant h, \quad 0 \leqslant t \leqslant n-k-s-1.$$

Let us sum these equations on $u = 0, 1, \ldots, h$ to obtain

$$\operatorname{tr}(\alpha_1^{p_1+h+t} \sum_{u=0}^{h} \beta^u v_1 c_1) = - \sum_{j=2}^{n} \sum_{q=0}^{(p_1-1)/s-1} \sum_{u=0}^{h} \alpha_j^t f_{h-u,q}(\alpha_j) \operatorname{tr}(\alpha_1^{u+qs} \beta^u v_j c_j),$$

$$0 \leqslant h \leqslant s-1, \ 0 \leqslant t \leqslant n-k-s-1.$$

Replacing t with $t-1$, we obtain that

$$\operatorname{tr}\left(\alpha_1^{p_1-1+h+t} \sum_{u=0}^{h} \beta^u v_1 c_1\right) = - \sum_{j=2}^{n} \sum_{q=0}^{(p_1-1)/s-1} \sum_{u=0}^{h} \alpha_j^{t-1} f_{h-u,q}(\alpha_j) \operatorname{tr}(\alpha_1^{u+qs} \beta^u v_j c_j),$$

$$0 \leqslant h \leqslant s-1, \ 1 \leqslant t \leqslant n-k-s. \quad \text{(A.9)}$$

According to (2.14) and the fact that

$$\bigcap_{u=h+1}^{s-1} \{u-s, u-s+1, \dots, u-s+n-k-1\} = \{-1, 0, 1, \dots, n-k-s+h\},$$

for $0 \leqslant h \leqslant s-1$, we have

$$\operatorname{tr}(\beta^u \alpha_1^{p_1-1+t} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^{t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j),$$

$$-1 \leqslant t \leqslant n-k-s+h,\ h+1 \leqslant u \leqslant s-1.$$

Replacing t with $t+h$, we have

$$\operatorname{tr}(\beta^u \alpha_1^{p_1-1+h+t} v_1 c_1) = -\sum_{j=2}^{n} \alpha_j^{h+t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j),$$

$$-h-1 \leqslant t \leqslant n-k-s,\ h+1 \leqslant u \leqslant s-1.$$

Summing these equations on $u = h+1, h+2, \dots, s-1$, we obtain

$$\operatorname{tr}\Big(\sum_{u=h+1}^{s-1} \beta^u \alpha_1^{p_1-1+h+t} v_1 c_1\Big) = -\sum_{j=2}^{n} \sum_{u=h+1}^{s-1} \alpha_j^{h+t-u+s} \operatorname{tr}(\beta^u \alpha_1^{u+p_1-s-1} v_j c_j),$$

$$-h-1 \leqslant t \leqslant n-k-s.$$

Finally, adding together this equation and (A.9), we obtain that

$$\operatorname{tr}\Big(\sum_{u=0}^{s-1} \beta^u \alpha_1^{p_1-1+h+t} v_1 c_1\Big)$$

$$= -\sum_{j=2}^{n}\sum_{q=0}^{(p_1-1)/s-1}\sum_{u=0}^{h}\alpha_j^{t-1}f_{h-u,q}(\alpha_j)\operatorname{tr}(\alpha_1^{u+qs}\beta^u v_j c_j)$$
$$-\sum_{j=2}^{n}\sum_{u=h+1}^{s-1}\alpha_j^{h+t-u+s}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_j c_j),$$

$$0\leqslant h\leqslant s-1,\ 1\leqslant t\leqslant n-k-s. \quad \text{(A.10)}$$

Going back to (2.17), let us perform the change $t\mapsto t+h$, then we obtain

$$\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1+h+t}v_1c_1\Big) = -\sum_{j=2}^{n}\alpha_j^{h+t}\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big),$$

$$-h\leqslant t\leqslant n-k-h-1. \quad \text{(A.11)}$$

For $t=1,2,\dots,n-k-s$ the left-hand sides of (A.10) and (A.11) coincide, and therefore,

$$\sum_{j=2}^{n}\alpha_j^{h+t}\operatorname{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big)$$
$$= \sum_{j=2}^{n}\sum_{q=0}^{(p_1-1)/s-1}\sum_{u=0}^{h}\alpha_j^{t-1}f_{h-u,q}(\alpha_j)\operatorname{tr}(\alpha_1^{u+qs}\beta^u v_j c_j)$$
$$+\sum_{j=2}^{n}\sum_{u=h+1}^{s-1}\alpha_j^{h+t-u+s}\operatorname{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_j c_j),$$

$$1\leqslant t\leqslant n-k-s.$$

Replacing t by $t+1$, we obtain that

$$\sum_{j=2}^{n}\alpha_j^{t}\Big(\sum_{q=0}^{(p_1-1)/s-1}\sum_{u=0}^{h}f_{h-u,q}(\alpha_j)\operatorname{tr}(\alpha_1^{u+qs}\beta^u v_j c_j)$$

$$+\sum_{u=h+1}^{s-1}\alpha_j^{h+1-u+s}\,\mathrm{tr}(\beta^u\alpha_1^{u+p_1-s-1}v_jc_j)-\alpha_j^{h+1}\,\mathrm{tr}\Big(\sum_{u=0}^{s-1}\beta^u\alpha_1^{p_1-1}v_jc_j\Big)\Big)=0,$$

$$0\leqslant t\leqslant n-k-s-1.$$

The proof is complete.

Appendix B: Omitted Proofs in Chapter 3

B.1 Proof of Proposition 24

Let $\mathcal{I} \subset \mathcal{R}, |\mathcal{I}| = \bar{k} - 1$ be a subset of helper racks. Since

$$(\bar{d} - \bar{k} + 1)u \geqslant d - k + 1,$$

Lemma 23 implies that

$$\sum_{i \in \mathcal{R} \setminus \mathcal{I}} \beta_i \geqslant l. \tag{B.1}$$

Let us sum the left-hand side on all $\mathcal{I} \subset \mathcal{R}, |\mathcal{I}| = \bar{k} - 1$:

$$\sum_{\substack{\mathcal{I} \subset \mathcal{R} \\ |\mathcal{I}| = \bar{k}-1}} \sum_{i \in \mathcal{R} \setminus \mathcal{I}} \beta_i = \sum_{i \in \mathcal{R}} \sum_{\substack{\mathcal{I} \subset \mathcal{R} \\ \mathcal{I} \not\ni i}} \beta_i = \binom{\bar{d} - 1}{\bar{k} - 1} \sum_{i \in \mathcal{R}} \beta_i.$$

Together with (B.1) we obtain

$$\binom{\bar{d} - 1}{\bar{k} - 1} \sum_{i_t \in \mathcal{R}} \beta_i \geqslant \binom{\bar{d}}{\bar{k} - 1} l$$

or

$$\sum_{i \in \mathcal{R}} \beta_i \geqslant \frac{\bar{d} l}{\bar{d} - \bar{k} + 1},$$

i.e., (3.2). Moveover, this bound holds with equality if and only if (B.1) holds with equality for every $\mathcal{I} \subset \mathcal{R}, |\mathcal{I}| = \bar{k} - 1$. Suppose for the sake of contradiction that the uniform download claim does not hold, and there is a rack i such that $\beta_i \neq l/\bar{s}$, for instance, $\beta_i < l/\bar{s}$, where $\bar{s} = \bar{d} - \bar{k} + 1$. Let $\mathcal{J} \subset \mathcal{R}, |\mathcal{J}| = \bar{s}, i \in \mathcal{J}$. There must be a rack $i_1 \in \mathcal{J}$ that contributes more than the average number of symbols, i.e., $\beta_{i_1} > l/\bar{s}$. Consider the subset $(\mathcal{J}\backslash\{i\}) \cup \{i_2\}$, where $i_2 \neq i$ is another element of $\mathcal{R}$ (which exists since $\bar{k} > 1$ implies $|\mathcal{J}| < |\mathcal{R}|$). We have that $\beta_{i_2} < l/\bar{s}$. Now take the subset $I = (\mathcal{J}\backslash\{i_1\}) \cup \{i_2\}$ and note that for it, (B.1) fails to hold with equality, a contradiction.

B.2 Proof of Proposition 26

Proof. Let $m' \in \mathcal{R}$ and let $\mathcal{I}$ be a subset of $u-v$ nodes in rack m', where $0 \leqslant v \leqslant u-1$. Let $\mathcal{J} \subseteq \mathcal{R}\backslash\{m'\}, |\mathcal{J}| = \bar{d} - \bar{k}$ be a subset of helper racks. These racks contain $u(\bar{d} - \bar{k}) = d - k + 1 - (u - v)$ nodes in these racks, and together with the nodes in the set $\mathcal{I}$ this forms a group of $d - k + 1$ nodes. Using Lemma 23, we have

$$\sum_{m \in \mathcal{J}} \sum_{e=1}^{u} \alpha_{m,e} + \sum_{e \in \mathcal{I}} \alpha_{m',e} \geqslant l. \tag{B.2}$$

Let us average over the $\binom{u}{u-v}$ choices of the set $\mathcal{I}$:

$$\binom{u}{u-v} \sum_{m \in \mathcal{J}} \sum_{e=1}^{u} \alpha_{m,e} + \sum_{\mathcal{I}:|\mathcal{I}|=u-v} \sum_{e \in \mathcal{I}} \alpha_{m',e} \geqslant \binom{u}{u-v} l.$$

Interchanging the sums in the second term on the left, we obtain

$$\binom{u}{u-v}\sum_{m\in\mathcal{J}}\sum_{e=1}^{u}\alpha_{m,e}+\binom{u-1}{u-v-1}\sum_{e=1}^{u}\alpha_{m',e}\geqslant\binom{u}{u-v}l.$$

or

$$\frac{u}{u-v}\sum_{m\in\mathcal{J}}\sum_{e=1}^{u}\alpha_{m,q}+\sum_{e=1}^{u}\alpha_{m',e}\geqslant\frac{u}{u-v}l. \tag{B.3}$$

Now let us average over the choice of $\mathcal{J}\subset\mathcal{R}\backslash\{m'\}$. Noting that

$$\sum_{\substack{\mathcal{J}\subset\mathcal{R}\backslash\{m'\}\\|\mathcal{J}|=\bar{s}-1}}\sum_{m\in\mathcal{J}}\sum_{e=1}^{u}\alpha_{m,e}=\binom{\bar{d}-2}{\bar{s}-2}\sum_{m\in\mathcal{R}\backslash\{m'\}}\sum_{e=1}^{u}\alpha_{m,e}$$

we obtain from (B.3)

$$\frac{u}{u-v}\binom{\bar{d}-2}{\bar{s}-2}\sum_{m\in\mathcal{R}\backslash\{m'\}}\sum_{e=1}^{u}\alpha_{m,e}+\binom{\bar{d}-1}{\bar{s}-1}\sum_{e=1}^{u}\alpha_{m',e}\geqslant\binom{\bar{d}-1}{\bar{s}-1}\frac{u}{u-v}l.$$

On account of the assumption that $\bar{d}\geqslant 2,\bar{s}\geqslant 2$ we find

$$\frac{u}{u-v}\sum_{m\in\mathcal{R}\backslash\{m'\}}\sum_{e=1}^{u}\alpha_{m,e}+\frac{\bar{d}-1}{\bar{s}-1}\sum_{e=1}^{u}\alpha_{m',e}\geqslant\frac{\bar{d}-1}{\bar{s}-1}\frac{u}{u-v}l. \tag{B.4}$$

Now let us average on the choice of $m'\in\mathcal{R}$:

$$\frac{u(\bar{d}-1)}{u-v}\sum_{m\in\mathcal{R}}\sum_{e=1}^{u}\alpha_{m,e}+\frac{\bar{d}-1}{\bar{s}-1}\sum_{m'\in\mathcal{R}}\sum_{e=1}^{u}\alpha_{m',e}\geqslant\frac{\bar{d}-1}{\bar{s}-1}\frac{u}{u-v}\bar{d}l$$

or

$$\alpha := \sum_{m\in\mathcal{R}}\sum_{e=1}^{u}\alpha_{m,e} \geqslant \frac{\bar{d}ul}{u(\bar{s}-1)+u-v} = \frac{\bar{d}ul}{s}.$$

Equality holds if and only if it holds in (B.2). This implies the uniform access condition, which is proved in exactly the same way as the uniform download condition in Prop. 24. □

B.3 Proof of Theorem 27

Proof of Part (a): The proof will be given for the repair of a node in a systematic rack. In the end we will argue that the claimed bound also applies to the repair of nodes in parity racks.

Without loss of generality, assume $\{\bar{k}+1,\ldots,\bar{k}+\bar{s}\}$ to be the $\bar{s}$ parity racks that are involved in the repair of the failed node. Let $\bar{k}+i, i=1,\ldots,\bar{s}$ be a helper rack. Since the repair scheme is linear, the information that this rack provides is obtained through a linear transformation of its contents. Denote the matrix of this transformation by $S_{\bar{k}+i,m_1}$ and call it the *repair matrix* for repairing a failed node in rack m_1 from rack $\bar{k}+i$ (and call its row space the *repair subspace* of the node. Note that it is an $\frac{l}{\bar{s}}\times ul$ matrix over F; moreover, for optimal repair, the rank of $S_{\bar{k}+i,m_1}$ necessarily is $l/\bar{s}$ for all $i\in[\bar{s}]$. The information that parity rack $\bar{k}+i$ transmits to repair the failed node in rack m_1 is given by

$$S_{\bar{k}+i,m_1}\boldsymbol{c}_{\bar{k}+i} = S_{\bar{k}+i,m_1}\Big(c_{k+(i-1)u+1},\ldots,c_{k+iu}\Big)^T$$

$$= S_{\bar{k}+i,m_1}\Big(\sum_{j=1}^{k} A_{(i-1)u+1,j}c_j, \ldots, \sum_{j=1}^{k} A_{iu,j}c_j\Big)^T, \quad i = 1, \ldots, \bar{s}. \tag{B.5}$$

For given i, j let us define the block matrix $\mathcal{A}_{i,j} = (A_{(i-1)u+1,j}, \ldots, A_{iu,j})^T$ (a part of column j in the encoding matrix that corresponds to rack m). Suppose that the index of the failed node in rack m_1 is j_1, and note that $(m_1-1)u+1 \leqslant j_1 \leqslant m_1 u$. Then from (B.5) we obtain

$$S_{\bar{k}+i,m_1}\boldsymbol{c}_{\bar{k}+i} = S_{\bar{k}+i,m_1}\mathcal{A}_{i,j_1}c_{j_1} + S_{\bar{k}+i,m_1}\sum_{\substack{j=1 \\ j\neq j_1}}^{k} \mathcal{A}_{i,j_1}c_j, \quad i = 1, \ldots, \bar{s}. \tag{B.6}$$

From (B.6) we observe that the information that parity rack $\bar{k}+i$ provides for the repair of node c_{j_1} contains interference from the other systematic nodes $c_j, j \neq j_1$. Moreover, as rack m_1 collects all the information sent from the helper racks $\bar{k}+i, 1 \leqslant i \leqslant \bar{s}$, in order to repair node c_j, it is necessary that

$$\operatorname{rank}\begin{bmatrix} S_{\bar{k}+1,m_1}\mathcal{A}_{1,j_1} \\ \vdots \\ S_{\bar{k}+\bar{s},m_1}\mathcal{A}_{\bar{s},j_1} \end{bmatrix} = l. \tag{B.7}$$

This relation holds true because Equations (B.6) evaluate a linear combination of the contents of the nodes in the host rack. To retrieve the l symbols of the failed node from this linear combination, condition (B.7) is necessary.

Let us further define the $ul \times ul$ matrix $\mathcal{D}_{i,m} = (\mathcal{A}_{i,(m-1)u+1}, \ldots, \mathcal{A}_{i,mu})$ by assembling together u columns of the form $\mathcal{A}_{.,.}$, i.e., $\mathcal{D}_{i,m} = (A_{\alpha,\beta}), (i-1)u+1 \leqslant$

$\alpha \leqslant iu, (m-1)u+1 \leqslant \beta \leqslant mu$. These matrices are defined for notational convenience and enable us to argue about entire racks rather than their elements. Since the code $\mathcal{C}$ is MDS, the matrix $\mathcal{D}_{i,m}$ is invertible for all $1 \leqslant i \leqslant \bar{r}$ and $1 \leqslant m \leqslant \bar{k}$. Rewriting (B.6) with this notation, we obtain

$$S_{\bar{k}+i,m_1}\boldsymbol{c}_{\bar{k}+i} = S_{\bar{k}+i,m_1}\mathcal{D}_{i,m_1}\boldsymbol{c}_{m_1} + S_{\bar{k}+i,m_1}\sum_{\substack{m=1\\ m\neq m_1}}^{\bar{k}}\mathcal{D}_{i,m}\boldsymbol{c}_m. \tag{B.8}$$

Since $(m_1-1)u+1 \leqslant j_1 \leqslant m_1 u$, from (B.7) we have

$$\operatorname{rank}\begin{bmatrix} S_{\bar{k}+1,m_1}\mathcal{D}_{1,m_1} \\ \vdots \\ S_{\bar{k}+\bar{s},m_1}\mathcal{D}_{\bar{s},m_1} \end{bmatrix} = l. \tag{B.9}$$

So far we have only considered the information provided by the parity racks. It remains to characterize the information transmitted by the systematic racks. From (B.8), in order to cancel out the interference from systematic rack $m \neq m_1$, rack m_1 needs to download from systematic rack m the vector

$$(S_{\bar{k}+1,m_1}\mathcal{D}_{1,m}\boldsymbol{c}_m, \ldots, S_{\bar{k}+\bar{s},m_1}\mathcal{D}_{\bar{s},m}\boldsymbol{c}_m)^T.$$

By Proposition 24, for optimal repair we necessarily have

$$\operatorname{rank}\begin{bmatrix} S_{\bar{k}+1,m_1}\mathcal{D}_{1,m} \\ \vdots \\ S_{\bar{k}+\bar{s},m_1}\mathcal{D}_{\bar{s},m} \end{bmatrix} = \frac{l}{\bar{s}}. \tag{B.10}$$

The rank conditions (B.9) and (B.10) give rise to the following subspace conditions. For any $m_1 \in [\bar{k}]$,

$$\bigoplus_{i=1}^{\bar{s}} \langle S_{\bar{k}+i,m_1}\mathcal{D}_{i,m_1}\rangle = F^l, \tag{B.11}$$

$$\langle S_{\bar{k}+1,m_1}\mathcal{D}_{1,m}\rangle = \ldots = \langle S_{\bar{k}+\bar{s},m_1}\mathcal{D}_{\bar{s},m}\rangle, \quad m \neq m_1. \tag{B.12}$$

The proof of the lower bounds in the theorem relies on these necessary conditions. Let us first bound the dimension of the intersection of the row spaces of the repair matrices.

Lemma 47. *Let $\mathcal{J} \subset [\bar{k}]$ be a subset of systematic nodes such that $|\mathcal{J}| \leqslant k-1$ and $m_1 \in \mathcal{J}$. Then for any $i, i' \geqslant 1$*

$$\dim \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle = \dim \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i',m}\rangle. \tag{B.13}$$

Proof. Let $a \in [\bar{k}]\backslash\mathcal{J}$. Since $\mathcal{D}_{i,a}$ is nonsingular, for each $i \in [\bar{s}]$ we have

$$\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\mathcal{D}_{i,a}\rangle = \Big(\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle\Big)\mathcal{D}_{i,a}.$$

On the previous line we take the intersection of the subspaces as indicated, then write a basis of the resulting subspace into rows of a matrix, which has ul columns. Since this is also the number of rows of $\mathcal{D}_{i,a}$, this operation is well defined.

Since $a \notin \mathcal{J}$, for any $i, i' \in [\bar{s}]$, from (B.12) we have

$$\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\mathcal{D}_{i,a}\rangle = \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i',m}\mathcal{D}_{i',a}\rangle.$$

Therefore, for any $i, i' \in [\bar{s}]$

$$\Big(\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle\Big)\mathcal{D}_{i,a} = \Big(\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i',m}\rangle\Big)\mathcal{D}_{i',a}. \tag{B.14}$$

Since $\mathcal{D}_{i,a}$ and $\mathcal{D}_{i',a}$ are invertible, (B.13) follows. □

Now consider the subspace $\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\mathcal{D}_{i,m_1}\rangle$, where $\mathcal{J}$ is as in Lemma 47. For any $i' \in [\bar{s}]$, we have

$$\begin{aligned}\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\mathcal{D}_{i,m_1}\rangle &= \langle S_{\bar{k}+i,m_1}\mathcal{D}_{i,m_1}\rangle \bigcap \Big(\bigcap_{m\in\mathcal{J}\setminus\{m_1\}} \langle S_{\bar{k}+i,m}\mathcal{D}_{i,m_1}\rangle\Big) \\ &= \langle S_{\bar{k}+i,m_1}\mathcal{D}_{i,m_1}\rangle \bigcap \Big(\bigcap_{m\in\mathcal{J}\setminus\{m_1\}} \langle S_{\bar{k}+i',m}\mathcal{D}_{i',m_1}\rangle\Big) \\ &\subseteq \bigcap_{m\in\mathcal{J}\setminus\{m_1\}} \langle S_{\bar{k}+i',m}\mathcal{D}_{i',m_1}\rangle. \end{aligned} \tag{B.15}$$

Summing on $i \in [\bar{s}]$ on both sides of (B.15), we obtain

$$\bigoplus_{i=1}^{\bar{s}} \Big(\bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\mathcal{D}_{i,m_1}\rangle\Big) \subseteq \bigcap_{m\in\mathcal{J}\setminus\{m_1\}} \langle S_{\bar{k}+i',m}\mathcal{D}_{i',m_1}\rangle. \tag{B.16}$$

Note that this is a direct sum because the subspaces $\bigcap_{m\in\mathcal{J}}\langle S_{\bar{k}+i,m}\mathcal{D}_{i,m_1}\rangle$ form a subset of the set of subspaces on the left-hand side of (B.11) and therefore (for different i) are disjoint.

Since $\mathcal{D}_{i,m_1}$ and $\mathcal{D}_{i',m_1}$ are invertible, we conclude that

$$\sum_{i=1}^{\bar{s}} \dim \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle \leqslant \dim \bigcap_{m\in\mathcal{J}\backslash\{m_1\}} \langle S_{\bar{k}+i',m}\rangle.$$

Note that from (B.13), we have

$$\sum_{i=1}^{\bar{s}} \dim \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle = \bar{s} \dim \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle.$$

Therefore,

$$\begin{aligned}\dim \bigcap_{m\in\mathcal{J}} \langle S_{\bar{k}+i,m}\rangle &\leqslant \frac{1}{\bar{s}} \dim \bigcap_{m\in\mathcal{J}\backslash\{m_1\}} \langle S_{\bar{k}+i',m}\rangle \\ &\leqslant \frac{1}{\bar{s}^{|\mathcal{J}|-1}} \dim\langle S_{\bar{k}+i',m}\rangle \\ &\leqslant \frac{l}{\bar{s}^{|\mathcal{J}|}}. \end{aligned} \tag{B.17}$$

If $l \geqslant \bar{s}^{\bar{k}-1}$, then (3.6) is proved, so let us assume that $l < \bar{s}^{\bar{k}-1}$.

Lemma 48. *Let $\mathcal{T} \subset [\bar{n}]\backslash\{\bar{k}+i\}$ be a subset such that*

- $1 \leqslant |\mathcal{T}| \leqslant \bar{n}-1$,
- $m_1 \in \mathcal{T}$,
- $\mathcal{T}$ *contains* $\min(\bar{k}-1, |\mathcal{T}|)$ *systematic racks.*

Then

$$\dim \bigcap_{m\in\mathcal{T}} \langle S_{\bar{k}+i,m}\rangle \leqslant \frac{l}{\bar{s}^{|\mathcal{T}|}}. \tag{B.18}$$

Remark: Some of the repair matrices in (B.18) refer to the repair scheme of a parity node (a node in a parity rack) using information from another parity rack. These matrices exist and are well defined because by assumption, the code $\mathcal{C}$ supports optimal repair of any node from any set of $\bar{d}$ helper racks.

Proof. By the assumption before the lemma, Eq. (B.17) holds for any $\mathcal{J}$ of size $\leqslant \bar{k}-1$, which proves the claim for the case $|\mathcal{T}| \leqslant \bar{k}-1$. At the same time, if $|\mathcal{T}| > \bar{k}-1$, take $|\mathcal{J}| = \bar{k}-1$ in (B.17) and note that $\mathcal{J} \subset \mathcal{T}$. In this case (B.17) implies (B.18) and the proof is complete. □

From (B.18) we observe that the subspaces $\langle S_{\bar{k}+i,m}\rangle, m \in \mathcal{T}$ have a vector in common if and only if $|\mathcal{T}| \leqslant \log_{\bar{s}} l$. Now consider a $ul \times (\bar{n}-1)$ matrix V whose rows correspond to the ul vectors in the standard basis of F^{ul} and columns to the repair matrices $S_{\bar{k}+i,m}, m \in [\bar{n}]\backslash\{\bar{k}+i\}$. Put $V_{im} = 1$ if the ith vector is one of the rows of the mth repair matrix and 0 otherwise. The code has the optimal access property if and only if the rows of the repair matrices are formed of standard basis vectors. Every column of V contains $l/\bar{s}$ ones, and if $|\mathcal{T}| \leqslant \log_{\bar{s}} l$, then every row contains at most $\log_{\bar{s}} l$ ones; thus

$$\frac{l}{\bar{s}}(\bar{n}-1) \leqslant ul \log_{\bar{s}} l.$$

It follows that

$$l \geqslant \bar{s}^{\frac{\bar{n}-1}{s}}, \tag{B.19}$$

where we used $s = \bar{s}u$. This concludes the proof of Part (a).

Proof of Part (b): We closely follow the arguments in Part (a) with the only difference that the set $\mathcal{T}$ can now be of size $\bar{n}$, which is possible because the repair matrices are independent of the choice of the helper racks.

Let us outline the argument. Let $|\mathcal{J}| = \bar{k} - 1$ and $l < \bar{s}^{\bar{k}-1}$. In this case (B.17) implies that

$$\dim \bigcap_{m \in \mathcal{J}} \langle S_m \rangle = 0$$

(even in the case when the repair scheme is chosen based on the location of the helpers, and all the more so in the current case). It follows that, for any $\mathcal{T} \subseteq [\bar{n}]$ such that $\mathcal{T} \supseteq \mathcal{J}$, we have

$$\dim \bigcap_{m \in \mathcal{T}} S_m = 0.$$

Therefore, for $l < \bar{s}^{\bar{k}-1}$, we have

$$\dim \bigcap_{m \in \mathcal{T}} S_m \leqslant \frac{l}{\bar{s}^{|\mathcal{T}|}}, \tag{B.20}$$

for $1 \leqslant |\mathcal{T}| \leqslant \bar{n}$. For the left-hand side of the above inequality to be greater than 1,

we necessarily have $|\mathcal{T}| \leqslant \log_{\bar{s}} l$.

Repeating the argument that led to (B.19), we obtain

$$l \geqslant \bar{s}^{\bar{n}/s}. \tag{B.21}$$

Thus, we have proved cases (a) and (b) of the theorem for repairing a failed node in a systematic rack. The same bounds hold for repairing a failed node in any of the parity racks. Indeed, note that a parity rack $\boldsymbol{c}_{\bar{k}+i}, i = 1, \ldots, \bar{r}$ is computed from the systematic racks as follows:

$$\boldsymbol{c}_{\bar{k}+i} = \sum_{j=1}^{\bar{k}} \mathcal{D}_{\bar{k}+i,j} \boldsymbol{c}_j \tag{B.22}$$

If a node in rack $\bar{k}+i$ has failed, we first choose $\bar{d}$ helper racks and isolate any $(\bar{k}-1)$-subset of the chosen $\bar{d}$-set. Then we write equations of the form (B.22) where on the right we use these $\bar{k}-1$ racks together with the host rack to express the code symbols in the remaining $\bar{r}$ racks. These equations are obtained from (B.22) using obvious matrix transformations (no complications arise because the code $\mathcal{C}$ is MDS). After that, we can repeat the proofs given above, which establishes our claim.

Appendix C: Omitted Proofs in Chapter 4

C.1 Proof of Proposition 37

Let us first prove a technical claim.

Claim 49. *For* $1 \leqslant i \leqslant h+1$, *we have* $\delta_1 = (\delta_i - b_0^{(i-1)}) \bmod n_1$.

Proof. We prove the claim by induction. Clearly, for $i = 1$ we have $\delta_1 = (\delta_1 - b_0^{(0)}) \bmod n_1$. Next suppose that for a given $i, 1 \leqslant i < h+1$ we have $\delta_1 = (\delta_i - b_0^{(i-1)}) \bmod n_1$. Reducing (4.10) modulo n_1, we obtain the equality $(\delta_i - b_0^{(i-1)}) = (\delta_{i+1} - b_0^{(i)}) \bmod n_1$. Thus also $\delta_1 = (\delta_{i+1} - b_0^{(i)}) \bmod n_1$, and this completes the proof. □

Now we are ready to prove Proposition 37. Clearly, for $i = 1$ we have $|\mathcal{Z}_1| = \delta_1 - 1 = n_1 - r_1$. Suppose for $1 \leqslant i' \leqslant i$ we have $|\mathcal{Z}_{i'}| = n_{i'} - r_{i'}$, where $1 \leqslant i < h+1$. Let us establish the induction step.

By the definition of $\mathcal{Z}_{i+1}$ and (4.10), we have $|\mathcal{Z}_{i+1}| \geqslant (\nu_i - a_i)n_i$. Consider the set

$$\mathcal{A}_{i+1} = (\nu_i - a_i)n_i + \{\delta_i - b_0^{(i-1)} - \delta_1 + 1, \delta_i - b_0^{(i-1)} - \delta_1 + 2, \ldots, n_i\}$$

formed by adding $(\nu_i - a_i)n_i$ to every element of the subset above, and let $\mathcal{B}_{i+1} = \mathcal{A}_{i+1} \cap \mathcal{D}_{i+1}$. Since $\bigcup_{l=0}^{\nu_i - a_i - 1}(\mathcal{Z}_i + ln_i) \subseteq \mathcal{D}_{i+1}$, we have

$$|\mathcal{Z}_{i+1}| = (\nu_i - a_i)n_i + |\mathcal{B}_{i+1} \backslash (\mathcal{Z}_i + (\nu_i - a_i)n_i)| + \Big| \bigcup_{l=\nu_i - a_i}^{\nu_i - 1} (\mathcal{Z}_i + ln_i) \Big|. \tag{C.1}$$

Next, we would like to determine the cardinality of the set $\mathcal{B}_{i+1} \backslash (\mathcal{Z}_i + (\nu_i - a_i)n_i)$. By Claim 49, we have $\delta_1 = (\delta_i - b_0^{(i-1)}) \bmod n_1$. Therefore, $|\mathcal{A}_{i+1}|$ is divisible by n_1. Note that $\mathcal{B}_{i+1} \subseteq \mathcal{A}_{i+1}$. Moreover, among the first $u_{i-1}^{(i)} n_{i-1}$ elements of $\mathcal{A}_{i+1}$ there are $u_{i-1}^{(i)} r_{i-1} - b_0^{(i-1)}$ elements that are in $\mathcal{B}_{i+1}$ but not in $\mathcal{Z}_i + (\nu_i - a_i)n_i$. For the next $u_{i-2}^{(i)} n_{i-2}$ elements in $\mathcal{A}_{i+1}$ there are $u_{i-2}^{(i)} r_{i-2}$ elements that are in $\mathcal{B}_{i+1}$ but not in $\mathcal{Z}_i + (\nu_i - a_i)n_i$, and so forth. Hence, we have

$$\begin{aligned} |\mathcal{B}_{i+1} \backslash (\mathcal{Z}_i + (\nu_i - a_i)n_i)| &= \sum_{j=1}^{i-1} u_j^{(i)} r_j + u_0^{(i)} r_1 + b_0^{(i)} - b_0^{(i-1)} \\ &= b_i. \end{aligned}$$

It follows that

$$\begin{aligned} |\mathcal{Z}_{i+1}| &= (\nu_i - a_i)n_i + b_i + \Big| \bigcup_{l=\nu_i - a_i}^{\nu_i - 1} (\mathcal{Z}_i + ln_i) \Big| \\ &= (\nu_i - a_i)n_i + b_i + a_i |\mathcal{Z}_i| \\ &= n_{i+1} - a_i n_i + b_i + a_i(n_i - r_i) \\ &= n_{i+1} - (a_i r_i - b_i) \\ &= n_{i+1} - r_{i+1}, \end{aligned} \tag{C.2}$$

where (C.2) uses $|\mathfrak{Z}_i| = n_i - r_i$, which is the induction hypothesis. This completes the proof.

C.2 Proof of Lemma 38

We again argue by induction on i. For $i = 1$, by (4.10) we have

$$\begin{aligned}
\delta_2 &= (\nu_1 - a_1)n_1 + \delta_1 + u_0^{(1)} n_1 + b_0^{(1)} - b_0^{(0)} \\
&= n_2 - a_1 n_1 + \delta_1 + u_0^{(1)}(n_1 - r_1) + b_1 \qquad \text{(C.3)} \\
&= n_2 - a_1(r_1 + \delta_1 - \delta_0) + \delta_1 + u_0^{(1)}(n_1 - r_1) + b_1 \\
&= n_2 - r_2 + \delta_1 - a_1(\delta_1 - \delta_0) + u_0^{(1)}(\delta_1 - \delta_0) \qquad \text{(C.4)} \\
&= n_2 - r_2 + \delta_1 - (a_1 - u_0^{(1)})(\delta_1 - \delta_0) \\
&= n_2 - r_2 + \delta_1 - \left\lceil \frac{r_2}{r_1} \right\rceil (\delta_1 - \delta_0),
\end{aligned}$$

where (C.3) follows from $n_2 = n_1\nu_1$ and (4.9), in (C.4) we used $a_1 r_1 = r_2 + b_1$, and the last equality follows from $a_1 = \lceil r_2/r_1 \rceil$ and $u_0^{(1)} = \lfloor (b_1^{(1)} + b_0^{(0)})/r_1 \rfloor = 0$.

For the induction step, let us fix $i, 1 \leqslant i < h$ and suppose that

$$\delta_{i+1} = n_{i+1} - r_{i+1} + \delta_i - \sum_{l=1}^{i} \left\lceil \frac{r_{i+1}}{r_l} \right\rceil (\delta_l - \delta_{l-1}), \qquad \text{(C.5)}$$

provided that conditions (4.11) are satisfied. Observe that

$$\delta_{i+2} = (\nu_{i+1} - a_{i+1})n_{i+1} + \delta_{i+1} + \sum_{j=1}^{i} u_j^{(i+1)} n_j + u_0^{(i+1)} n_1 + b_0^{(i+1)} - b_0^{(i)}$$

$$= n_{i+2} - a_{i+1}n_{i+1} + \delta_{i+1} + \sum_{j=1}^{i} u_j^{(i+1)}(n_j - r_j) + u_0^{(i+1)}(n_1 - r_1) + b_{i+1} \qquad \text{(C.6)}$$

Substituting n_{i+1} from (C.5), we obtain

$$a_{i+1}n_{i+1} = a_{i+1}\Big(r_{i+1} + \delta_{i+1} - \delta_i + \sum_{l=1}^{i} \Big\lceil \frac{r_{i+1}}{r_l} \Big\rceil (\delta_l - \delta_{l-1})\Big). \qquad \text{(C.7)}$$

In addition, also by the induction hypothesis, we have

$$\begin{aligned}
&u_0^{(i+1)}(n_1 - r_1) = u_0^{(i+1)}(\delta_1 - \delta_0),\\
&u_j^{(i+1)}(n_j - r_j) = u_j^{(i+1)}(\delta_j - \delta_{j-1}) + \sum_{l=1}^{j-1} u_j^{(i+1)} \Big\lceil \frac{r_j}{r_l} \Big\rceil (\delta_l - \delta_{l-1}), \quad 1 \leqslant j \leqslant i.
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\sum_{j=1}^{i} u_j^{(i+1)}(n_j - r_j) + u_0^{(i+1)}(n_1 - r_1)\\
&\qquad = \sum_{l=1}^{i} \Big(u_l^{(i+1)} + \sum_{j=l+1}^{i} u_j^{(i+1)} \Big\lceil \frac{r_j}{r_l} \Big\rceil\Big)(\delta_l - \delta_{l-1}) + u_0^{(i+1)}(\delta_1 - \delta_0). \qquad \text{(C.8)}
\end{aligned}$$

Substituting (C.7) and (C.8) into (C.6), we obtain

$$\begin{aligned}
\delta_{i+2} = n_{i+2} &- r_{i+2} + \delta_{i+1} - a_{i+1}(\delta_{i+1} - \delta_i)\\
&+ \sum_{l=2}^{i} \Big(u_l^{(i+1)} + \sum_{j=l+1}^{i} u_j^{(i+1)} \Big\lceil \frac{r_j}{r_l} \Big\rceil - a_{i+1} \Big\lceil \frac{r_{i+1}}{r_l} \Big\rceil\Big)(\delta_l - \delta_{l-1})\\
&+ \Big(u_0^{(i+1)} + u_1^{(i+1)} + \sum_{j=2}^{i} u_j^{(i+1)} \Big\lceil \frac{r_j}{r_1} \Big\rceil - a_{i+1} \Big\lceil \frac{r_{i+1}}{r_1} \Big\rceil\Big)(\delta_1 - \delta_0).
\end{aligned}$$

Thus, if the corresponding conditions of (4.11) are satisfied, then we have

$$\delta_{i+2} = n_{i+2} - r_{i+2} + \delta_{i+1} - \sum_{l=1}^{i+1} \left\lceil \frac{r_{i+2}}{r_l} \right\rceil (\delta_{i+1} - \delta_i).$$

This completes the induction step.

C.3 Proof of Proposition 42

Part (a): The generator matrix of the truncated code $\mathcal{C}_{[0,j]}$ is given in (4.21), where $\text{rank}(G_0) = k$.

Since G_0 has full rank, by Definition 10, the j-th column distance of $\mathcal{C}$ is equal to

$$d_j^c = \min\{\text{wt}(u_{[0,j]}G_j^c) \mid u_0 \neq 0\},$$

where $u_{[0,j]} = (u_0, \ldots, u_j) \in \mathbb{F}_q^{k(j+1)}$ is an input sequence truncated at the j-th time instant. Obviously,

$$d_j^c \leqslant \min\{\text{wt}(u_{[0,j]}G_j^c) \mid u_0 \neq 0, u_1 = u_2 = \ldots = u_j = 0\}. \tag{C.9}$$

Consider the $k \times n(j+1)$ submatrix $[G_0, G_1, \ldots, G_j]$ that forms the first k rows of G_j^c. By elementary row operations it is possible to make some, say first, k columns of G_1 be all-zero, and the same is true for some k columns of the matrices $G_i, i = 2, \ldots, j$. (Note that to accomplish this, we use all the rows of G_j^c and not just the rows of

the submatrix $[G_0, G_1, \ldots, G_j]$.) As a result, the vector $b(u_0) := u_0[G_0, G_1, \ldots, G_j]$ will contain at least kj zero coordinates for any choice of $u_0 \in \mathbb{F}_q^k$, and so the effective length of the set of vectors $\{b(u_0)\}$ is $n(j+1) - kj$. The set of vectors $\{b(u_0) \mid u_0 \in \mathbb{F}_q^k\}$ is a subset of the code $\mathcal{C}_{[0,j]}$ and it forms a linear code $\mathcal{K}$ of effective length $n(j+1) - kj$ with (r, δ) locality if the original convolutional code $\mathcal{C}$ has (column or row) (r, δ) locality. The distance of the code $\mathcal{K}$ gives an upper bound on d_j^c, and is itself bounded above as in (1.6). Substituting the parameters of the code $\mathcal{K}$, we obtain the bound (4.24) for the distance d_j^c. Part (a) is proved.

Remark: Without the locality assumption the above argument proves the Singleton bound (4.26) for the column distance of the code $\mathcal{C}$.

Part (b): Recall the following observation (e.g., [24]):

Proposition 50. *Let $j \geqslant 0$. Let H_j^c be the parity check matrix for the truncated convolutional code $\mathcal{C}_{[0,j]}$. Then the following properties are equivalent:*

1. *$d_j^c = d$;*
2. *none of the first n columns of H_j^c is contained in the span of any other $d-2$ columns and one of the first n columns of H_j^c is in the span of some other $d-1$ columns of H_j^c.*

Now suppose that d_j^c attains (4.24) with equality while $d_{j-1}^c < (n-k)j + \delta - \left\lceil \frac{k}{r} \right\rceil (\delta - 1)$. By Proposition 50, there exists a column among the first n columns of H_{j-1}^c such that it is in the span of some other $d_{j-1}^c - 1$ columns of H_{j-1}^c. Note that

H^c_{j-1} is a submatrix of H^c_j. Specifically, we have

$$H^c_j = \begin{pmatrix} & & & 0 \\ & H^c_{j-1} & & \vdots \\ & & & 0 \\ H_j & H_{j-1} & \cdots & H_1 & H_0 \end{pmatrix},$$

where $H_i, 1 \leqslant i \leqslant j$ are $(n-k) \times n$ matrices and $\text{rank}(H_0) = n-k$. Because of this, there exists a column among the first n columns of H^c_j such that it is in the span of some other $d^c_{j-1} - 1 + n - k < d^c_j - 1$ columns of H^c_j, which by Proposition 50 contradicts our assumption about d^c_j. Hence, it follows that the optimality of the j-th column distance implies the optimality of the i-th column distance for all $i \leqslant j$ for convolutional codes with (column or row) locality.

Bibliography

[1] S. Akhlagi, A. Kiani, and M. R. Ghabavati. Cost-bandwidth tradeoff in distributed storage systems. *Computer Commuunications*, 33(17):2105–2115, 2010.

[2] S. B. Balaji, M. N. Krishnan, M. Vajha, V. Ramkumar, B. Sasidharan, and P. V. Kumar. Erasure coding for distributed storage: An overview. *Science China Information Sciences*, 61(100301):1–45, 2018.

[3] S. B. Balaji and P. V. Kumar. A tight lower bound on the sub-packetization level of optimal-access MSR and MDS codes. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 2381–2385, 2018. Expanded version available online as arXiv:1710.05876.

[4] S. Ballentine, A. Barg, and S. Vlăduţ. Codes with hierarchical locality from covering maps of curves. *IEEE Trans. Inf. Theory*, 65(10):6056–6071, 2019.

[5] A. Barg, I. Tamo, and S. Vlăduţ. Locally recoverable codes on algebraic curves. *IEEE Trans. Inform. Theory*, 63(8):4928–4939, 2017.

[6] A. Beemer, R. Coatney, V. Guruswami, H. H. Lopez, and F. Pinero. Explicit optimal-length locally repairable codes of distance 5. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 800–804, 2018.

[7] E. L. Blokh and V. V. Zyablov. Coding of generalized cascade codes. *Probl. Inform. Trans.*, 10(3):218–222, 1974.

[8] 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 (ISIT)*, pages 1225–1229, 2011.

[9] V. R. Cadambe, S. A. Jafar, H. Maleki, K. Ramchandran, and C. Suh. Asymptotic interference alignment for optimal repair of MDS codes in distributed storage. *IEEE Trans. Inf. Theory*, 59(5):2974–2987, 2013.

[10] H. Cai, Y. Miao, M. Schwartz, and X. Tang. On optimal locally repairable codes with super-linear length. *IEEE Trans. Inf. Theory*, 66(8):4853–4868, 2020.

[11] B. Chen, S.-T. Xia, J. Hao, and F.-W. Fu. Constructions of optimal (r, δ) locally repairable codes. *IEEE Trans. Inf. Theory*, 64(4):2499–2511, 2018.

[12] Z. Chen and A. Barg. Cyclic and convolutional codes with locality. *IEEE Trans. Inf. Thoery*, 2020. DOI: 10.1109/TIT.2020.3031207.

[13] Z. Chen and A. Barg. Explicit constructions of MSR codes for clustered distributed storage: The rack-aware storage model. *IEEE Trans. Inf. Theory*, 66(2):886–899, 2020.

[14] Z. Chen, M. Ye, and A. Barg. Enabling optimal access and error correction for the repair of Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 2020. DOI: 10.1109/TIT.2020.3017666.

[15] A. Datta. Locally repairable rapidRAID systematic codes–one simple convoluted way to get it all. In *2014 IEEE Information Theory Workshop (ITW 2014)*, pages 60–64, 2014.

[16] H. Dau, I. Duursma, and H. Chu. On the I/O costs of some repair schemes for full-length Reed-Solomon codes. In *2018 IEEE International Symposium Information Theory*, pages 1700–1704, 2018.

[17] H. Dau, I. Duursma, H. M. Kiah, and O. Milenkovic. Repairing Reed-Solomon codes with multiple erasures. *IEEE Trans. Inf. Theory*, 54(10):6567–6582, 2018.

[18] H. Dau and O. Milenkovic. Optimal repair schemes for some families of Reed-Solomon codes. In *2017 IEEE International Symposium on Information Theory*, pages 346–350, 2017.

[19] H. Dau and E. Viterbo. Repair schemes with optimal I/O costs for full-length Reed-Solomon codes with two parities. In *2018 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2018.

[20] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran. Network coding for distributed storage systems. *IEEE Trans. Inf. Theory*, 56(9):4539–4551, 2010.

[21] M. Esmaeili, T. A. Gulliver, N. P. Secord, and S. A. Mahmoud. A link between quasi-cyclic codes and convolutional codes. *IEEE Trans. Inf. Theory*, 44(1):431–435, 1998.

[22] R. Freij-Hollanti, T. Westerbäck, and C. Hollanti. Locally repairable codes with availability and hierarchy: Matroid theory via examples. In *24th International Zürich Seminar on Communications (IZS), Zurich, Switzerland, March 2-4, 2016*, pages 45–49. ETH-Zürich, 2016. https://doi.org/10.3929/ethz-a-010645448.

[23] B. Gastón, J. Pujol, and M. Villanueva. A realistic distributed storage system that minimizes data storage and repair bandwidth. In *Proc. Data Compression Conf*, 2013. Preprint: arXiv:1301.1549.

[24] H. Gluesing-Luerssen, J. Rosenthal, and R. Smarandache. Strongly-MDS convolutional codes. *IEEE Trans. Inf. Theory*, 52(2):584–598, 2006.

[25] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin. On the locality of codeword symbols. *IEEE Trans. Inf. Theory*, 58(11):6925–6934, 2012.

[26] S. Goparaju, A. Fazeli, and A. Vardy. Minimum storage regenerating codes for all parameters. *IEEE Trans. Inf. Theory*, 63(10):6318–6328, 2017.

[27] M. Grezet and C. Hollanti. The complete hierarchical locality of the punctured simplex code. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 2833–2837, 2019.

[28] V. Guruswami and M. Wootters. Repairing Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 63(9):5684–5698, 2017.

[29] V. Guruswami, C. Xing, and C. Yuan. How long can optimal locally repairable codes be? *IEEE Trans. Inf. Theory*, 6(6):3662–3670, 2019.

[30] L. Holzbaur, R. Freij-Hollanti, and A. Wachter-Zeh. Cyclic codes with locality and availability, 2018. arXiv preprint arXiv:1812.06897.

[31] H. Hou, P. P. C. Lee, K. W. Shum, and Y. Hu. Rack-aware regenerating codes for data centers. *IEEE Trans. Inf. Theory*, 65(8):4730–4745, 2019.

[32] Y. Hu, P. P. C. Lee, and X. Zhang. Double regenerating codes for hierarchical data centers. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 245–249. IEEE, 2016.

[33] Y. Hu, X. Li, M. Zhang, P. Lee, X. Zhang, P. Zhou, and D. Feng. Optimal repair layering for erasure-coded data centers: From theory to practice. *ACM Transactions on Storage (TOS)*, 13(4), 2017. Article #33.

[34] P. Huang, E. Yaakobi, and P. H. Siegel. Multi-erasure locally recoverable codes over small fields: A tensor product approach. *IEEE Trans. Inf. Theory*, 66(5):2609–2624, 2020.

[35] F. Ivanov, A. Kreshchuk, and V. Zyablov. On the local erasure correction capacity of convolutional codes. In *2018 International Symposium on Information Theory and Its Applications (ISITA), Singapore*, pages 296–300, 2018.

[36] L. Jin, G. Luo, and C. Xing. Optimal repairing schemes for Reed-Solomon cods with alphabet sizes linear in lengths under the rack-aware model. arXiv:1911.08016, Nov. 2019.

[37] R. Johannesson and K. S. Zigangirov. *Fundamentals of Convolutional Coding*. J. Wiley & Sons, Inc., Hoboken, NJ, 2nd edition, 2015.

[38] J. Justesen, E. Paaske, and M. Ballan. Quasi-cyclic unit memory convolutional codes. *IEEE Trans. Inf. Theory*, 36(3):540–547, 1990.

[39] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar. Codes with local regeneration and erasure correction. *IEEE Trans. Inform. Theory*, 60(8):4637–4660, 2014.

[40] A. M. Kermarrec, N. Le Scouarnec, and G. Straub. Repairing multiple failures with coordinated and adaptive regenerating codes. In *2011 Int. Sympos. Network Coding (NetCod)*, pages 1–6. IEEE, 2011.

[41] S. Kruglik, K. Nazirkhanova, and A. Frolov. New bounds and generalizations of locally recoverable codes with availability. *IEEE Trans. Inf. Theory*, 65(7):4156–4166, 2019.

[42] K. Lally and P. Fitzpatrick. Algebraic structure of quasicyclic codes. *Discrete Applied Mathematics*, 111:157–175, 2001.

[43] J. Li and B. Li. Cooperative repair with minimum-storage regenerating codes for distributed storage. In *Proc. IEEE INFOCOM*, pages 316–324. IEEE, 2014.

[44] J. Li, X. Tang, and C. Tian. A generic transformation to enable optimal repair in mds codes for distributed storage systems. *IEEE Trans. Inf. Theory*, 64(9):6257–6267, 2018.

[45] W. Li, H. Dau, Z. Wang, H. Jafarkhani, and E. Viterbo. On the I/O costs in repairing short-length Reed-Solomon codes. In *2019 IEEE International Symposium Information Theory*, pages 1087–1091, 2019.

[46] X. Li, L. Ma, and C. Xing. Construction of asymptotically good locally repairable codes via automorphism groups of function fields. *IEEE Trans. Inf. Theory*, 65(11):7087–7094, 2019.

[47] X. Li, L. Ma, and C. Xing. Optimal locally repairable codes via elliptic curves. *IEEE Trans. Inform. Theory*, 65(1):108–117, 2019.

[48] S. Ling and P. Solé. On the algebraic structure of quasi-cyclic codes, I. *IEEE Trans. Inf. Theory*, 47(7):2751–2760, 2001.

[49] S. Ling and P. Solé. On the algebraic structure of quasi-cyclic codes II: Chain rings. *Designs, Codes and Cryptography*, 30:113–130, 2003.

[50] Y. Luo, C. Xing, and C. Yuan. Optimal locally repairable codes of distance 3 and 4 via cyclic codes. *IEEE Trans. Inf. Theory*, 65(2):1048–1053, 2018.

[51] J. Mardia, B. Bartan, and M. Wootters. Repairing multiple failures for scalar MDS codes. *IEEE Trans. Inf. Theory*, 65(5):2661–2672, 2019.

[52] U. Martínez-Peñas and D. Napp. Locally repairable convolutional codes with sliding window repair. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 2838–2842, 2019. expanded version in arXiv preprint arXiv:1901.02073.

[53] S. Pawar, S. El Rouayheb, and K. Ramchandran. Securing dynamic distributed storage systems against eavesdropping and adversarial attacks. *IEEE Trans. Inf. Theory*, 57(10):6734–6753, 2011.

[54] J. Pernas, C. Yuen, B. Gastón, and J. Pujol. Non-homogeneous two-rack model for distributed storage systems. In *2013 IEEE International Symposium on Information Theory (ISIT)*, pages 1237–1241, 2013.

[55] N. Prakash, V. Abdrashitov, and M. Medard. The storage vs repair bandwidth trade-offs for clustered storage systems. *IEEE Trans. Inf. Theory*, 64(8):5783–5805, 2018.

[56] 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 Trans. Inf. Theory*, 57(8):5227–5239, 2011.

[57] K. V. Rashmi, N. B. Shah, K. Ramchandran, and P. V. Kumar. Regenerating codes for errors and erasures in distributed storage. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1202–1206, 2012.

[58] N. Raviv, N. Silberstein, and T. Etzion. Constructions of high-rate minimum storage regenerating codes over small fields. *IEEE Trans. Inf. Theory*, 63(4):2015–2038, 2017.

[59] A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath. Centralized repair of multiple node failures with applications to communication efficient secret sharing. *IEEE Trans. Inf. Theory*, 64(12):7529–7550, 2018.

[60] A. S. Rawat, D. S. Papailiopoulos, A. G. Dimakis, and S. Vishwanath. Locality and availability in distributed storage. *IEEE Trans. inf. theory*, 62(8):4481–4493, 2016.

[61] S. Sahraei and M. Gastpar. Increasing availability in distributed storage systems via clustering. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 1705–1709, 2018. Preprint: arXiv:1710.02653v2.

[62] K. Saints and C. Heegard. On hyperbolic cascaded Reed-Solomon codes. In *International Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*, pages 291–303. Springer, 1993.

[63] B. Sasidharan, G. K. Agarwal, and P. V. Kumar. Codes with hierarchical locality. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 1257–1261, 2015. Expanded version in arXiv preprint arXiv:1501.06683.

[64] 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. Preprint: arXiv:1607.07335.

[65] 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 Trans. Inf. Theory*, 58(3):1837–1852, 2012.

[66] 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.

[67] K. W. Shum and Y. Hu. Cooperative regenerating codes. *IEEE Trans. Inf. Theory*, 59(11):7229–7258, 2013.

[68] J.-y. Sohn, B. Choi, and J. Moon. A class of MSR codes for clustered distributed storage. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 2366–2370, 2018.

[69] J.-y. Sohn, B. Choi, S. W. Yoon, and J. Moon. Capacity of clustered distributed storage. *IEEE Trans. Inf. Theory*, 65(1):81–107, 2019.

[70] G. Solomon and H. C. A. van Tilborg. A connection between block and convolutional codes. *SIAM J. Appl. Math.*, 37(2):358–369, 1979.

[71] I. Tamo and A. Barg. A family of optimal locally recoverable codes. *IEEE Trans. Inf. Theory*, 60(8):4661–4676, 2014.

[72] I. Tamo, A. Barg, and A. Frolov. Bounds on the parameters of locally recoverable codes. *IEEE Trans. inf. theory*, 62(6):3070–3083, 2016.

[73] I. Tamo, A. Barg, S. Goparaju, and R. Calderbank. Cyclic LRC codes, binary LRC codes, and upper bounds on the distance of cyclic codes. *International Journal of Information and Coding Theory*, 3(4):345–364, 2016.

[74] I. Tamo, Z. Wang, and J. Bruck. Zigzag codes: MDS array codes with optimal rebuilding. *IEEE Trans. Inf. Theory*, 59(3):1597–1616, 2013.

[75] I. Tamo, Z. Wang, and J. Bruck. Access versus bandwidth in codes for storage. *IEEE Trans. Inf. Theory*, 60(4):2028–2037, 2014.

[76] 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)*, pages 216–227, 2017.

[77] I. Tamo, M. Ye, and A. Barg. The repair problem for Reed-Solomon codes: Optimal repair of single and multiple erasures with almost optimal node size. *IEEE Trans. Inf. Theory*, 65(5):2673–2695, 2019.

[78] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello. LDPC block and convolutional codes based on circulant matrices. *IEEE Trans. Inf. Theory*, 50(12):2966–2984, 2004.

[79] M. A. Tebbi, T. H. Chan, and C. W. Sung. A code design framework for multi-rack distributed storage. In *Proc. IEEE Information Theory Workshop (ITW 2014)*, pages 55–59, 2014.

[80] V. Tomás, J. Rosenthal, and R. Smarandache. Decoding of convolutional codes over the erasure channel. *IEEE Trans. Inf. Theory*, 58(1):90–108, 2012.

[81] M. Vajha, V. Ramkumar, B. Puranik, G. Kini, E. Lobo, B. Sasidharan, P. V. Kumar, A. Barg, M. Ye, S. Hussain, S. Narayanamurthy, and S. Nandi. Clay codes: Moulding MDS codes to yield an MSR code. In *16th USENIX Conference on File and Storage Technologies (FAST 2018), Oakland, CA*, pages 139–154, Feb. 2018.

[82] A. Wang and Z. Zhang. Repair locality with multiple erasure tolerance. *IEEE Trans. Inf. Theory*, 60(11):6979–6987, 2014.

[83] S. Yang, A. Hareedy, R. Calderbank, and L. Dolecek. Hierarchical coding to enable scalability and flexibility in heterogeneous cloud storage. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2019. arXiv preprint arXiv:1905.02279.

[84] M. Ye and A. Barg. Explicit constructions of high-rate MDS array codes with optimal repair bandwidth. *IEEE Trans. Inf. Theory*, 63(4):2001–2014, 2017.

[85] M. Ye and A. Barg. Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization. *IEEE Trans. Inf. Theory*, 63(10):6307–6317, 2017.

[86] M. Ye and A. Barg. Cooperative repair: Constructions of optimal MDS codes for all admissible parameters. *IEEE Trans. Inf. Theory*, 65(3):1639–1656, 2019.

[87] B. Zhu, X. Li, H. Li, and K. W. Shum. Replicated convolutional codes: A design framework for repair-efficient distributed storage codes. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1018–1024, 2016.