Locality and Availability of Array Codes Constructed from Subspaces

Natalia Silberstein*‡, Tuvi Etzion†, and Moshe Schwartz*
*Electrical and Computer Engineering, Ben-Gurion University of the Negev,
Beer Sheva 8410501, Israel, schwartz@ee.bgu.ac.il
†Computer Science, Technion – Israel Institute of Technology,
Haifa 3200003, Israel, etzion@cs.technion.ac.il
‡Yahoo! Labs, Haifa 31905, Israel, natalys@cs.technion.ac.il

***Abstract*—Ever-increasing amounts of data are created and processed in internet-scale companies such as Google, Facebook, and Amazon. The efficient storage of such copious amounts of data has thus become a fundamental and acute problem in modern computing. No single machine can possibly satisfy such immense storage demands. Therefore, distributed storage systems (DSS), which rely on tens of thousands of storage nodes, are the only viable solution. Such systems are broadly used in all modern internet-scale systems. However, the design of a DSS poses a number of crucial challenges, markedly different from single-user storage systems. Such systems must be able to reconstruct the data efficiently, to overcome failure of servers, to correct errors, etc. Lots of research was done in the last few years to answer these challenges and the research is increasing in parallel to the increasing amount of stored data.**

The main goal of this paper is to consider codes which have two of the most important features of distributed storage systems, namely, locality and availability. Our codes are array codes which are based on subspaces of a linear space over a finite field. We present several constructions of such codes which are q-analog to some of the known block codes. Some of these codes possess independent intellectual merit. We examine the locality and availability of the constructed codes. In particular we distinguish between two types of locality and availability, node vs. symbol, locality and availability. To our knowledge this is the first time that such a distinction is given in the literature.

I. Introduction

Designing efficient mechanisms to store, maintain, and efficiently access large volumes of data is a highly relevant problem. Indeed, ever-increasing amounts of information are being generated and processed in the data centers of Amazon, Facebook, Google, Dropbox, and many others. The demand for ever-increasing amounts of cloud storage is supplied through the use of Distributed Storage Systems (DSS), where data is stored on a network of nodes (hard drives and solid-state drives).

In the DSS paradigm, it is essential to store data redundantly, in order to tolerate inevitable node failures [1], [8], [20]. Currently, the resilience against node failures is typically afforded by *replication*, where several copies of each data object are stored on different storage nodes. However, replication is highly inefficient in terms of storage capacity. Recently, *erasure-correcting codes* have been used in DSS to reduce the large storage overhead of replicated systems [3], [5], [12].

This work was supported in part by the Israeli Science Foundation (ISF), Jerusalem, Israel, under Grant no. 10/12, and by the Israeli Science Foundation (ISF), Jerusalem, Israel, under Grant no. 130/14.

Apart from storage space, other metrics should be considered when designing an actual DSS. However, in contrast with storage space, these metrics are adversely affected by the straightforward use of simple erasure-correcting codes. One such metric is the *repair bandwidth*: the amount of data that needs to be transferred when a node has failed, and is thus replaced. This metric is highly relevant as a prohibitively large fraction of the network bandwidth in a DSS may be consumed by such repair operations. Let us term *all* the information stored by a DSS as *the file*. Traditional erasure-correcting codes, and in particular *maximum distance separable (MDS)* codes, usually require that *all* the file be downloaded in order to regenerate a failed node. Recently, Dimakis et al. [4] established a tradeoff between the repair bandwidth and the storage capacity of a node, and introduced a new family of erasure-correcting codes, called *regenerating codes*, which attain this tradeoff. In particular, they proved that if a *large number* of storage nodes can be contacted during the repair of a failed node, and only *a fraction of their stored data* is downloaded, then the repair bandwidth can be minimized.

Local repairability of a DSS is an additional property which is highly sought. The corresponding performance metric is termed the *locality* of the coding scheme: the number of nodes that must participate in a repair process when a particular node fails. Local repairability is of significant interest when a cost is associated with contacting each node in the system. This is indeed the case in real world scenarios, for example as the result of network constraints. Codes which enable local repairs of failed system nodes are called *locally repairable codes (LRCs)*. These codes were introduced by Gopalan et al. in [9]. LRCs which also minimize the repair bandwidth, called codes with local regeneration, were considered in [13], [14], [18].

Regenerating codes and LRCs are attractive primarily for the storage of *cold* data: archival data that is rarely accessed. On the other hand, they do not address the challenges posed by the storage of frequently accessed *hot* data. For example, hot-data storage must enable efficient reads of the same data segments by several users *in parallel*. This property is referred to as *availability*. Codes which provide both locality and availability were first proposed in [19].

Regenerating codes are described in terms of stored

978-1-5090-4096-4/17/$31.00 ©2017 IEEE

information in nodes (servers). In other words, regenerating codes are usually array codes [22]. Reconstructing the files and repairing failed nodes are the main tasks of regenerating codes. LRCs and codes with availability are usually described as block codes, and access and/or repair is described in terms of symbols.

In this work we combine the two approaches and discuss two types of locality (availability, respectively), node locality (availability) which resembles the first approach and symbol locality (availability) which resembles the second approach. To our knowledge, such a combined approach was not considered in the literature before.

Our solution approach will be based on array codes, constructed via subspaces of a finite vector space. A subspace approach for DSS was considered for the first time in [10] and later in [17]. Our approach is slightly different from the approach in these two papers. We will design array codes based on subspaces and analyze their locality and availability properties.

The rest of this paper is organized as follows. Preliminaries are given in Section II. Our subspace approach, constructions of codes, and analysis of their locality and availability, are presented in Section III-B. For lack of space we omit all proofs. They will be provided soon in an arXiv version of this work.

II. Preliminaries

Let $\mathbb{F}_q$ denote the finite field of size q. For a natural number $m \in \mathbb{N}$, we use the notation $[m] \triangleq \{1, 2, \dots, m\}$. We use lower-case letters to denote scalars. Overlined letters denote vectors, which by default are assumed to be column vectors. Matrices are denoted by upper-case letters. Most literature denotes codewords (which are usually vectors) by overlined lower-case letters. However, since we also have codewords which are arrays (matrices), these will be denoted by bold lower-case letters. Thus, typically, we shall have a generator matrix G, whose jth column is $\overline{g}_j$, and whose (i, j)th entry is $g_{i,j}$. An array code will usually be denoted by C, whose typical codeword will be denoted by $\mathbf{c}$. We use 0 to denote the scalar zero, $\overline{0}$ for the all-zero vector, and $\mathbf{0}$ for the all-zero matrix. Also, given a (possibly empty) set of vectors, $v_1, \dots, v_m \in \mathbb{F}_q^n$, their span is denoted by $\langle v_1, \dots, v_m \rangle$.

Our main object of study is a linear array code, formally defined as follows: A $[b \times n, M, d]$ *array code* over $\mathbb{F}_q$, denoted C, is a linear subspace of $b \times n$ matrices over $\mathbb{F}_q$. Matrices $\mathbf{c} \in C$ are referred to as *codewords*. The elements of a codeword are denoted by $c_{i,j}$, $i \in [b]$, $j \in [n]$, and are referred to as *symbols*. Columns of codewords are denoted by $\overline{c}_j$, $j \in [n]$. We denote by $M \triangleq \dim(C)$ the *dimension* of the code as a linear space over $\mathbb{F}_q$. The *weight* of an array is defined as the number of non-zero columns, i.e., for $\mathbf{c} \in C$,

$$\mathrm{wt}(\mathbf{c}) \triangleq \left| \{ \overline{c}_j \ : \ \overline{c}_j \neq \overline{0}, j \in [n] \} \right| .$$

Finally, the *minimum distance* of the code, denoted d, is the defined as the minimal weight of a non-zero codeword,

$$d \triangleq \min_{\substack{\mathbf{c} \in C \\ \mathbf{c} \neq \mathbf{0}}} \mathrm{wt}(\mathbf{c}).$$

We make two observations to avoid confusion with other notions of error-correcting codes. The first observation is that by reading the symbols of codewords, column by column, and within each column, from first to last entry, we may flatten the $b \times n$ codewords to vectors of length bn. This results in a code over $\mathbb{F}_q$ of length bn, dimension M, but more often than not, a different minimum distance, since the above definition considers non-zero *columns* and not non-zero symbols. Assume G is an $M \times bn$ generator matrix for the flattened code. By abuse of notation, we shall also call G the *generator matrix* for the original code C. Note that in G, columns $(j-1)b + 1, \dots, jb$, correspond to the symbols appearing in the jth codeword column in C. We shall call these b columns in G by the jth *thick column* of G, similarly to [13]. Thus, G is a matrix comprised of n thick columns, corresponding to the n columns of codewords in C.

The second observation is that we may use the well known isomorphism $\mathbb{F}_q^b \cong \mathbb{F}_{q^b}$, and consider each column of a codeword as a single element from $\mathbb{F}_{q^b}$. We get an $\mathbb{F}_q$-linear code over $\mathbb{F}_{q^b}$ (sometimes called a *vector-linear code*), of length n, minimum distance d, but with a dimension (taken as usual over $\mathbb{F}_{q^b}$) not necessarily M.

In a typical distributed-storage setup, we would like to store a file containing M sectors. We choose $\mathbb{F}_q$ such that it is large enough to contain all possible sectors as symbols. The file is encoded into an array $\mathbf{c} \in C$ from a $[b \times n, M, d]$ array code. Each codeword column of $\mathbf{c}$ is stored in a different node. The minimum distance d of the code ensures that any failure of at most $d - 1$ nodes may be corrected.

Two important properties of codes for distributed storage are *locality* and *availability*. An important feature of this paper is the distinction between *symbol* locality and *node* locality (respectively, availability).

Let C be a $[b \times n, M, d]$ array code. We say a codeword column $j \in [n]$ has *node locality* r_{n}, if its content may be obtained via linear combinations of the contents of the recovery-set columns. More precisely, there exists a recovery set $S = \{j_1, \dots, j_{r_{\mathrm{n}}}\} \subseteq [n] \setminus \{j\}$ of r_{n} other codeword columns, and scalars $a_{\ell,m}^{(i)} \in \mathbb{F}_q$, $i, \ell \in [b]$, $m \in [r_{\mathrm{n}}]$, such that for all $i \in [b]$,

$$c_{i,j} = \sum_{\ell=1}^{b} \sum_{m=1}^{r_{\mathrm{n}}} a_{\ell,m}^{(i)} c_{\ell, j_m} \tag{1}$$

simultaneously for all codewords $\mathbf{c} \in C$. If all codeword columns have this property, we say the code has node locality of r_{n}.

Similarly, we say the code has *symbol locality* r_{s}, if for every coordinate, $i \in [b]$ and $j \in [n]$, there exists a recovery set $S = \{j_1, \dots, j_{r_{\mathrm{s}}}\} \subseteq [n] \setminus \{j\}$ of r_{s} other codeword columns, and scalars $a_{\ell,m} \in \mathbb{F}_q$, $\ell \in [b]$, $m \in [r_{\mathrm{s}}]$, such that for every codeword $\mathbf{c} \in C$,

$$c_{i,j} = \sum_{m=1}^{r_{\mathrm{s}}} \sum_{\ell=1}^{b} a_{\ell,m} c_{\ell, j_m}. \tag{2}$$

Thus, each code symbol may be recovered from the code symbols in r_{s} other codeword columns. It is obvious that $r_{\mathrm{s}} \leqslant r_{\mathrm{n}}$.

Once locality is defined, we can also define availability. The *node availability*, denoted t_{n}, (respectively, the *symbol availability*, denoted t_{s}) is the number of pairwise-disjoint recovery sets (as in the definition of locality) that exist for any codeword column (respectively, symbol). Note that each recovery set should of size at most r_{n} (respectively, r_{s}).

We also recall some useful facts regarding Gaussian coefficients. Let V be a vector space of dimension n over $\mathbb{F}_q$. For any integer $0 \leqslant k \leqslant n$, we denote by $\begin{bmatrix} V \\ k \end{bmatrix}$ the set of all k-dimensional subspaces of V. The *Gaussian coefficient* is defined for n, k, and q as

$$\begin{bmatrix} n \\ k \end{bmatrix}_q = \frac{(q^n-1)(q^{n-1}-1)\dots(q^{n-k+1}-1)}{(q^k-1)(q^{k-1}-1)\dots(q-1)}.$$

Whenever the size of the field, q, is clear from the context, we shall remove the subscript q.

It is well known that the number of k-dimensional subspaces of an n-dimensional space over $\mathbb{F}_q$ is given by $\begin{bmatrix} n \\ k \end{bmatrix}$. In a more general form, the number of k'-dimensional subspaces of V which intersect a given k-dimensional subspace of V in an i-dimensional subspace is given by

$$q^{(k'-i)(k-i)}\begin{bmatrix} n-k \\ k'-i \end{bmatrix}\begin{bmatrix} k \\ i \end{bmatrix}. \tag{3}$$

Additionally, the Gaussian coefficients satisfy the following recursions,

$$\begin{aligned}\begin{bmatrix} n \\ k \end{bmatrix} &= \begin{bmatrix} n-1 \\ k \end{bmatrix} + q^{n-k}\begin{bmatrix} n-1 \\ k-1 \end{bmatrix} \\ &= q^k\begin{bmatrix} n-1 \\ k \end{bmatrix} + \begin{bmatrix} n-1 \\ k-1 \end{bmatrix}. \end{aligned} \tag{4}$$

For more on Gaussian coefficients, the reader is referred to [23, Chapter 24].

III. A Subspace Approach to LRCs

Let C be a $[b \times n, M, d]$ array code over $\mathbb{F}_q$. Throughout this section we further assume that $b \leqslant M$. We now describe an approach to viewing such array codes which will lead to the main results of this section.

Denote $V \triangleq \mathbb{F}_q^M$ the M-dimensional vector space over $\mathbb{F}_q$. Let G be a generator matrix for the (flattened) array code C. For each $j \in [n]$, we define $V_j \in \bigcup_{k=0}^{b} \begin{bmatrix} V \\ k \end{bmatrix}$, to be the column space of the jth thick column of G, i.e.,

$$V_j \triangleq \left\langle \overline{g}_{(j-1)b+1}, \overline{g}_{(j-1)b+2}, \dots, \overline{g}_{jb} \right\rangle.$$

We say V_j is associated with the jth thick column of G, or equivalently, associated with the jth column of the codewords of C.

The following equivalence is fundamental to the constructions and analysis of this section.

Lemma 1. *Let C be a $[b \times n, M, d]$ array code over $\mathbb{F}_q$, and let V_j, $j \in [n]$, be the subspaces associated with the codeword columns. Then $S = \{j_1, \dots, j_m\} \subseteq [n] \setminus \{j\}$ is a recovery set for codeword column $j \in [n]$, if and only if*

$$V_j \subseteq V_{j_1} + V_{j_2} + \dots + V_{j_m}.$$

Similarly, S is a recovery set for symbol (i, j), $i \in [b]$, if

$$\overline{g}_{(j-1)b+i} \in V_{j_1} + V_{j_2} + \dots + V_{j_m},$$

where $\overline{g}_{(j-1)b+i}$ is the ith column in the jth thick column of a generating matrix G for C.

With this equivalence, we may obtain the node/symbol locality/availability using subspace properties of the thick columns of a generating matrix.

A. Generalized Simplex Codes via Subspaces

We start with a construction of array codes which may be considered as a generalization and a q-analog of the classical simplex code, the dual of the Hamming code (see [15, p. 30]).

Construction A. *Fix a finite field $\mathbb{F}_q$, positive integers $b \leqslant M$, and $V = \mathbb{F}_q^M$. Construct a $b \times \begin{bmatrix} M \\ b \end{bmatrix}$ array code whose set of columns are associated with the subspaces $\begin{bmatrix} V \\ b \end{bmatrix}$, each appearing exactly once.*

We make a note here, which is also relevant for the constructions to follow. Once we fix the set of subspaces associated with the codeword columns, the code is constructed in the following way: for each $j \in [n]$, and associated subspace V_j, we arbitrarily choose a set of b vectors from $\mathbb{F}_q^M$ that form a basis for V_j. These b vectors are placed (in some arbitrary order) as the columns comprising the jth thick column of a generator matrix G. The resulting matrix G generates the constructed code[1].

We are now ready for the first claim on the properties of the codes from Construction A.

Theorem 2. *The array code obtained from Construction A is a $[b \times \begin{bmatrix} M \\ b \end{bmatrix}, M, d]$ array code, with*

$$d = \begin{bmatrix} M \\ b \end{bmatrix} - \begin{bmatrix} M-1 \\ b \end{bmatrix} = q^{M-b}\begin{bmatrix} M-1 \\ b-1 \end{bmatrix}.$$

Additionally, except for the all-zero array codeword, all other codewords have the same constant weight d.

We observe that the codes of Construction A form a generalization of simplex codes. When we choose $b = 1$ in Construction A, the simplex code is obtained, a fact that was used in the proof of Theorem 2.

Lemma 3. *The array code obtained from Construction A, with parameters $b < M$, has node locality of $r_{\mathrm{n}} = 2$, and symbol locality of*

$$r_{\mathrm{s}} = \begin{cases} 1 & b > 1, \\ 2 & b = 1. \end{cases}$$

We note that we ignored the case of $b = M$ in the previous lemma, since then the array codewords have a single column, and locality is not defined.

We now turn to consider availability. Symbol availability is trivial.

[1] Permuting the thick columns in the construction results in equivalent codes. If a canonical representation is required, we may choose the basis of each thick column to be in reduced row echelon form.

Corollary 4. *The array code obtained from Construction* A, *with parameters* $1 \leqslant b < M$, *has symbol availability*

$$t_{\mathrm{s}} = \begin{cases} \left[\begin{smallmatrix} M-1 \\ b-1 \end{smallmatrix}\right] - 1 & 1 < b < M, \\ \frac{q^{M-1}-1}{2} & b = 1. \end{cases}$$

Unlike locality, it appears that determining the node availability is a difficult task. We consider only the simplest non-trivial case of $b = 2$.

Lemma 5. *The array code obtained from Construction* A, *with parameters* $2 = b < M$, *has node availability*

$$t_{\mathrm{n}} = \frac{1}{2}\left(\begin{bmatrix} M \\ 2 \end{bmatrix} - 1\right),$$

when q *is even, and*

$$t_{\mathrm{n}} \geqslant \frac{1}{2}\left(\begin{bmatrix} M \\ 2 \end{bmatrix} - 1 - q(q^2+q-1)\begin{bmatrix} M-2 \\ 2 \end{bmatrix}\right),$$

when q *is odd.*

B. Codes from Subspace Designs

In this subsection we focus on constructing codes by using certain subspace designs. We first present a different generalization of simplex codes by using spreads. The resulting code is known, and we analyze it for completeness, and for motivating another construction that uses subspace designs.

Consider a finite field $\mathbb{F}_q$ and the vector space $V \triangleq \mathbb{F}_q^M$. A *b-spread of* V is a set $\{V_1, V_2, \dots, V_n\} \subseteq \left[\begin{smallmatrix} M \\ b \end{smallmatrix}\right]$ such that $V_i \cap V_j = \{\overline{0}\}$ for all $i, j \in [n]$, $i \neq j$, and additionally, $\bigcup_{i\in[n]} V_i = V = \mathbb{F}_q^M$. Thus, except for the zero vector, $\overline{0}$, a spread is a partition of $\mathbb{F}_q^M$ into subspaces. It is known that a b-spread exists if and only if $b|M$. Simple counting shows that the number of subspaces in a spread is

$$n = \frac{q^M - 1}{q^b - 1} = \frac{\left[\begin{smallmatrix} M \\ 1 \end{smallmatrix}\right]}{\left[\begin{smallmatrix} b \\ 1 \end{smallmatrix}\right]}.$$

Let us start with a code obtained from a single spread. This code was already described in [16], in the context of self-repairing codes, and we bring it here for completeness.

Construction B. *Fix a finite field* $\mathbb{F}_q$, *positive integers* $b|M$, *and* $V = \mathbb{F}_q^M$. *Construct a* $b \times \left[\begin{smallmatrix} M \\ 1 \end{smallmatrix}\right] / \left[\begin{smallmatrix} b \\ 1 \end{smallmatrix}\right]$ *array code whose set of columns are associated with the subspaces of a* b*-spread of* V, *each appearing exactly once.*

Theorem 6. *The array code obtained from Construction* B *is a* $[b \times \left[\begin{smallmatrix} M \\ 1 \end{smallmatrix}\right] / \left[\begin{smallmatrix} b \\ 1 \end{smallmatrix}\right], M, q^{M-b}]$ *array code. Additionally, except for the all-zero array codeword, all other codewords have the same constant weight.*

Lemma 7. *The array code obtained from Construction B has symbol locality* $r_s = 2$ *and the bound on the node locality* $2 \leqslant r_n \leqslant \min\{b+1, M/b\}$.

The code of Construction B is also a generalization of the simplex code. Indeed, when we take $b = 1$ the resulting generator matrix is that of a simplex code.

Corollary 8. *When* $M = 2b$, *the code from Construction* B *is an MDS array code with* $r_{\mathrm{n}} = r_{\mathrm{s}} = 2$.

Up to this point we constructed codes by specifying their generator matrix. We now turn to consider their dual codes by reversing the roles of generator and parity-check matrices.

The dual code of the code from Construction A has a small distance $d = 2$, and is therefore not very interesting. However, the code from Construction B presents a more interesting situation.

Lemma 9. *Let* C *be a code from Construction* B. *Then its dual,* $C^{\perp}$, *is a* $[b \times \left[\begin{smallmatrix} M \\ 1 \end{smallmatrix}\right] / \left[\begin{smallmatrix} b \\ 1 \end{smallmatrix}\right], b\left[\begin{smallmatrix} M \\ 1 \end{smallmatrix}\right] / \left[\begin{smallmatrix} b \\ 1 \end{smallmatrix}\right] - M, 3]$ *array code. Additionally,* $C^{\perp}$ *is a perfect array code.*

We note that the code of Lemma 9 has already been described as a perfect byte-correcting code in [6], [11].

At this point we stop to reflect back on Construction A and Construction B. We contend that the two are in fact two extremes of a more general construction using the q-analog of Steiner systems.

Definition 10. *Let* F_q *be a finite field. A* q-analog of a Steiner system *(a* q-Steiner system *for short), denoted* $S_q[t, k, n]$, *is a set of subspaces,* $\mathcal{B} \subseteq \left[\begin{smallmatrix} \mathbb{F}_q^n \\ k \end{smallmatrix}\right]$, *such that every subspace from* $\left[\begin{smallmatrix} \mathbb{F}_q^n \\ t \end{smallmatrix}\right]$ *is contained in exactly one element of* $\mathcal{B}$.

In light of Definition 10, we note that the subspaces associated with the columns of Construction A form a q-Steiner system $S_q[b, b, M]$. Similarly, the subspaces associated with the columns of Construction B form a q-Steiner system $S_q[1, b, M]$. Both are therefore extreme (and trivial) cases of a more general construction we now describe.

Construction C. *Fix a finite field* $\mathbb{F}_q$, *and let* $\mathcal{B} \subseteq \left[\begin{smallmatrix} \mathbb{F}_q^M \\ b \end{smallmatrix}\right]$ *be a* q*-Steiner system* $S_q[t, b, M]$. *Construct an array code whose set of columns are associated with the subspace set* $\mathcal{B}$, *each appearing exactly once.*

The main problem with the approach of Construction C is the fact that we need a q-Steiner system. Such systems are extremely hard to find [2], [21], with the only known ones (different $S_2[2, 3, 13]$), found by computer search [2] (though we hope the recent flurry of activity will produce new q-Steiner systems and make Construction C more appealing).

An alternative approach uses a structure that is "almost" a q-Steiner system, and is more readily available – a subspace transversal design (see [7]).

Definition 11. *Let* $\mathbb{F}_q$ *be a finite field. A* subspace transversal design *of group size* $q^m = q^{n-k}$, *block dimension* k, *and strength* t, *denoted by* $\mathrm{STD}_q(t, k, m)$ *is a triple* $(\mathcal{V}, \mathcal{G}, \mathcal{B})$, *where*

1) $\mathcal{V} \triangleq \left[\begin{smallmatrix} \mathbb{F}_q^n \\ 1 \end{smallmatrix}\right] \setminus \mathcal{V}_0^{(n,k)}$, *called* the points, *where* $\mathcal{V}_0^{(n,k)}$ *is defined to be the set of all* 1*-dimensional subspaces of* $\mathbb{F}_q^n$ *all of whose vectors start with* k *zeros, and where* $|\mathcal{V}| = \left[\begin{smallmatrix} k \\ 1 \end{smallmatrix}\right] q^m$.
2) $\mathcal{G}$ *is a partition of* $\mathcal{V}$ *into* $\left[\begin{smallmatrix} k \\ 1 \end{smallmatrix}\right]$ *classes of size* q^m, *called* the groups.
3) $\mathcal{B} \subseteq \left[\begin{smallmatrix} \mathbb{F}_q^n \\ k \end{smallmatrix}\right]$, *called* the blocks, *is a collection of subspaces that contain only points from* $\mathcal{V}$, *with* $|\mathcal{B}| = q^{mt}$.

4) *Each block meets each group in exactly one point.*
5) *Each t-dimensional subspace of $\mathbb{F}_q^n$, with points only from $\mathcal{V}$, which meets each group in at most one point, is contained in exactly one block.*

An $\mathrm{STD}_q(t,k,m) = (\mathcal{V},\mathcal{G},\mathcal{B})$ *is called* resolvable *if the set $\mathcal{B}$ may be partitioned into sets $\mathcal{B}_1,\dots,\mathcal{B}_s$, called* parallel classes, *where each point is contained in exactly one block of each parallel class $\mathcal{B}_i$.*

Unlike q-Steiner systems, subspace transversal designs are known to exist in a wide range of parameters, as shown in the following theorem [7].

Theorem 12. *[7, Th. 7] For any $1 \leqslant t \leqslant k \leqslant m$, and any finite field $\mathbb{F}_q$, there exists a resolvable $\mathrm{STD}_q(t,k,m) = (\mathcal{V},\mathcal{G},\mathcal{B})$, where the block set $\mathcal{B}$ may be partitioned into $q^{m(t-1)}$ parallel classes, each one of size q^m, such that each point is contained in exactly one block of each parallel class.*

Construction D. *Fix a finite field $\mathbb{F}_q$, $M \geqslant 2b$, and let $(\mathcal{V},\mathcal{G},\mathcal{B})$ be a $\mathrm{STD}_q(t,b,M-b)$ with parallel classes $\mathcal{B}_1,\mathcal{B}_2,\dots,\mathcal{B}_s$. Construct the following two array codes:*

- *An array code C_{par} whose set of columns are associated with the subspaces in a single parallel class, $\mathcal{B}_i$, each appearing exactly once.*
- *An array code C whose set of columns are associated with the subspaces in $\mathcal{B}$, each appearing exactly once.*

The code C_{par} is in fact an auxiliary code we shall use to prove the parameters of the code C, and is perhaps of interest on its own.

Theorem 13. *Let C_{par} be the code from Construction* D. *Then C_{par} is a $[b \times q^{M-b}, M, q^{M-b} - q^{M-2b}]$ array code, with $2^b - 1$ codewords of full weight q^{M-b}, and the other non-zero codewords of weight $q^{M-b} - q^{M-2b}$. Moreover, the symbol locality of C_{par} is $r_{\mathrm{s}} = 2$, and its node locality is*

$$r_{\mathrm{n}} = \begin{cases} 3 & q = 2, \\ 2 & q > 2. \end{cases}$$

Corollary 14. *Let C_{par} be the code from Construction* D. *Then its dual code, $C_{\mathrm{par}}^{\perp}$ is a $[b \times q^{M-b}, bq^{M-b} - M, 3]$ array code that is asymptotically perfect.*

Example 15. *Let $b = 3$, $M = 6$, $q = 2$. A generator matrix G for the $[2 \times 8, 6, 7]$ MDS array code C_{par} from Construction* D *is given by*

$$G = \left(\begin{array}{c|c|c|c|c|c|c|c} 100 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 010 & 010 & 010 & 010 & 010 & 010 & 010 & 010 \\ 001 & 001 & 001 & 001 & 001 & 001 & 001 & 001 \\ 000 & 100 & 001 & 010 & 101 & 011 & 111 & 110 \\ 000 & 010 & 101 & 011 & 111 & 110 & 100 & 001 \\ 000 & 001 & 010 & 101 & 011 & 111 & 110 & 100 \end{array}\right).$$

□

We now move on to examine the second code of Construction D. To avoid degenerate cases, we consider only $t \geqslant 2$.

Theorem 16. *Let C be the code from Construction* D, *with $t \geqslant 2$. Then C is a $[b \times q^{(M-b)t}, M, d]$ array code*

$$d = q^{(M-b)(t-1)}(q^{M-b} - q^{M-2b}).$$

The symbol and node locality of the code satisfy $r_{\mathrm{s}} = 1$, and $r_{\mathrm{n}} \geqslant 2$. Its symbol availability is $t_{\mathrm{s}} = q^{(M-b)(t-1)} - 1$.

References

[1] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G. M. Voelker, "Total recall: system support for automated availability management," *Networked Sys. Design and Implem. (NSDI)*, pp. 337–350, 2004.

[2] M. Braun, T. Etzion, P. R. J. Östergård, A. Vardy, and A. Wassermann, "Existence of q-analogs of Steiner systems," *Forum of Mathematics, Pi*, vol. 4, no. e7, pp. 1–14, 2016.

[3] A. Datta and F. Oggier, "An overview of codes tailor-made for networked distributed data storage," *arXiv:1109.2317*, Sep. 2011.

[4] A. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inform. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[5] A. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. of the IEEE*, vol. 99, pp. 476–489, 2011.

[6] T. Etzion, "Perfect byte-correcting codes," *IEEE Trans. Inform. Theory*, vol. 44, no. 7, pp. 3140–3146, Nov. 1998.

[7] T. Etzion and N. Silberstein, "Codes and designs related to lifted MRD codes," *IEEE Trans. Inform. Theory*, vol. 59, no. 2, Feb. 2013.

[8] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *ACM SIGOPS operating systems review*, vol. 37, no. 5, 2003, pp. 29–43.

[9] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inform. Theory*, vol. 58, no. 11, pp. 6925–6934, Nov. 2012.

[10] H. D. L. Hollmann, "Storage codes; coding rate and repair locality," in *Proc. Int. Conf. on Computing, Networking and Communications (ICNC), San Diego, CA, USA*, Jan. 2013, pp. 830–834.

[11] S. J. Hong and A. M. Patel, "A general class of maximal codes for computer applications," *IEEE Trans. Comput.*, vol. C-21, pp. 1322–1331, 1972.

[12] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, "Erasure coding in Windows Azure storage," in *Proc. USENIX ATC 12, Boston, MA, USA*, 2012, pp. 15–26.

[13] G. Kamath, N. Prakash, V. Lalitha, and P. Kumar, "Codes with local regeneration and erasure correction," *IEEE Trans. Inform. Theory*, vol. 60, no. 8, pp. 4637–4660, Aug. 2014.

[14] G. Kamath, N. Silberstein, N. Prakash, A. Rawat, V. Lalitha, O. Koyluoglu, P. Kumar, and S. Vishwanath, "Explicit MBR all-symbol locality codes," in *Proc. 2013 IEEE Int. Symp. on Inform. Theory (ISIT2013), Istanbul, Turkey*, Jul. 2013, pp. 504–508.

[15] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. North-Holland, 1978.

[16] F. Oggier and A. Datta, "Self-repairing homomorphic codes for distributed storage systems," in *Proc. INFOCOM, Shanghai, China*, Apr. 2011, pp. 1215–1223.

[17] N. Raviv and T. Etzion, "Distributed storage systems based on intersecting subspace codes," in *Proc. 2015 IEEE Int. Symp. on Inform. Theory (ISIT2015), Hong Kong, SAR China*, Jun. 2015, pp. 1462–1466.

[18] A. Rawat, O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," *IEEE Trans. Inform. Theory*, vol. 60, no. 1, pp. 212–236, Jan. 2014.

[19] A. Rawat, D. Papailiopoulos, A. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," in *Proc. 2014 IEEE Int. Symp. on Inform. Theory (ISIT2014), Honolulu, HI, USA*, Jun. 2014, pp. 631–635.

[20] S. C. Rhea, P. R. Eaton, D. Geels, H. Weatherspoon, B. Y. Zhao, and J. Kubiatowicz, "Pond: The oceanstore prototype," in *Proc. 2th USENIX Conference on File and Storage Technologies (FAST), San Francisco, CA, USA*, vol. 3, Mar. 2003, pp. 1–14.

[21] M. Schwartz and T. Etzion, "Codes and anticodes in the Grassman graph," *J. Combin. Theory Ser. A*, vol. 97, no. 1, pp. 27–42, Jan. 2002.

[22] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inform. Theory*, vol. 59, pp. 1597–1616, Mar. 2013.

[23] J. H. van Lint and R. M. Wilson, *A Course in Combinatorics, 2nd Edition*. Cambridge Univ. Press, 2001.