ISIT'2013 1569720433

High-Rate Regenerating Codes through Layering

Birenjith Sasidharan, P. Vijay Kumar
Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore.
Email: biren, vijay@ece.iisc.ernet.in

***Abstract*—In this paper, we provide explicit constructions for a class of exact-repair regenerating codes that possess a layered structure. These regenerating codes correspond to interior points on the storage-repair-bandwidth tradeoff where the cut-set bound of network coding is known to be not achievable under exact repair. The codes presented in this paper compare very well in comparison to schemes that employ space-sharing between MSR and MBR points, and come closest of all-known explicit constructions to interior points of the tradeoff. The codes can be constructed for a wide range of parameters, are high-rate, can repair multiple nodes simultaneously and no computation at helper nodes is required to repair a failed node. We also construct optimal codes with locality in which the local codes are layered regenerating codes.**

I. Introduction

A. Regenerating Codes

In a distributed storage system, information pertaining to a single file, comprised of K symbols drawn from a finite field $\mathbb{F}_q$ is distributed across multiple nodes. Apart from reliability, desirable features include the ability to retrieve the data file by connecting to a subset of the nodes, and the ability to repair a failed node without excessive data download. In the framework of regenerating codes introduced in [1], a codeword is a $\mathbb{F}_q$-matrix of size $(\alpha \times n)$, where each column corresponds to the data stored by a single node. A failed node is regenerated by downloading $\beta \leq \alpha$ symbols from any arbitrary set of d nodes, consuming a *repair bandwidth* of $d\beta$. These d nodes are referred to as helper nodes. Since the entire file can be recovered from any arbitrary set of k nodes, we must have $k \leq d \leq n-1$. Thus a regenerating code is parameterized by the set, $((n,k,d),(\alpha,\beta),K)$.

Our interest here is in *exact-repair* regenerating codes i the contents of a failed node are duplicated in the failed node as opposed to functional repair under which the replacement node is only guaranteed to have the same functional capabilities as the failed node, see [1], [2], [3]. A regenerating code is said to be linear if the $(\alpha \times n)$ code matrix is a linear function of the K data symbols. The regenerating codes constructed in the present paper are linear and have the additional feature that no computations are needed at a helper node, a simple transfer of the contents of the helper node suffice. This is distinct from the class of repair-by-transfer regenerating codes, [4], which have the additional feature that no computations are needed at the replacement node.

B. Storage-Repair-Bandwidth Tradeoff

The cut-set bound of network coding is used in [1] to establish that the parameters of a regenerating code must necessarily satisfy the inequality

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

A regenerating codes is said to be optimal if the above inequality holds with equality and if in addition, reducing either α or β causes the bound to be violated. For given n,k,d,K, there is a tradeoff between the values of (α,β) for which equality holds in (1). The two extreme points in this tradeoff are termed the minimum storage regeneration (MSR) and minimum bandwidth regeneration (MBR) points respectively.

In [5], a product-matrix framework is introduced that permits the construction of MBR codes for all values for $[n,k,d]$, and MSR codes for $d \leq 2k-3$. In [6], high-rate MSR codes with parameters $[n, k = n-2, d = n-1]$ are constructed using Hadamard designs. In [7], high-rate MSR codes are constructed for $d = n-1$; here efficient node-repair is guaranteed only in the case of systematic nodes. A construction for MSR codes with $d = n-1 \geq 2k-1$ is presented in [8] and [9]. The construction of MSR codes for arbitrary values of $[n,k,d]$ remains an open problem, although it has been proven in [10] that exact-repair MSR codes exist for any parameter set $[n,k,d]$ as the file size grows to infinity. In [4], a construction for a family of repair-by-transfer MBR codes is presented. A construction of regenerating codes in which the bottom row form parities of top rows, and where rows themselves are MDS-coded can be found in [11].

The non-extreme points on the tradeoff will be referred to as interior points. As the tradeoff is a piecewise linear relation, there are k points of slope discontinuity, corresponding to $\alpha = (d-p)\beta, \quad p = 0,1,\ldots k-1$. Setting $p = (k-1)$ and 0 respectively yields the MSR and MBR points respectively. The remaining values correspond to interior points. In [4], the authors prove that the interior points of the storage-repair-bandwidth-tradeoff cannot be achieved under exact repair. This raises the open question as to how close one can come to the tradeoff at an interior point. The constructions presented in this paper, are the only known class of explicit codes that correspond to interior points. Their storage-repair-bandwidth performance is not far from that of the tradeoff curve and the codes do very well in comparison with schemes that space share between MSR and MBR points.

We now introduce a normalized version of the classical storage-repair-bandwidth tradeoff that permits comparison across codes possessing the same values of $(\frac{k}{n}, \frac{d}{n})$ but

different n, K. The normalised tradeoff is in terms of two quantities $\Omega := \frac{n\alpha}{K}$ and $\Theta := \frac{nd\beta}{K}$, which we define as the storage overhead and normalized repair bandwidth of the code respectively[1] When we set $d = n - \gamma$, and $\alpha = (d-p)\beta,\ p \in \{0, 1, \cdots, k-1\}$, the tradeoff in (1) translates to the normalised form given below:

$$\Omega \geq \left(\frac{k}{n} - \frac{(k-p)(k-p-1)}{2n(n-\gamma)}\right)^{-1} = \Omega^* \quad (2)$$

$$\Theta \geq \frac{n-\gamma}{n-\gamma-p}\left(\frac{k}{n} - \frac{(k-p)(k-p-1)}{2n(n-\gamma)}\right)^{-1} \quad (3)$$

$$= \Theta^*, \quad (4)$$

where Ω^* and Θ^* represent the minimum possible values of Ω and Θ respectively, and p is the varying parameter.

C. Vector Codes

Regenerating codes can be viewed as vector codes. An $[n, K, d_{\min}, \alpha]$ linear *vector code* $\mathcal{C}$ over a field $\mathbb{F}_q$ is a subset of $(\mathbb{F}_q^{\alpha})^n$ for some $\alpha > 1$, such that given $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$ and $a, b \in \mathbb{F}_q$, $a\mathbf{c} + b\mathbf{c}'$ also belongs to $\mathcal{C}$. As a vector space over $\mathbb{F}_q$, $\mathcal{C}$ has dimension K, termed the scalar dimension of the code.

D. Gabidulin Codes

Let $\mathcal{G} = \{g(x) = \sum_{i=0}^{D-1} g_i x^{q^i} \mid g_i \in \mathbb{F}_q\}$ denote the set of all linearized polynomials of q-degree $\leq (D-1)$ over $\mathbb{F}_{q^N}$, and let $\{P_i\}_{i=1}^{K}$, $N \geq K \geq D$, be a collection of linearly independent elements over $\mathbb{F}_q$ in $\mathbb{F}_{q^N}$. Consider for each $g \in \mathcal{G}$, the vector $(g(P_1), g(P_2), \cdots, g(P_K))$. By representing each element $g(P_i)$ as an N-element vector over $\mathbb{F}_q$, we obtain an $(N \times K)$ matrix over $\mathbb{F}_q$. The resultant collection of q^D matrices turns out to form a maximal rank distance (MRD) code known as the Gabidulin code [13]. In the current paper, we will in several places deal with vectors of the form $(g(P_1), \cdots, g(P_K))$, and it follows that these may also be regarded as codewords drawn from the Gabidulin code.

E. Results

In this paper, we first construct an $(n, k, d = k)$-regenerating code having a layered structure which we term as the canonical code $\mathcal{C}_{\text{can}}$. This code has two auxiliary parameters w and γ satisfying $w \geq 2, \gamma \geq 1, w + \gamma \leq n$ and only requires field size $q > w + \gamma$. We show how starting from a canonical code, it is possible to build a second class of regenerating code with $k < d$ which we term as a layered regenerating code by making suitable use of linearized polynomial evaluations (or equivalently, codewords in the Gabidulin code) as is done in [14]. The extension to the case $k < d$ requires however, an expansion in field size from q to q^K where K is the scalar dimension of the underlying canonical code. Layered regeneration codes possess the attributes listed in the abstract. Finally, we construct codes with local regeneration

[1]If we assume a Poisson-process model of node failures, the number of failures in a fixed interval is proportional the number of nodes n. This explains why the repair bandwidth is normalized to n, details are provided in [12].

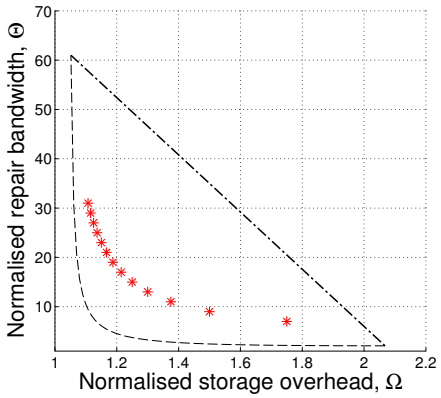

Fig. 1. Plot comparing the performance of the canonical code with $(n = 61, k = d = 58)$ for varying $w \in \{4, 6, 8, \ldots, 28\}$.

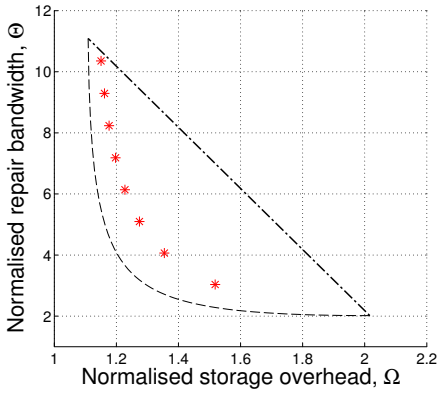

Fig. 2. Plot comparing performance of the layered regenerating code with $(n = 61, k = 55, d = 60)$ for varying $w \in \{2, 3, \ldots, 9\}$.

following the techniques in [15], [14], in which the local codes correspond to the canonical code.

The performance of this class of codes is compared against MBR, MSR codes using the normalized tradeoff. The layered codes operate in the interior region between the MSR and MBR points, and the auxiliary parameter $2 \leq w \leq d$ turns out to determine the specific interior point in the tradeoff. For a wide range of parameters (n, k, d), these codes outperform codes that space-share between MSR and MBR codes [2]. Figures 1 and 2 show the respective performance of canonical codes with $(n = 61, k = 58, d = 58)$, and layered regenerating codes with $(n = 61, k = 55, d = 60)$. As can be seen, in terms of performance, the codes are not far from the tradeoff.

II. Construction of the $(n, k = d, d)$-Canonical Layered Regenerating Code

In this section, we will describe the construction of a family of high-rate, $((n, k, d = k), (\alpha, \beta), K)$ regenerating codes indexed by two auxiliary parameters w, γ satisfying $w \geq 2, \gamma \geq 1, w + \gamma \leq n$. The construction we provide in

[2]Exact-repair MSR codes are not known to exist for every value of (n, k, d)-tuple. Hence the achievability of the space-sharing line joining MSR point and MBR point is not always guaranteed.

this section, assumes $(n, w+\gamma) = 1$. A similar construction can be provided for the general case $(n, w+\gamma) > 1$.

A. Construction of the Canonical Code $\mathcal{C}_{can}$

Let us set

$$L = \frac{1}{n}\binom{n}{w+\gamma} \text{(number of patterns)},$$

$$V = \frac{1}{w}\text{lcm}(w, w+1, \cdots, w+\gamma-1) \text{ (repetition factor)},$$

$M = LV$ (number of layers), $K_c = LVnw$ (scalar dimension).

The structure of the canonical code, can be inferred from the sequential encoding process shown in Fig. 3:

(a) The K_c-tuple message vector $\underline{u} \in \mathbb{F}_q^{K_c}$ is first partitioned into LVn w-tuples:

$$\underline{u} \Rightarrow \left\{ \underline{u}_\tau^{(\ell,\nu)} \mid 1 \le \ell \le L,\ 1 \le \nu \le V,\ 0 \le \tau \le n-1 \right\}.$$

(b) Each w-tuple is then encoded using an $[w+\gamma, w, \gamma+1]$ MDS code to yield LVn codewords

$$\underline{u}_\tau^{(\ell,\nu)} \in \mathbb{F}_q^w \quad \Rightarrow \quad \underline{c}_\tau^{(\ell,\nu)} \in \mathbb{F}_q^{w+\gamma}.$$

(c) The collection of n codewords $\{\underline{c}_\tau^{(\ell,\nu)}\}_{\tau=0}^{n-1}$ is then "threaded" to form a layer $A^{(\ell\,\nu)}$ of the code matrix:

$$\{\underline{c}_\tau^{(\ell,\nu)}\}_{i=0}^{n-1} \xRightarrow{\text{Pattern } \pi^{(\ell)}} A^{(\ell,\nu)}.$$

This threading is carried out with the aid of a pattern $\pi^{(\ell)}$ and is explained in Sections II-B,II-C.

(d) The LV layers are then stacked to form the code matrix

$$C = \begin{bmatrix} A^{(1,1)} \\ A^{(1,2)} \\ \vdots \\ A^{(L,V)} \end{bmatrix}.$$

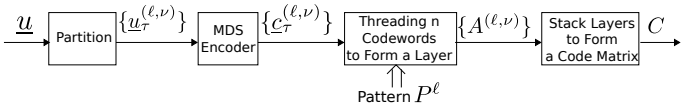

Fig. 3. Encoder of the canonical layered regenerating code.

B. Patterns

There are $\binom{n}{w+\gamma}$ subsets of $[n]$ of size $(w+\gamma)$. We partition these into equivalence classes by declaring two elements to be equivalent if one is a cyclic shift of the other. Since $(n, w+\gamma) = 1$, all equivalence classes contain n elements and the number of equivalence classes is given by $L = \frac{1}{n}\binom{n}{w+\gamma}$. Let

$$\left\{ \pi^{(\ell)} \mid 1 \le \ell \le L \right\} = \left\{ (\pi_1^{(\ell)}, \pi_2^{(\ell)}, \cdots, \pi_{w+\gamma}^{(\ell)}) \mid 1 \le \ell \le L \right\},$$

be the collection of subsets obtained by selecting one representative from each equivalence class. We assume the elements within the subsets to be in ascending numerical order, i.e.,

$$\pi_1^{(\ell)} < \pi_2^{(\ell)} < \cdots < \pi_{w+\gamma}^{(\ell)}, \quad \text{for all } \ell.$$

We will associate with each such ordered subset, a collection of n two-dimensional patterns, each of size $(w+\gamma)\times(w+\gamma)$. This collection includes the fundamental pattern:

$$P^{(\ell)}(0) = \left\{ \left(i, \pi_i^{(\ell)} \right) \mid 1 \le i \le w+\gamma \right\},$$

as well as its n (columnar) cyclic shifts

$$P^{(\ell)}(\tau) = \left\{ \left(i, \pi_i^{(\ell)} \oplus \tau \right) \mid 1 \le i \le w+\gamma \right\},$$

for $0 \le \tau \le (n-1)$, in which $\pi_i^{(\ell)} \oplus \tau$ is addition modulo n. Given a pattern $P^{(\ell)}(\tau)$ we will refer to the $(w+\gamma)$-tuple

$$\pi^{(\ell)} \oplus \tau = (\pi_1^{(\ell)} \oplus \tau,\ \pi_2^{(\ell)} \oplus \tau, \cdots,\ \pi_{w+\gamma}^{(\ell)} \oplus \tau),$$

as its footprint. Thus $\pi^{(\ell)}$ is the footprint of $P^{(\ell)}(0)$.

C. Threading Codewords to Form a Layer

We fix (ℓ, ν) and hence describe the threading process as it applies to the (ℓ, ν)th layer. Consider the collection of n codewords $\{\underline{c}_\tau^{(\ell,\nu)}\}_{\tau=0}^{n-1}$ associated to a layer. The symbols of the τth codeword $\underline{c}_\tau^{(\ell,\nu)}$, $0 \le \tau \le n-1$, are placed (in some arbitrary order) into the n locations

$$P^{(\ell)}(\tau) = \left\{ \left(i, \pi_i^{(\ell)} \oplus \tau \right) \mid 1 \le i \le w+\gamma \right\},$$

$0 \le \tau \le (n-1)$. We will refer to the codewords $\{\underline{c}_\tau^{(\ell,\nu)}\}_{\tau=0}^{n-1}$ as the n threads within a layer. The threading yields a $(w+\gamma\times n)$ matrix which we will denote by $A^{(\ell,\nu)}$. We then repeat this process for each layer, i.e., for all pairs (ℓ, ν). Finally we vertically stack the matrices $A^{(\ell,\nu)}$ to obtain the code matrix as described above. With this the encoding process is complete.

D. Parameters of the Canonical Code

1) Parameter α: Clearly from the layered structure,

$$\begin{aligned} \alpha &= LV(w+\gamma) \\ &= \frac{\text{lcm}(w, w+1, \cdots, w+\gamma-1)}{w}\binom{n-1}{w+\gamma-1}. \end{aligned}$$

2) Parameters d, β: It is straightforward to show that node repair is not possible for $d < n-\gamma$, for this reason, we set $d = n-\gamma$. It remains, of course, to show that a failed node can be repaired by downloading a *fixed* number β of symbols from each of the d helper nodes.

Without loss of generality, we will assume that node η_1 has failed and that the helper nodes are nodes $(\eta_{\gamma+1}, \ldots, \eta_n)$. We will refer to the nodes $(\eta_2, \cdots, \eta_\gamma)$ as idle nodes as they do not participate in the repair process. Let us assume further, that node h is one of the helper nodes. Our interest is in determining the number of symbols that need to be transferred from node h to node η_1 for the purposes of node repair. Node h can transfer one symbol to the replacement for node η_1 iff there is a thread in some layer to which both nodes η_1 and h contribute code symbols. Recall the each thread is an $[w+\gamma, w, \gamma+1]$ MDS code. Let us count the number of threads such that

- both nodes η_1 and h contribute a single code symbol to that thread

- $(p-1)$ of the idle nodes $\{\eta_i \mid 2 \leq i \leq \gamma\}$ each contribute one code symbol to the thread, the remaining idle nodes do not contribute any code symbol to the thread

The total number of such threads, across all the L distinct layers in the code matrix is given by

$$\binom{\gamma-1}{p-1}\binom{n-\gamma}{w+\gamma-p}.$$

Within the erasure code, the situation is that p symbols have been erased and thus a total of $w+\gamma-p$ symbols can serve as helper nodes for node η_1 of which node h is one. Since any w nodes suffice to help node η_1 recover from the erasure, it suffices if node h "on average" contributes a fraction $\frac{w}{w+\gamma-p}$ of code symbols. It is not hard to show that we can ensure that this average is realized, by suitably utilizing the V repetitions of each layer. The number V has been chosen such that for all p,

$$\frac{w+\gamma-p}{w} \mid V.$$

Thus we can ensure that the helper node will always pass on $V\left(\frac{w}{w+\gamma-p}\right)$ code symbols when counted across all V repetitions of the corresponding erasure code. It follows that $d = n-\gamma$ and the value of β is given by

$$\beta \;=\; V\sum_{p=1}^{\gamma}\binom{\gamma-1}{p-1}\binom{n-\gamma}{w+\gamma-p}\frac{w}{w+\gamma-p}. \quad (5)$$

3) Determining k, K and Code Rate R: Arguing as above, if $k < (n-\gamma)$, we will fail to decode at least one thread. Hence $k \geq (n-\gamma)$. On the other hand, by connecting to $d = (n-\gamma)$ we can recover the entire data and hence $k = d$. The scalar dimension of the code is clearly given by $K_c = LVnw$. Not surprisingly, the rate R of the code is given by $\frac{w}{w+\gamma}$.

Remark 1 (Extension to the Case $(n, w+\gamma) \neq 1$): The construction of the canonical code has a straightforward extension to the case when $(n, w+\gamma) \neq 1$. The key difference is that when we partition $(w+\gamma)$ subsets of $[n]$, the equivalence classes will not be of the same size. However, one can, through judicious repetition of some patterns, compensate for this. Details may be found in [12] .

III. Construction of the Layered Regenerating Code for $(n, k < d, d)$

The layered regenerating code $\mathcal{C}_{\text{lrc}}$ builds on code $\mathcal{C}_{\text{can}}$. It also makes use of linearized polynomials (Gabidulin code) along the lines of their usage in [14]. The K message symbols of $\mathcal{C}_{\text{lrc}}$ $\{m_i\}_{i=1}^{K}$ of $\mathcal{C}_{\text{lrc}}$ are first used to construct a linearized polynomial

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

The linearized polynomial is then evaluated at K_c elements $\{\theta_i\}_{i=1}^{K_c}$ of $\mathbb{F}_{q^N}$ which when viewed as vectors over $\mathbb{F}_q$, are linearly independent. The resulting K_c evaluations $\{f(\theta_i)\}$ are than fed as input to an encoder for the canonical code $\mathcal{C}_{\text{can}}$. The output of the encoder for the canonical code is then a codeword belonging to $\mathcal{C}_{\text{lrc}}$.

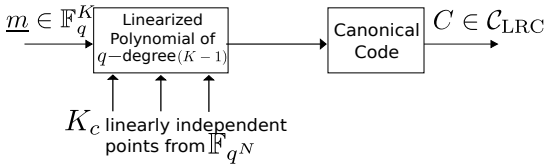

Fig. 4. Encoder of a Layered Regenerating Code.

A. Determining Parameters of $\mathcal{C}_{lrc}$

The parameters α, d, β of $\mathcal{C}_{\text{lrc}}$ are identical to that of the canonical code $\mathcal{C}_{\text{can}}$. It remains to determine expressions for k, K. Towards this, we study the canonical code from a generator-matrix viewpoint.

1) A Generator Matrix for the Canonical Code $\mathcal{C}_{can}$: To obtain a generator matrix for $\mathcal{C}_{\text{can}}$, one needs to first vectorize the code matrix, thus replacing the code matrix by a vector of size $n\alpha = nLV(w+\gamma)$. The generator matrix then describes the linear relation between the LVw input symbols of the canonical code $\mathcal{C}_{\text{can}}$ and the $n\alpha$ output symbols. Let $N_b = n\alpha$. Then the generator matrix G_{can} is of size $(K_c \times N_b)$ where $K_c = LVnw$. From the distributed storage network point of view, the natural vectorization is one in which the symbols in the code matrix are read out in left-to-right, top-to-bottom order. Thus the first α code symbols are those lying in the first column of the code matrix.

2) Rank Accumulation in the Matrix G: The matrix G has the following uniform rank-accumulation property, namely that if one selects a set S containing s thick columns (the collection of α columns of the generator matrix corresponding to the symbols in a node is referred to as a thick column) drawn from amongst the n thick columns comprising G, then the rank the submatrix $G|_S$ of G is independent of the choice of S. Hence the rank of $G|_S$ may simply be denoted by ρ_s. It is straightforward to show that

$$\rho_s \;=\; V\sum_{p=1}^{\min\{s,w+\gamma\}}\binom{s}{p}\binom{n-s}{w+\gamma-p}\min\{p,w\}.$$

We define the rank-accumulation profile of the matrix G as the collection of integers $\{a_i\}_{i=1}^{n}$ given by

$$a_1 \;=\; \rho_1, \qquad a_i \;=\; \rho_i - \rho_{i-1},\ 2 \leq i \leq n.$$

$$\text{We then have, } \rho_s \;=\; \sum_{i=1}^{s} a_i, \quad 1 \leq s \leq n.$$

3) Relating k and K of $\mathcal{C}_{lrc}$: We begin with a useful lemma that is proved in [16]. The proof can also be found in [12].

Lemma 3.1: Let k_0 be the smallest number of thick columns of the generator matrix G of the canonical code $\mathcal{C}_{\text{can}}$ such that the submatrix of G obtained by selecting any k_0 thick columns of G results in a submatrix of rank $\geq K$. Then, by connecting to any k_0 nodes associated to the layered regenerating code $\mathcal{C}_{\text{lrc}}$, a data collector will be able to recover the message symbols $\{m_i\}_{i=1}^{K}$.

It follows from Lemma 3.1 that in order to relate the parameters K, k of $\mathcal{C}_{\text{lrc}}$, it suffices to study the canonical code

$\mathcal{C}_{\text{can}}$ and determine the smallest number k_0 of columns of its generator matrix G, such that the corresponding sub matrix has rank at least K. But from the uniform rank accumulation property of the generator matrix G of the canonical code $\mathcal{C}_{\text{can}}$, this is simply given by

$$k_0 \;=\; \min\{k \mid \rho_k \;\geq\; K\}. \tag{6}$$

Equivalently, the scalar dimension (or the filesize) of the layered regenerating code $\mathcal{C}_{\text{lrc}}$ K for a given value of k is given by

$$K \;=\; V \sum_{p=1}^{\min\{k,w+\gamma\}} \min\{w,p\}\binom{k}{p}\binom{n-k}{w+\gamma-p}. \tag{7}$$

IV. Codes with Canonical-Code-Locality

In this section, we will briefly describe how it is possible to construct codes with locality in which each of the local codes is the canonical (layered regenerating) code $\mathcal{C}_{\text{can}}$. The same technique can also be used to generate codes with locality in which the local codes are the layered regeneration codes $\mathcal{C}_{\text{lrc}}$.

A. Locality in Vector Codes

Let $\mathcal{C}$ be an $[n, K, d_{\min}, \alpha]$ vector code over a field $\mathbb{F}_q$, possessing a $(K \times n\alpha)$ generator matrix G. The i^{th} code symbol, $\mathbf{c}_i$, is said to have (exact) (r, δ) locality, $\delta \geq 2$, if it is possible to puncture the code in coordinates corresponding to a set of indices S with $i \in S$, such that the punctured code $\mathcal{C}|_S$ has length $r + \delta - 1$, and minimum distance δ. The code $\mathcal{C}$ is said to have (r, δ) all-symbol locality if all code symbols have (r, δ) locality. The codes obtained through puncturing will be called local codes. Our interest here is in the construction of a code with exact, all-symbol locality, whose local codes correspond to the canonical code $\mathcal{C}_{\text{can}}$ introduced in Section II-A.

The property of locality allows to minimise the number of node accesses during node-repair. The concept of locality was introduced in [17] for scalar codes for single erasures. Subsequently it was generalised to multiple erasures and later to vector codes; see [15], [18], [19] and [14], [20]. Codes combining benefits of regenerating codes and codes with locality are constructed in [14], [19] and [16].

B. Code Construction

Let $t \geq 2$ and $\{\phi_i\}_{i=1}^{tK_c}$ a collection of elements in $\mathbb{F}_{q^N}$ and let $\{\underline{\phi}_i\}$ denote the representation of the $\{\phi_i\}$ as elements of $\mathbb{F}_q^N$. Given a message vector $[m_1, m_2 \ldots, m_K]^T$, we construct the linearized polynomial

$$h(x) \;=\; \sum_{i=1}^{K} m_i x^{q^{i-1}}, \quad m_i \in \mathbb{F}_{q^N}, \quad N \geq tK_c,$$

and form the tK_c-tuple $[h(\phi_1), h(\phi_2), \ldots, h(\phi_{tK_c})]^T$. This evaluation vector is then partitioned into t evaluation vectors each counting K_c components which are then fed to t respective encoders for the canonical code. The corresponding outputs of these encoders are then concatenated to form the desired codeword. It can be shown that the resultant code is optimal in terms of having the best possible minimum distance for the given scalar dimension.

Acknowledgment

This work is supported in part by the National Science Foundation under Grant No. 0964507 and in part by the NetApp Faculty Fellowship program.

References

[1] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *Information Theory, IEEE Transactions on*, vol. 56, no. 9, pp. 4539–4551, 2010.

[2] Wu, Y. and Dimakis, A.G., "Reducing repair traffic for erasure coding-based storage via interference alignment," in *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*. IEEE, 2009, pp. 2276–2280.

[3] Rashmi, KV and Shah, N.B. and Kumar, P.V. and Ramchandran, K., "Explicit construction of optimal exact regenerating codes for distributed storage," in *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*. IEEE, 2009, pp. 1243–1249.

[4] N. Shah, K. Rashmi, P. Vijay Kumar, and K. Ramchandran, "Distributed Storage Codes With Repair-by-Transfer and Nonachievability of Interior Points on the Storage-Bandwidth Tradeoff," *Information Theory, IEEE Transactions on*, vol. 58, no. 3, pp. 1837–1852, march 2012.

[5] K. Rashmi, N. Shah, and P. Kumar, "Optimal Exact-Regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction," *Information Theory, IEEE Transactions on*, vol. 57, no. 8, pp. 5227–5239, aug. 2011.

[6] D. S. Papailiopoulos, A. G. Dimakis, and V. R. Cadambe, "Repair Optimal Erasure Codes through Hadamard Designs," *CoRR*, vol. abs/1106.1634, 2011.

[7] I. Tamo, Z. Wang, and J. Bruck, "Zigzag Codes: MDS Array Codes with Optimal Rebuilding," *CoRR*, vol. abs/1112.0371, 2011.

[8] C. Suh and K. Ramchandran, "Exact-repair MDS code construction using interference alignment," *Information Theory, IEEE Transactions on*, vol. 57, no. 3, pp. 1425–1442, 2011.

[9] N. Shah, K. Rashmi, P. Kumar, and K. Ramchandran, "Interference Alignment in Regenerating Codes for Distributed Storage: Necessity and Code Constructions," *Information Theory, IEEE Transactions on*, vol. 58, no. 4, pp. 2134–2158, april 2012.

[10] V. R. Cadambe, S. A. Jafar, H. Maleki, K. Ramchandran, and C. Suh, "Asymptotic interference alignment for optimal repair of mds codes in distributed storage," *IEEE Transactions on Information Theory*, vol. 59, no. 5, pp. 2974–2987, 2013.

[11] D. Papailiopoulos, J. Luo, A. Dimakis, C. Huang, and J. Li, "Simple regenerating codes: Network coding for cloud storage," in *INFOCOM, 2012 Proceedings IEEE*, march 2012, pp. 2801 –2805.

[12] B. Sasidharan and P. V. Kumar, "High–rate Regnerating Codes through Layering," *arXiv preprint*, 2013.

[13] E. M. Gabidulin, "Theory of Codes with maximum rank distance," *Information Transmission, Problems of*, vol. 21, no. 7, pp. 1–12, 1985.

[14] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," *CoRR*, vol. abs/1210.6954, 2012.

[15] N. Prakash, G. M. Kamath, V. Lalitha, and P. V. Kumar, "Optimal linear codes with a local-error-correction property," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, july 2012, pp. 2776–2780.

[16] N. Prakash, G. M. Kamath, V. Lalitha, P. V. Kumar, A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Explicit MBR All–Symbol Locality Codes," *preprint*, 2013.

[17] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the Locality of Codeword Symbols," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 18, p. 100, 2011.

[18] D. S. Papailiopoulos and A. G. Dimakis, "Locally repairable codes," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, july 2012, pp. 2771–2775.

[19] Kamath, G.M. and Prakash, N. and Lalitha, V. and Kumar, P.V., "Codes with Local Regeneration," *arXiv preprint arXiv:1211.1932*, 2012.

[20] F. Oggier and A. Datta, "Self-repairing homomorphic codes for distributed storage systems," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 1215–1223.