This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2022.3202140

A Generalization of Array Codes with Local Properties and Efficient Encoding/Decoding

Hanxu Hou, *Member, IEEE*, Yunghsiang S. Han, *Fellow, IEEE*, Patrick P. C. Lee, *Senior Member, IEEE*, You Wu, Guojun Han, *Senior Member, IEEE*, and Mario Blaum, *Life Fellow, IEEE*

***Abstract*—An (n,k) *recoverable property* array code is composed of $m \times n$ arrays such that any k out of n columns suffice to retrieve all the information symbols, where $n > k$. Note that maximum distance separable (MDS) array code is a special (n,k) recoverable property array code of size $m \times n$ with the number of information symbols being km. Expanded-Blaum-Roth (EBR) codes and Expanded-Independent-Parity (EIP) codes are two classes of (n,k) recoverable property array codes that can repair any one symbol in a column by locally accessing some other symbols within the column, where the number of symbols m in a column is a prime number. By generalizing the constructions of EBR and EIP codes, we propose new (n,k) recoverable property array codes, such that any one symbol can be locally recovered and the number of symbols in a column can be not only a prime number but also a power of an odd prime number. Also, we present an efficient encoding/decoding method for the proposed generalized EBR (GEBR) and generalized EIP (GEIP) codes based on the LU factorization of a Vandermonde matrix. We show that the proposed decoding method has less computational complexity than existing methods. Furthermore, we show that the proposed GEBR codes have both a larger minimum symbol distance and a larger recovery ability of erased lines for some parameters when compared to EBR codes. We also present a necessary and sufficient condition of enabling EBR codes to recover any r erased lines of a slope for any parameter r, which was an open problem in [2]. Moreover, we show that EBR codes can recover any r consecutive erased lines of any slope for any parameter r.**

***Index Terms*—Array codes, Expanded-Blaum-Roth codes, Expanded-Independent-Parity codes, local repair, efficient encoding/decoding.**

I. Introduction

Modern distributed storage systems require data redundancy to maintain data availability and durability in the presence of failures. Two major redundancy mechanisms are replication and erasure coding. Compared to replication, erasure coding can deliver higher data reliability with much lower storage overhead.

There are many constructions of erasure correcting codes. In this work, we focus on *array codes*, which are a class of erasure correcting codes with only XOR and cyclic-shift operations being involved in the coding process. Array codes have been widely used in storage systems, such as the Redundant Array of Independent Disk (RAID) [3]. Consider an array code of size $m \times n$ elements, in which each element stores one *symbol* in the array code. Among the n columns, the first k columns store $m \times k$ *information* symbols to form k information columns, and the remaining $r = n - k$ columns store $m \times r$ *parity* symbols, encoded from the $m \times k$ information symbols, to form r parity columns. The value of m depends on the code construction, and the m symbols in each column are stored in the same disk (or node) of a storage system.

Maximum distance separable (MDS) array codes are a special class of array codes, where any k out of the n columns can retrieve all $m \times k$ information symbols stored in the k information columns (i.e., providing fault tolerance against any r disk failures). More generally, we define (n,k) *recoverable property array codes* as the $m \times n$ array codes such that we can recover all the information symbols from any k out of the n columns, where the number of information symbols is no larger than km. When the number of information symbols is km, (n,k) recoverable property array codes are reduced to MDS array codes. There are many existing MDS array codes in the literature, and most of them are designed to tolerate two or three failed columns. For example, EVENODD [4], [5] and RDP [6] are two important codes that can correct double disk failures. STAR codes [7], [8] and triple-fault-tolerance codes [9] can correct three disk failures. Examples of array codes that can tolerate four or more column failures include Generalized RDP codes [10], Independent-Parity (also called generalized EVENODD) codes [11], Blaum-Roth (BR) codes [12], the codes in [13], and Rabin-like codes [14], [15].

To minimize the storage overhead, it is important to design codes with the larger length for a given overhead. Modern distributed storage systems often store the data files that are geographically distributed across nodes, racks, and data centers. Data should be accessible even if some nodes, racks, or data centers are offline. This motivates designing storage codes that can locally recover single-symbol failure and quickly recover large correlated failures such as multi-node failure, rack failure and data center failure, and have fast encoding and decoding algorithms. Recently, Expanded-Blaum-Roth (EBR) [2], [16] codes and Expanded-Independent-Parity (EIP) codes

This paper was presented in part at the IEEE Global Communications Conference (GLOBECOM), 2020 [1]. H. Hou is with the School of Electrical Engineering & Intelligentization, Dongguan University of Technology (E-mail: houhanxu@163.com). Y. S. Han is with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China (E-mail: yunghsiang@gmail.com). P. P. C. Lee is with Department of Computer Science and Engineering, The Chinese University of Hong Kong (E-mail: pclee@cse.edu.cuhk.hk). Y. Wu is with Beijing Didi Infinity Technology and Development Co., Ltd. (E-mail: 278008313@qq.com) G. Han is with the School of Information Engineering, Guangdong University of Technology (E-mail: gjhan@gdut.edu.cn). M. Blaum is with IBM Research Division-Almaden (E-mail: mblaum@hotmail.com). This work was partially supported by the National Key R&D Program of China (No. 2020YFA0712300), the National Natural Science Foundation of China (No. 62071121, 61871136), Basic Research Enhancement Program of China under Grant 2021-JCJQ-JJ-0483 and Research Grants Council of HKSAR (AoE/P-404/18).

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2022.3202140

[2] extend BR codes [12] and Independent-Parity codes [11], respectively, and propose to tolerate any r column failures and locally repair one failed symbol within any column (called *local repair property*) by adding some parity symbols into each column. This improves the performance of repairing a failed symbol, as the repair can be locally done within a column without accessing the symbols in other columns. In addition, EBR codes can recover some erased lines of a slope. Therefore, one possible application of EBR codes and EIP codes is the sectors or pages failures in a device, like locally recoverable codes (LRC) [17]. Another possible application is in large-scale distributed storage. We need to explicitly deal with significant correlated failures, such as rack failures, data center failures, and some other correlated failures. EBR codes can quickly recover some erased lines of a slope that can naturally be employed in large-scale distributed storage to recover correlated failures, i.e., some correlated nodes are erased in a way corresponding to the erased lines of a slope.

A. Basics of EBR and EIP Codes

An EBR code is represented by an $m \times m$ array, where $m = k + r$ and $m > 2$ is a prime number. It stores $\alpha \times k$ information symbols in the k information columns with α information symbols each, for some $\alpha < m$, and uses the $\alpha \times k$ sub-array of information symbols as input for encoding. Specifically, for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, m-1$, let $a_{i,j} \in \mathbb{F}_q$ be the element in row i and column j of the $m \times m$ array, where q is a power of 2. For $j = 0, 1, \ldots, k-1$, the m symbols $a_{0,j}, a_{1,j}, \ldots, a_{m-1,j}$ in column j are represented as an *information polynomial*

$$a_j(x) = a_{0,j} + a_{1,j}x + \ldots + a_{m-1,j}x^{m-1}$$

over the quotient ring $\mathbb{F}_q[x]/(1+x^m)$. Given the α information symbols $a_{0,j}, a_{1,j}, \ldots, a_{\alpha-1,j}$, we compute $m - \alpha$ symbols $a_{\alpha,j}, a_{\alpha+1,j}, \ldots, a_{m-1,j}$ for local repair, such that the polynomial $a_j(x)$ is a multiple of $(1+x)g(x)$, where $g(x)$ is a factor of $1+x+\cdots+x^{m-1}$. Similarly, the m symbols in column j with $j = k, k+1, \ldots, m-1$ are represented as a *parity polynomial* $a_j(x)$ over $\mathbb{F}_q[x]/(1 + x^m)$. The relationship between the information polynomials and the parity polynomials is given as

$$\mathbf{H}_{r\times m} \cdot \begin{bmatrix} a_0(x) & a_1(x) & \cdots & a_{m-1}(x) \end{bmatrix}^T = \mathbf{0}^T,$$

where $\mathbf{H}_{r\times m}$ is the $r \times m$ parity-check matrix

$$\mathbf{H}_{r\times m} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & x & x^2 & \cdots & x^{m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x^{r-1} & x^{2(r-1)} & \cdots & x^{(r-1)(m-1)} \end{bmatrix}, \quad (1)$$

and $\mathbf{0}^T$ is an all-zero column of length r. In solving the above linear equations, all the r parity polynomials are multiples of $(1 + x)g(x)$. The resulting codes with the parity-check matrix in Eq. (1) are denoted by EBR$(m, r, q, g(x))$.

An EIP code is an $m \times (m + r)$ array, where $m = k$ and m is a prime number. It stores $\alpha \times m$ information symbols in m columns with α information symbols each, for some $\alpha < m$, and uses the $\alpha \times m$ sub-array of information symbols for encoding. Let $a_{i,j}$ be the element in row i and column j, where $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, m+r-1$. For $j = 0, 1, \ldots, m-1$, given the α information symbols $a_{0,j}, a_{1,j}, \ldots, a_{\alpha-1,j}$, we compute $m - \alpha$ symbols $a_{\alpha,j}, a_{\alpha+1,j}, \ldots, a_{m-1,j}$ for local repair, such that the information polynomial

$$a_j(x) = a_{0,j} + a_{1,j}x + \ldots + a_{m-1,j}x^{m-1}$$

is in $\mathbb{F}_q[x]/(1 + x^m)$ and is a multiple of $(1 + x)g(x)$, where $g(x)$ is a factor of $1 + x + \cdots + x^{m-1}$. For $j = m, m + 1, \ldots, m + r - 1$, the parity polynomials $a_j(x)$ representing the m symbols stored in column j are computed by

$$\begin{aligned} &\begin{bmatrix} a_m(x) & a_{m+1}(x) & \cdots & a_{m+r-1}(x) \end{bmatrix} \\ &= \begin{bmatrix} a_0(x) & a_1(x) & \cdots & a_{m-1}(x) \end{bmatrix} \cdot \\ &\quad \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & x & \cdots & x^{r-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x^{m-1} & \cdots & x^{(r-1)(m-1)} \end{bmatrix}. \end{aligned}$$

The above code is denoted by EIP$(m, r, q, g(x))$.

B. Contributions

In this paper, we propose a generalization that can be used to construct (n, k) recoverable property array codes with new parameters. The following are our main contributions:

1) First, we give constructions of generalized EBR (GEBR) codes and generalized EIP (GEIP) codes that can support more parameters when compared to EBR codes and EIP codes, respectively. We show that the $m \times (m = n = k+r)$ GEBR codes satisfy the (n, k) recoverable property (i.e., all the information symbols can be reconstructed from any k out of $n = m$ columns) if m is a power of an odd prime. The $m \times m$ EBR codes [2] are a special case of our GEBR codes with m a prime number.
2) Second, we present an efficient decoding method for GEBR and GEIP codes based on the LU factorization of a Vandermonde matrix. We show that the proposed LU decoding method has less complexity than existing methods.
3) Third, we show that GEBR codes have a larger minimum symbol distance than EBR codes for some parameters. We also show that GEBR codes can recover more erased lines than EBR codes for some parameters. In addition, we present a necessary and sufficient condition of recovering any r erased lines of slope i for $0 \le i \le r-1$ for any r such that $1 \le r \le m-3$ for EBR codes. Note that the lines of slope i are taken toroidally, and an erased line of slope i means that the m symbols in a line of slope i are erased.

LRC codes [17]–[19] can also locally repair a single-symbol. An example is the code used by Facebook in its f4 storage system [20]. More general construction of LRC is called grid-like codes (with global parity symbols) or product codes (without global parity symbols) [21]. Some constructions of LRC are given in [22]–[24]. The main difference between LRC

and ours is as follows. LRC contains both local parity symbols and global parity symbols, while all the parity symbols in our codes are local parity symbols. Each symbol in our codes can be repaired by either some symbols in the same column or the symbols along a line, while not in LRC [17]. Please refer to Section VII for the detailed comparison of LRC, product codes, and the proposed codes.

C. Paper Organization

The rest of the paper is organized as follows. Section II gives the generalized coding method. Section III presents the construction of GEBR codes based on the coding method and proposes the LU decoding method. Section IV presents the construction of GEIP codes. Section V discusses the minimum symbol distance for the proposed codes. Section VI shows that GEBR codes can recover some erased lines. Section VII compares GEBR codes with other related codes. Section VIII concludes the paper.

II. Generalized Coding Method of Array Codes with Local Properties

In this section, we present a coding method for array codes that can encode an $\alpha \times k$ sub-array of information symbols into an $m \times (k+r)$ array, where each element in the array is in the finite field $\mathbb{F}_q$, q is a power of 2, $m = p\tau$, p is a prime number, and α, k, r, τ are positive integers with $\alpha \leq (p-1)\tau$. The primary objective of the coding method is to extend the constructions of EBR and EIP codes to support much more parameters. In particular, EBR codes can be viewed as a special construction of the proposed coding method with $\tau = 1$ and $k + r = p$, while EIP codes are an explicit construction of the coding method with $\tau = 1$ and $k = p$.

Let $s_{i,j} \in \mathbb{F}_q$ be the element of the $m \times (k+r)$ array in row i and column j, where $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, k+r-1$. The αk information symbols are $s_{i,j}$ with $i = 0, 1, \ldots, \alpha - 1$ and $j = 0, 1, \ldots, k-1$, and the other $m(k+r) - \alpha k$ elements of the array are parity symbols.

For $j = 0, 1, \ldots, k+r-1$, we represent the m symbols stored in column j (i.e., $s_{0,j}, s_{1,j}, \ldots, s_{m-1,j}$) by a polynomial $s_j(x)$ of degree $m-1$ over the ring $\mathbb{F}_q[x]$, i.e.,

$$s_j(x) = s_{0,j} + s_{1,j}x + s_{2,j}x^2 + \cdots + s_{m-1,j}x^{m-1},$$

where $s_j(x)$ with $j = 0, 1, \ldots, k-1$ is an information polynomial and $s_j(x)$ with $j = k, k+1, \ldots, k+r-1$ is a parity polynomial. Let $\mathcal{R}_m(q) = \mathbb{F}_q[x]/(1+x^m)$ be the ring of polynomials modulo $1 + x^m$ with coefficients in $\mathbb{F}_q$. We observe that multiplication by x^i in $\mathcal{R}_m(q)$ can be interpreted as a *cyclic shift*, and hence it does not involve finite field arithmetic nor XOR operations.

Given α information symbols $s_{0,j}, s_{1,j}, \cdots, s_{\alpha-1,j}$, we need to determine $m - \alpha$ parity symbols $s_{\alpha,j}, s_{\alpha+1,j}, \cdots, s_{m-1,j}$, where $j = 0, 1, \ldots, k-1$. Let $g(x)$ be a polynomial with coefficients in $\mathbb{F}_q$ such that $g(x)$ divides $1 + x^\tau + \cdots + x^{(p-1)\tau}$ and $\gcd(g(x), 1+x^\tau) = 1$. Let $1 + x^\tau + \cdots + x^{(p-1)\tau} = g(x)h(x)$. Note that $g(x)$ may not be an irreducible polynomial so that we can factorize $g(x)$ as a product of powers of irreducible polynomials over $\mathbb{F}_q$, i.e.,

$$g(x) = (f_1(x))^{\ell_1} \cdot (f_2(x))^{\ell_2} \cdots (f_t(x))^{\ell_t},$$

where $\ell_i \geq 1$ for $i = 1, 2, \ldots, t$ and $\deg(f_i(x)) \geq \deg(f_j(x))$ for $i > j$.

Let $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$ be the cyclic code of length $m = p\tau$ over $\mathbb{F}_q$ with generator polynomial $(1+x^\tau)g(x)$ and minimum distance d. For $j = 0, 1, \ldots, k-1$, we create $m - \alpha$ parity symbols $s_{\alpha,j}, s_{\alpha+1,j}, \cdots, s_{m-1,j}$ by encoding α information symbols $s_{0,j}, s_{1,j}, \cdots, s_{\alpha-1,j}$, such that the polynomial $s_j(x)$ is in $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$.

Given k information polynomials $s_0(x), s_1(x), \ldots, s_{k-1}(x)$, we can compute the r parity polynomials $s_k(x), s_{k+1}(x), \ldots, s_{k+r-1}(x)$ by taking the product

$$\begin{aligned} &\begin{bmatrix} s_k(x) & s_{k+1}(x) & \cdots & s_{k+r-1}(x) \end{bmatrix} \\ = &\begin{bmatrix} s_0(x) & s_1(x) & \cdots & s_{k-1}(x) \end{bmatrix} \cdot \mathbf{P}_{k\times r} \end{aligned} \tag{2}$$

with operations performed in $\mathcal{R}_{p\tau}(q)$, where $\mathbf{P}_{k\times r}$ is the $k \times r$ encoding matrix which is the remainder sub-matrix of the systematic generator matrix by deleting the identity matrix. We can also compute the r parity polynomials by

$$\begin{bmatrix} s_0(x) & s_1(x) & \cdots & s_{k+r-1}(x) \end{bmatrix} \cdot \mathbf{H}^T_{r\times(k+r)} = \mathbf{0} \tag{3}$$

over $\mathcal{R}_{p\tau}(q)$, where $\mathbf{H}_{r\times(k+r)}$ is an $r \times (k+r)$ *parity-check matrix*.

Note that $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$ is an *ideal* in $\mathcal{R}_{p\tau}(q)$, because $\forall c(x) \in \mathcal{R}_{p\tau}(q), \forall s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$, we have $c(x)s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$. Recall that $g(x)h(x) = 1 + x^\tau + \cdots + x^{(p-1)\tau}$. The polynomial $h(x)$ is called *parity-check polynomial* of $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$, since the multiplication of any polynomial in $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$ and $h(x)$ is zero. We show in the next theorem that $\mathcal{R}_{p\tau}(q)$ is isomorphic to $\mathbb{F}_q[x]/g(x)(1+x^\tau) \times \mathbb{F}_q[x]/(h(x))$ under some specific conditions.

Theorem 1. *When* $\gcd(g(x), h(x)) = 1$ *and* $\gcd(1 + x^\tau, h(x)) = 1$, *the ring* $\mathcal{R}_{p\tau}(q)$ *is isomorphic to* $\mathbb{F}_q[x]/g(x)(1+x^\tau) \times \mathbb{F}_q[x]/(h(x))$.

Proof: When $\gcd(g(x), h(x)) = 1$ and $\gcd(1 + x^\tau, h(x)) = 1$, we have $\gcd(g(x)(1+x^\tau), h(x)) = 1$. By the Chinese Remainder Theorem, we can find an isomorphism between $\mathcal{R}_{p\tau}(q)$ and $\mathbb{F}_q[x]/(g(x)(1+x^\tau)) \times \mathbb{F}_q[x]/(h(x))$. The mapping θ is defined by

$$\theta(a(x)) = (a(x) \bmod (g(x)(1+x^\tau)), a(x) \bmod (h(x))),$$

where $a(x) \in \mathcal{R}_{p\tau}(q)$.

Let $h(x) \bmod g(x)(1+x^\tau)$ be the remainder of dividing $h(x)$ by $g(x)(1+x^\tau)$ which is in $\mathbb{F}_q[x]/(g(x)(1+x^\tau))$ and $g(x)(1+x^\tau) \bmod h(x)$ be the remainder of dividing $g(x)(1+x^\tau)$ by $h(x)$ which is in $\mathbb{F}_q[x]/(h(x))$. Since $\gcd(g(x)(1+x^\tau), h(x)) = 1$, there exists the inverse of $h(x) \bmod g(x)(1+x^\tau)$ in $\mathbb{F}_q[x]/(g(x)(1+x^\tau))$ and denote $(h(x) \bmod g(x)(1+x^\tau))^{-1}$ as the inverse. Similarly, denote $(g(x)(1+x^\tau) \bmod h(x))^{-1}$ as the inverse of $g(x)(1+x^\tau) \bmod h(x)$ in $\mathbb{F}_q[x]/(h(x))$. The inverse mapping θ^{-1} is

$$\theta^{-1}(a_1(x), a_2(x)) = \big(a_1(x)h(x)(h(x) \bmod (g(x)(1+x^\tau)))^{-1} + a_2(x)g(x)(1+x^\tau)((g(x)(1+x^\tau)) \bmod (h(x)))^{-1}\big) \bmod (1+x^{p\tau}),$$

where $a_1(x) \in \mathbb{F}_q[x]/(g(x)(1+x^\tau))$ and $a_2(x) \in \mathbb{F}_q[x]/(h(x))$.

Then we have

$$\begin{aligned}
&\theta(\theta^{-1}(a_1(x),a_2(x)))\\
=&\theta\Big(\big(a_1(x)h(x)(h(x) \bmod (g(x)(1+x^\tau)))\big)^{-1} + a_2(x)g(x)\\
&(1+x^\tau)((g(x)(1+x^\tau)) \bmod (h(x)))^{-1}\big) \bmod (1+x^{p\tau})\Big)\\
=&\Big(\Big(\big(a_1(x)h(x)(h(x) \bmod (g(x)(1+x^\tau)))\big)^{-1} \bmod (g(x)(1+x^\tau))\\
&+ a_2(x)g(x)(1+x^\tau)((g(x)(1+x^\tau)) \bmod (h(x)))^{-1}\big)\\
&\bmod (1+x^{p\tau}) \bmod (g(x)(1+x^\tau))\Big),\\
&\Big(\big(a_1(x)h(x)(h(x) \bmod (g(x)(1+x^\tau)))\big)^{-1} \bmod (h(x)) +\\
&a_2(x)g(x)(1+x^\tau)((g(x)(1+x^\tau)) \bmod (h(x)))^{-1}\big)\\
&\bmod (1+x^{p\tau}) \bmod (h(x))\Big)\Big)\\
=&(a_1(x),a_2(x)),
\end{aligned}$$

and the theorem is proved. ∎

Note that the result in Lemma 2 in [25] can be viewed as a special case of our Theorem 1 with $g(x)=1$ and $q=2$. By Theorem 1, we can directly obtain that $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ is isomorphic to $\mathbb{F}_q[x]/(h(x))$, and we give the isomorphism in the next lemma.

Lemma 2. *When* $\gcd(g(x),h(x)) = 1$ *and* $\gcd(1+x^\tau,h(x)) = 1$*, the ring* $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ *is isomorphic to* $\mathbb{F}_q[x]/(h(x))$*, the isomorphism* $\theta : \mathcal{C}_{p\tau}(g(x),\tau,q,d) \to \mathbb{F}_q[x]/(h(x))$ *is* $\theta(a(x)) = a(x) \bmod h(x)$ *and the inverse isomorphism* $\theta^{-1} : \mathbb{F}_q[x]/(h(x)) \to \mathcal{C}_{p\tau}(g(x),\tau,q,d)$ *is* $\theta^{-1}(a(x)) = a(x)\cdot g(x)(1+x^\tau)\cdot(g(x)(1+x^\tau)) \bmod (h(x))^{-1} \bmod (1+x^{p\tau})$.

In the next lemma, we show a necessary condition of a polynomial in the ring $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$.

Lemma 3. *If the polynomial* $s_j(x) = \sum_{i=0}^{m-1} s_{i,j}x^i$ *is in* $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$*, then the coefficients of polynomial* $s_j(x)$ *satisfy the following equation*

$$\sum_{\ell=0}^{p-1} s_{\ell\tau+\mu,j} = 0, \tag{4}$$

where $\mu = 0,1,\ldots,\tau-1$.

Proof: The proof is similar to that in Theorem 1 in [25]. ∎

When $g(x)=1$, the next lemma shows that the necessary condition given in Lemma 3 is also the sufficient condition.

Lemma 4. *[25, Theorem 1] When* $g(x)=1$*, the polynomial* $s_j(x) = \sum_{i=0}^{m-1} s_{i,j}x^i$ *is in* $\mathcal{C}_{p\tau}(1,\tau,q,d)$ *if and only if Eq.* (4) *holds.*

When $g(x)=1$, we have that the weight (the number of non-zero coefficients) of $s_j(x) \in \mathcal{C}_{p\tau}(1,\tau,q,d)$ is a positive even integer by Lemma 4.

When $g(x)=1$ and $q=2$, the ring $\mathcal{C}_{p\tau}(1,\tau,2,d)$ has been used in the literature to give efficient repair for a family of binary MDS array codes [9], [25] and to provide new constructions of regenerating codes with lower computational complexity [26]. When $g(x)=1$, $q=2$, and $\tau=1$, the ring is discussed in [12], [13], [15], [27]–[29]. When $\tau=1$, the ring is used to construct array codes with local properties [2].

When $g(x)=1$, if we delete the last τ rows of the $m\times(k+r)$ array of our coding method, then the obtained $(p-1)\tau\times(k+r)$ array is reduced to the coding method given in [25].

Note that the ring $\mathcal{C}_{p\tau}(1,\tau,2,2)$ is reduced to a finite field of size $2^{(p-1)\tau}$ if and only if 2 is a primitive element in $\mathbb{Z}_p$ and $\tau = p^i$ for some non-negative integer i [30]. When τ is a power of p and p is a prime number such that 2 is a primitive element in $\mathbb{Z}_p$, we have $g(x)=1$, $d=2$ and $h(x) = 1+x^\tau+\cdots+x^{(p-1)\tau}$ is an irreducible polynomial in $\mathbb{F}_2[x]$.

III. Generalized Expanded-Blaum-Roth Codes

In this section, we first give the construction of GEBR codes and then propose the LU decoding method that can be used in the encoding/decoding procedures of GEBR codes.

A. Construction

The proposed GEBR code is a set of arrays of size $m\times(k+r)$ by encoding $k\alpha$ information symbols, where $m=p\tau$, $\alpha<m$, $k+r\le m$, τ is a positive integer, and p is an odd prime number. The constructed GEBR code is denoted by $\mathsf{GEBR}(p,\tau,k,r,q,g(x))$ with parity-check matrix given as

$$\mathbf{H}_{r\times(k+r)} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1\\ 1 & x & x^2 & \cdots & x^{k+r-1}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 1 & x^{r-1} & x^{2(r-1)} & \cdots & x^{(r-1)(k+r-1)} \end{bmatrix}. \tag{5}$$

Note that we have more than one solution of $s_k(x),s_{k+1}(x),\ldots,s_{k+r-1}(x)$ in Eq. (3). We need to choose one solution such that all r polynomials $s_k(x),s_{k+1}(x),\ldots,s_{k+r-1}(x)$ are in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$. Since we will show the (n,k) recoverable condition of $\mathsf{GEBR}(p,\tau,k,r,q,g(x))$ in Theorem 5 and Theorem 8, we assume that the parameters $p,\tau,k,r,q,g(x)$ satisfy the (n,k) recoverable condition given in Theorem 5 or Theorem 8, and there exists only one solution such that all r polynomials $s_k(x),s_{k+1}(x),\ldots,s_{k+r-1}(x)$ are in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$. For general polynomial $g(x)$, we require that $\gcd(g(x),h(x))=1$, $\gcd(1+x^\tau,h(x))=1$, $1+x^i$ and $h(x)$ are relatively prime over $\mathbb{F}_q[x]$ for $i=1,2,\ldots,k+r-1$, $\mathsf{GEBR}(p,\tau,k,r,q,g(x))$ are (n,k) recoverable property codes. Please refer to Theorem 5 for the detailed proof. When $g(x)=1$, let $\tau=\gamma p^\nu$, where $\nu\ge 0$, $0<\gamma$ and $\gcd(\gamma,p)=1$. The codes $\mathsf{GEBR}(p,\tau,k,r,q,1)$ are (n,k) recoverable property codes if and only if $k+r\le p^{\nu+1}$. Please refer to Theorem 8 for the detailed proof.

The encoding procedure is described as follows. We first replace each entry $a(x)$ of the parity-check matrix in Eq. (5) by $a(x)\cdot g(x)(1+x^\tau)\cdot(g(x)(1+x^\tau)) \bmod (h(x))^{-1} \bmod (1+x^{p\tau})$ that is in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ by Lemma 2 and then solve the r polynomials $s_k(x),s_{k+1}(x),\ldots,s_{k+r-1}(x)$ based on the modified parity-check matrix over $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$.

From row i of the parity-check matrix in Eq. (5), where $i = 0, 1, \ldots, r-1$, the summation of the $k+r$ symbols in each line of slope i of the $m \times (k+r)$ array is zero, i.e.,

$$s_{\ell,0} + s_{\ell-i,1} + s_{\ell-2i,2} + \cdots + s_{\ell-(k+r-1)i,k+r-1} = 0,$$

for $\ell = 0, 1, \ldots, m-1$. The indices are taken modulo m throughout the paper unless otherwise specified. For example, when $i = 1$, we have

$$s_0(x) + xs_1(x) + \cdots + x^{k+r-1}s_{k+r-1}(x) = 0 \bmod (1+x^m).$$

Recall that

$$\begin{aligned} x^j s_j(x) =& x^j \Big(\sum_{\ell=0}^{m-1} s_{\ell,j} x^\ell\Big) \\ =& s_{m-j,j} + s_{m-j+1,j}x + \cdots + s_{0,j}x^j + \\ & s_{1,j}x^{1+j} + \cdots + s_{m-j-1,j}x^{m-1}, \end{aligned}$$

for $j = 1, 2, \ldots, k+r-1$. We can obtain that the summation of the $k+r$ symbols $s_{\ell,0}, s_{\ell-1,1}, \ldots, s_{\ell-(k+r-1),k+r-1}$ in each line of slope $i = 1$ is zero. The following array is an example of $m = 5$ and $k+r = 4$, where the symbols $s_{1,0}, s_{0,1}, s_{4,2}, s_{3,3}$ with bold font are in one line of slope $i = 1$:

$$\begin{bmatrix} s_{0,0} & \boldsymbol{s_{0,1}} & s_{0,2} & s_{0,3} \\ \boldsymbol{s_{1,0}} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & \boldsymbol{s_{3,3}} \\ s_{4,0} & s_{4,1} & \boldsymbol{s_{4,2}} & s_{4,3} \end{bmatrix}.$$

Note that the code proposed in [16] is a special case as $\mathsf{GEBR}(p, \tau = 1, k, r = p-k, q, g(x) = 1)$, and $\mathsf{GEBR}(p, \tau = 1, k, r = p-k, q, g(x))$ is the EBR code in [2]. In the $p \times p$ array of EBR code in [2], the summation of the p symbols along a line of slopes $0, 1, \ldots, r-1$ is zero, the polynomial corresponding to the p symbols in one column is a polynomial in $\mathbb{F}_q[x]/(1+x^p)$ which is a multiple of $g(x)(1+x)$. In the $p\tau \times p\tau$ array of our $\mathsf{GEBR}(p, \tau, k, r = p\tau - k, q, g(x))$, the summation of the symbols in each line of slope $0, 1, \ldots, r-1$ is zero, as like EBR codes. The difference is that the polynomial corresponding to each column of EBR codes is a multiple of $g(x)(1+x)$, where $g(x)$ is a factor of $1 + x + \cdots + x^{p-1}$. While the polynomial corresponding to each column of $\mathsf{GEBR}(p, \tau, k, r = p\tau - k, q, g(x))$ is a multiple of $g(x)(1+x^\tau)$, where $g(x)$ is a factor of $1 + x^\tau + \cdots + x^{(p-1)\tau}$ such that $\gcd(g(x), 1+x^\tau) = 1$.

B. The (n, k) Recoverable Property

When we say that a code is (n, k) recoverable or satisfies the (n, k) recoverable property, it means that the code can recover up to any r erased columns out of the $k+r$ columns. One necessary and sufficient (n, k) recoverable condition of $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ with $\gcd(g(x), h(x)) = 1$ and $\gcd(1+x^\tau, h(x)) = 1$ is given in the next theorem.

Theorem 5. *When* $\gcd(g(x), h(x)) = 1$ *and* $\gcd(1 + x^\tau, h(x)) = 1$*, we can compute all the* $k\alpha$ *information symbols from any* k *out of* $k+r$ *polynomials* $s_0(x), s_1(x), \ldots, s_{k+r-1}(x)$*, if and only if, the two polynomials* $1+x^i$ *and* $h(x)$ *are relatively prime over* $\mathbb{F}_q[x]$*, where* $i = 1, 2, \ldots, k+r-1$.

Proof: By Lemma 2, it is sufficient to show that the determinant of any $r \times r$ sub-matrix of $\mathbf{H}_{r\times(k+r)}$ in Eq. (5) is invertible over $\mathbb{F}_q[x]/h(x)$. Any $r \times r$ sub-matrix of $\mathbf{H}_{r\times(k+r)}$ is a Vandermonde matrix, the determinant can be written as the multiplication of power of x and r different factors $1+x^i$, where $i \in \{1, 2, \ldots, k+r-1\}$. Note that the coefficient of constant term of $h(x)$ is non-zero, we have $\gcd(x^j, h(x)) = 1$ for any positive integer j. The determinant can be viewed as a polynomial in $\mathbb{F}_q[x]/h(x)$ after modulo $h(x)$, and is invertible over the ring $\mathbb{F}_q[x]/h(x)$. Therefore, we can compute all the $k\alpha$ information symbols from any k out of $k+r$ polynomials, if and only if, $1+x^i$ is invertible over $\mathbb{F}_q[x]/h(x)$ for all $i = 1, 2, \ldots, k+r-1$. ∎

If τ is a power of 2 and $g(x) = 1$, we have

$$h(x) = 1 + x^\tau + \ldots + x^{(p-1)\tau} = (1 + x + \ldots + x^{p-1})^\tau.$$

We can check that $\gcd(g(x) = 1, h(x)) = 1$ and $\gcd(1 + x^\tau, h(x) = (1+x+\ldots+x^{p-1})^\tau) = 1$. The (n, k) recoverable condition in Theorem 5 is reduced to that $1+x^i$ and $1 + x + \ldots + x^{p-1}$ are relatively prime over $\mathbb{F}_q[x]$ when τ is a power of 2. Note that $1+x^i$ and $1 + x + \ldots + x^{p-1}$ are relatively prime over $\mathbb{F}_q[x]$ for $i = 1, 2, \ldots, p-1$ [12]. Therefore, when τ is a power of 2 and $g(x) = 1$, $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ are (n, k) recoverable if $k + r \le p$ and are not (n, k) recoverable if $k + r > p$.

According to Theorem 5, $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ are (n, k) recoverable if and only if $\gcd(h(x), 1+x^i) = 1$ for $i = 1, 2, \ldots, k+r-1$. In the following, we present an equivalent necessary and sufficient (n, k) recoverable condition.

Lemma 6. *The codes* $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ *are* (n, k) *recoverable if and only if the following equation*

$$(1+x^i)s(x) = c(x) \bmod (1+x^{p\tau}) \tag{6}$$

has a unique solution $s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$*, given that* $c(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$ *and* $i \in \{1, 2, \ldots, k+r-1\}$.

Proof: First we prove that Eq. (6) has a unique solution $s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$, given that $c(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$, if and only if, $1+x^i$ and $h(x)$ are relatively prime over $\mathbb{F}_q[x]$ for $i \in \{1, 2, \ldots, k+r-1\}$.

($\Longrightarrow$) Since both $s(x)$ and $c(x)$ are in $\in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$, we have

$$\begin{aligned} & (1+x^i)s(x) = c(x) \bmod (1+x^{p\tau}) \\ \leftrightarrow \quad & (1+x^i)a(x)(1+x^\tau)g(x) = b(x)(1+x^\tau)g(x) \\ & \bmod (1+x^\tau)g(x)h(x) \\ \leftrightarrow \quad & (1+x^i)a(x) = b(x) \bmod h(x). \end{aligned} \tag{7}$$

Assume that $\gcd(1+x^i, h(x)) = d(x)$. Then, $d(x) | b(x)$. Since $c(x)$ is any element in $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$, this is possible only when $d(x) = 1$.

($\Longleftarrow$) Note that, in Eq. (7), both $\deg(a(x))$ and $\deg(b(x))$ are less than $\deg(h(x))$. Given any valid $b(x)$, we need to prove that there exists only one solution $a(x)$ for Eq. (7). Let $1 + x^i \bmod h(x) = e(x)$. Since $\gcd(1+x^i, h(x)) = 1$, we have $\gcd(e(x), h(x)) = 1$ such that the inverse of $e(x)$ exists. Note that $\deg(a(x)) < \deg(h(x))$ and the only solution $a(x)$

for Eq. (7) is $a(x) = e(x)^{-1}b(x) \bmod h(x)$. By Theorem 5, this completes the proof. ∎

Next we prove that to determine the condition given in Eq. (6), we only need to check for the case $c(x) = 0$.

Lemma 7. *Eq.* (6) *has a unique solution* $s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$*, given that* $c(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$ *if and only if Eq.* (6) *has a unique solution* $s(x) = 0$ *when* $c(x) = 0$*.*

Proof: The "if" part is obvious such that we only need to prove the "only if" part. By Eq. (6), we define the transformation $g_i : \mathcal{C}_{p\tau}(g(x), \tau, q, d) \longrightarrow \mathcal{C}_{p\tau}(g(x), \tau, q, d)$ as

$$g_i(s(x)) = (1 + x^i)s(x) \bmod (1 + x^{p\tau}),$$

where $s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$. It is easy to see that this transformation is linear. Since $(1 + x^i)s(x) = 0$ has unique solution $s(x) = 0$, the kernel of g_i is of dimension 0. According to the rank theorem, the dimension of the range of g_i is the same as the dimension of $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$. Hence, g_i is a one-to-one and onto mapping such that, for any $g_i(s(x)) = c(x)$, Eq. (6) has a unique solution $s(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$, given that $c(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$. ∎

When $g(x) = 1$, we can locally recover any one symbol with the minimum parity symbol (one local parity symbol for $p - 1$ data symbols in each data column), which is practically interesting in storage systems. For example, the LRC implemented in Facebook [20] employs one local parity symbol in each group. In the following, we investigate the necessary and sufficient (n, k) recoverable condition when $g(x) = 1$. When $g(x) \neq 1$, the (n, k) recoverable condition of $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ is a subject of future work.

Theorem 8. *Let* $\tau = \gamma p^\nu$*, where* $\nu \geq 0$*,* $0 < \gamma$ *and* $\gcd(\gamma, p) = 1$*. Then, the codes* $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ *are* (n, k) *recoverable if and only if* $k + r \leq p^{\nu+1}$*.*

Proof: We need to show that the codes $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ are (n, k) recoverable if $k + r \leq p^{\nu+1}$ and are not (n, k) recoverable if $k + r > p^{\nu+1}$.

We first consider that $k+r \leq p^{\nu+1}$. Let $1 \leq i \leq k+r-1 \leq p^{\nu+1}-1$. In order to prove that $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ are (n, k) recoverable, we have to show that Eq. (6) has a unique solution $s(x) = 0$ when $c(x) = 0$ for all $1 \leq i \leq p^{\nu+1} - 1$ by Lemma 7.

Suppose that we can find a non-zero polynomial $s(x) \in \mathcal{C}_{p\tau}(1, \tau, q, d)$ such that $(1+x^i)s(x) = 0 \bmod (1+x^{p\tau})$ holds, we can deduce a contradiction as follows. Without loss of generality, let $s(x) = \sum_{v=0}^{p\tau-1} s_v x^v$ and $s_0 = 1$. Then we have

$$s_{((\ell-1)i) \bmod p\tau} + s_{(\ell i) \bmod p\tau} = 0,$$

for $0 \leq \ell \leq p\tau - 1$. By induction, we have

$$s_{(\ell i) \bmod p\tau} = 1. \tag{8}$$

Let $c = \gcd(i, p\tau)$. Recall that $1 \leq i \leq p^{\nu+1} - 1$, we have $c = \gcd(i, p\tau) = \gcd(i, \gamma p^{\nu+1}) = \gcd(i, \gamma p^\nu) = \gcd(i, \tau)$. Then,

$$\{(\ell i) \bmod p\tau : 0 \leq \ell \leq p\tau - 1\} = \{\ell c : 0 \leq \ell \leq \frac{p\tau}{c} - 1\}. \tag{9}$$

Since, in particular, c divides τ, we have

$$\{\ell\tau : 0 \leq \ell \leq p - 1\} \subset \{\ell c : 0 \leq \ell \leq \frac{p\tau}{c} - 1\}. \tag{10}$$

By Eq. (8), Eq. (9) and Eq. (10), we have $s_{\ell\tau} = 1$ for all $0 \leq \ell \leq p - 1$ and therefore $\sum_{\ell=0}^{p-1} s_{\ell\tau} = 1$, which contradicts to $\sum_{\ell=0}^{p-1} s_{\ell\tau} = 0$ (since $s(x) \in \mathcal{C}_{p\tau}(1, \tau, q, d)$).

Next, we consider that $k + r > p^{\nu+1}$. We will show that Eq. (6) has a non-zero solution $s(x) \in \mathcal{C}_{p\tau}(1, \tau, q, d)$ when $c(x) = 0$ and $i \in \{1, 2, \ldots, k + r - 1\}$.

Note that $m = p\tau = \gamma p^{\nu+1}$. Let

$$s_0(x) = 1 + x^{p^{\nu+1}} + x^{2p^{\nu+1}} + \cdots + x^{(\gamma-1)p^{\nu+1}}, \tag{11}$$
$$s_1(x) = x^{p^\nu} + x^{p^\nu + p^{\nu+1}} + \cdots + x^{p^\nu + (\gamma-1)p^{\nu+1}}, \tag{12}$$

and $s(x) = s_0(x) + s_1(x) \bmod (1 + x^m)$. We first show that the polynomial $s(x) = s_0(x) + s_1(x)$ is in $\mathcal{C}_{p\tau}(1, \tau, q, d)$.

Note that $s_1(x) = x^{p^\nu} s_0(x)$, we have $1 + x^m = 1 + x^{\gamma p^{\nu+1}}$ and $(1 + x^{p^{\nu+1}})s_0(x) = 1 + x^{\gamma p^{\nu+1}}$. Note that $1 + x^\tau = 1 + x^{\gamma p^\nu}$ divides $1 + x^{\gamma p^{\nu+1}}$. Since $\gcd(\gamma, p) = 1$, by Euclidean algorithm, $\gcd(1 + x^{p^{\nu+1}}, 1 + x^{\gamma p^\nu}) = 1 + x^{p^\nu}$. Since $(1 + x^{p^{\nu+1}})(1 + x^{p^{\nu+1}} + x^{2p^{\nu+1}} + \cdots + x^{(\gamma-1)p^{\nu+1}}) = 1 + x^{\gamma p^{\nu+1}} = (1 + x^{\gamma p^\nu})q(x)$ and $\gcd(1 + x^{p^{\nu+1}}, 1 + x^{\gamma p^\nu}) = 1 + x^{p^\nu}$, we have $(1 + x^{\gamma p^\nu}) | (1 + x^{p^\nu})(1 + x^{p^{\nu+1}} + x^{2p^{\nu+1}} + \cdots + x^{(\gamma-1)p^{\nu+1}})$. Hence, $s(x) = s_0(x) + s_1(x) = (1 + x^{p^\nu})(1 + x^{p^{\nu+1}} + x^{2p^{\nu+1}} + \cdots + x^{(\gamma-1)p^{\nu+1}}) \bmod (1 + x^m)$ is also divided by $1 + x^{\gamma p^\nu} = 1 + x^\tau$, i.e., $s(x) \in \mathcal{C}_{p\tau}(1, \tau, q, d)$. It is clear that $s_0(x) + s_1(x) \neq 0$.

From the definitions of $s_0(x)$ and $s_1(x)$, we have

$$\begin{aligned} x^{p^{\nu+1}} s_0(x) =& x^{p^{\nu+1}} + x^{2p^{\nu+1}} + x^{3p^{\nu+1}} + \cdots + x^{\gamma p^{\nu+1}} \\ =& x^{p^{\nu+1}} + x^{2p^{\nu+1}} + x^{3p^{\nu+1}} + \cdots + 1 \\ =& s_0(x) \bmod (1 + x^m), \end{aligned}$$

where the second equation above comes from that $x^{\gamma p^{\nu+1}} = x^m = 1 \bmod (1 + x^m)$. Similarly, we can obtain that $x^{p^{\nu+1}} s_1(x) = s_1(x) \bmod (1 + x^m)$. Therefore, we have

$$(1+x^{p^{\nu+1}})s(x) = (1+x^{p^{\nu+1}})(s_0(x)+s_1(x)) = 0 \bmod (1+x^m),$$

and $s(x) \neq 0$ is a solution to $(1 + x^{p^{\nu+1}})s(x) = 0 \bmod (1 + x^m)$. The theorem is proved. ∎

We can directly obtain the following result from Theorem 8.

Corollary 9. *The codes* $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ *with* $k+r = p\tau$ *are* (n, k) *recoverable if and only if* $\tau = p^\nu$*, where* ν *is a non-negative integer.*

By Theorem 5, the codes $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ are (n, k) recoverable if and only if $\gcd(1 + x^i, h(x)) = 1$ for all $i = 1, 2, \ldots, k + r - 1$. When $g(x) = 1$, we have $h(x) = 1 + x^\tau + \cdots + x^{(p-1)\tau}$. According to Corollary 9, the codes $\mathsf{GEBR}(p, \tau, k, r = m - k, q, 1)$ are (n, k) recoverable if p is an odd prime and τ is a power of p. Combining the results in Theorem 5 and Corollary 9, we can directly obtain the following theorem.

Theorem 10. *If* p *is an odd prime and* τ *is a power of* p*, then we have* $\gcd(1 + x^i, 1 + x^\tau + \ldots + x^{(p-1)\tau}) = 1$ *for all* $i = 1, 2, \ldots, p\tau - 1$*.*

Since $\mathcal{C}_{p\tau}(g(x), \tau, q, d)$ is a cyclic code, the proposed $\mathsf{GEBR}(p, \tau, k, r, q, g(x))$ can recover either a burst of up to $\tau + \deg(g(x))$ (consecutive) erased symbols or up to $d-1$ erased symbols in a column. Specifically, we can recover a burst of up

to $\tau + \deg(g(x))$ erased symbols as follows. First, cyclic-shift the polynomial such that the $\tau + \deg(g(x))$ erased symbols are in the last $\tau + \deg(g(x))$ locations. Then, encode the first α symbols of the obtained shifted polynomial systematically. Finally, apply the inverse cyclic-shift to the encoded polynomial to obtain the decoded polynomial.

Example 1. *Consider the code* $\mathsf{GEBR}(3,3,6,3,2,1)$*, i.e.,* $\tau = p = 3$*,* $k = 6$ *and* $r = 3$*. The entries of an array in the code can be represented by the nine polynomials*

$$\begin{aligned} s_j(x) =& s_{0,j} + s_{1,j}x + s_{2,j}x^2 + s_{3,j}x^3 + s_{4,j}x^4 + s_{5,j}x^5 + \\ & (s_{0,j} + s_{3,j})x^6 + (s_{1,j} + s_{4,j})x^7 + (s_{2,j} + s_{5,j})x^8, \end{aligned}$$

where $0 \le j \le 8$ *and the 36 information symbols are* $s_{i,0}, s_{i,1}, s_{i,2}, s_{i,3}, s_{i,4}, s_{i,5}$ *with* $0 \le i \le 5$*. The parity-check matrix of the code is*

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 & x^8 \\ 1 & x^2 & x^4 & x^6 & x^8 & x^{10} & x^{12} & x^{14} & x^{16} \end{bmatrix}.$$

According to Corollary 9, the code $\mathsf{GEBR}(3,3,6,3,2,1)$ *is* (n,k) *recoverable. Also, each column can recover up to three consecutive erasures.*

C. Efficient Decoding

In the encoding/decoding procedures of $\mathsf{GEBR}(p,\tau,k,r,q,g(x))$, we need to solve a Vandermonde linear system over $\mathcal{R}_{p\tau}(q)$ such that the solved polynomials are in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$. In the following, we present an efficient decoding method for solving the Vandermonde linear system over $\mathcal{R}_{p\tau}(q)$ based on the LU factorization of the Vandermonde matrix. The efficient LU decoding algorithm we propose relies on an efficient algorithm for division by $1+x^b$, and we first present the efficient division.

1) Efficient Division by $1+x^b$ *over* $\mathcal{R}_{p\tau}(q)$*:* We need to first give an efficient decoding algorithm for dividing by $1+x^b$ over $\mathcal{R}_{p\tau}(q)$ before showing the efficient LU decoding method, where b is a positive integer such that $1+x^b$ and $h(x)$ are relatively prime. Given the integer b and the polynomial $f(x) \in \mathcal{C}_{p\tau}(g(x),\tau,q,d)$, we want to solve $r(x) \in \mathcal{C}_{p\tau}(g(x),\tau,q,d)$ from the equation

$$(1+x^b)r(x) = f(x) \bmod (1+x^{p\tau}). \tag{13}$$

The next lemma shows a decoding algorithm for solving $r(x) \in \mathcal{C}_{p\tau}(g(x),\tau,q,d)$ from Eq. (13) when $\gcd(b,p)=1$.

Lemma 11. *Consider Eq.* (13) *with* $1 \le b < p\tau$ *and* $\gcd(b,\tau) = a$*. If* $\gcd(b,p)=1$*, then we can first compute the coefficients* r_j *of the polynomial* $r(x)$ *with* $j = 0,1,\ldots,a-1$ *by*

$$r_j = \sum_{u=1}^{\frac{p-1}{2}} \sum_{\ell=1}^{\tau} f_{(2u-1)\tau b + \ell b + j} \tag{14}$$

and the other coefficients of $r(x)$ *iteratively by*

$$r_{b\ell+j} = f_{b\ell+j} + r_{b(\ell-1)+j} \tag{15}$$

with the index ℓ *running from 1 to* $p\tau/a - 1$ *and* $j = 0,1,\ldots,a-1$.

Proof: See Appendix A. ∎

Lemma 20 in [2] and Lemma 4 in [25] are special case of Lemma 11 with $\tau = 1$ and $g(x) = 1$, respectively. By Lemma 11, there are $a(\frac{p-1}{2} \cdot \tau - 1) + (p\tau - a)$ XORs involved in solving $r(x)$ from Eq. (13). In particular, if $a = 1$, we have that the number of involved XORs is $\frac{3p\tau-\tau-4}{2}$.

Example 2. *Consider the example of* $p = 7$ *and* $\tau = 2$*. Let* $f(x) = 1 + x + x^6 + x^7 + x^{10} + x^{11} + x^{12} + x^{13} \in \mathcal{C}_{7\cdot 2}((1+x^2+x^6),2,2,4)$*, i.e.,* $f_0 = f_1 = f_6 = f_7 = f_{10} = f_{11} = f_{12} = f_{13} = 1$ *and* $f_2 = f_3 = f_4 = f_5 = f_8 = f_9 = 0$*. We want to solve* $r(x)$ *from* $(1+x^3)r(x) = f(x)$*. According to Eq.* (14)*, we have*

$$r_0 = f_9 + f_{12} + f_7 + f_{10} + f_5 + f_8 = 1.$$

The other coefficients can be computed as

$$\begin{aligned} r_1 =& r_4 = r_6 = r_9 = r_{10} = r_{11} = 0, \\ r_2 =& r_3 = r_5 = r_7 = r_8 = r_{12} = r_{13} = 1, \end{aligned}$$

according to Eq. (15)*. Therefore,* $r(x) = 1 + x^2 + x^3 + x^5 + x^7 + x^8 + x^{12} + x^{13}$*. We can check that* $r(x) \in \mathcal{C}_{7\cdot 2}((1+x^2+x^6),2,2,4)$.

When τ is a power of an odd prime p and $g(x) = 1$, $\mathsf{GEBR}(p,\tau,k,r = m-k,q,1)$ are (n,k) recoverable by Corollary 9. If $\gcd(b,p) = 1$, then we can solve Eq. (13) by Lemma 11; otherwise, if b is a multiple of p, then the decoding method is as follows.

Lemma 12. *Consider that* $\tau = p^\nu$*, where* ν *is a positive integer. Let* $b = up^s$*, where* $\gcd(u,p) = 1$ *and* $1 \le s \le \nu$*. We can compute the coefficients* $r_{p^{\nu+1}-p^s+j}$ *of the polynomial* $r(x)$ *in Eq.* (13) *with* $j = 0,1,\ldots,p^s-1$ *by*

$$r_{p^{\nu+1}-p^s+j} = \sum_{i=0}^{\frac{p^{\nu-s+1}-3}{2}} f_{2iup^s+up^s+j}, \tag{16}$$

and the other coefficients of $r(x)$ *iteratively by*

$$r_{p^{\nu+1}-p^s+j+\ell up^s} = f_{p^{\nu+1}-p^s+j+\ell up^s} + r_{p^{\nu+1}-p^s+j+(\ell-1)up^s},$$

where $\ell = 1,2,\ldots,p^{\nu-s+1}-1$*. Recall that the indices are taken modulo* $m = p^{\nu+1}$ *throughout the paper.*

Proof: See Appendix B. ∎

By Lemma 12, there are $p^s(\frac{p^{\nu-s+1}-3}{2}) + p^{\nu+1} - p^s = \frac{3p\tau-5p^s}{2}$ XORs involved in solving $r(x)$ from Eq. (13).

Example 3. *Consider the example of* $p = \tau = 3$*. Let* $f(x) = 1 + x + x^3 + x^7 \in \mathcal{C}_{3\cdot 3}(1,3,2,2)$*, i.e.,* $f_0 = f_1 = f_3 = f_7 = 1$ *and* $f_2 = f_4 = f_5 = f_6 = f_8 = 0$*. We want to solve* $r(x)$ *from* $(1+x^3)r(x) = f(x)$*. According to Eq.* (16) *in Lemma 12, we have*

$$\begin{aligned} r_6 =& f_3 = 1, \\ r_7 =& f_4 = 0, \\ r_8 =& f_5 = 0, \end{aligned}$$

and the other coefficients are $r_0 = r_2 = r_5 = 0$ *and* $r_1 = r_3 = r_4 = 1$*. Therefore,* $r(x) = x + x^3 + x^4 + x^6$*, which is in* $\mathcal{C}_{3\cdot 3}(1,3,2,2)$.

2) LU Decoding Method: Let $\mathbf{V}_{r\times r}(\mathbf{a})$ be an $r \times r$ Vandermonde matrix,

$$\mathbf{V}_{r\times r}(\mathbf{a}) = \begin{bmatrix} 1 & x^{a_1} & \cdots & x^{(r-1)a_1} \\ 1 & x^{a_2} & \cdots & x^{(r-1)a_2} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x^{a_r} & \cdots & x^{(r-1)a_r} \end{bmatrix}, \tag{17}$$

where $\mathbf{a} = (a_1, \cdots, a_r)$ and $a_1, a_2, \ldots, a_r$ are distinct integers that range from 0 to $k + r - 1$. Let $\mathbf{u} = (u_1(x), \ldots, u_r(x)) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)^r$ and $\mathbf{v} = (v_1(x), \ldots, v_r(x)) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)^r$. Consider the linear equations

$$\mathbf{u}\mathbf{V}_{r\times r}(\mathbf{a}) = \mathbf{v} \bmod (1 + x^{p\tau}). \tag{18}$$

We first review the LU factorization of a Vandermonde matrix given in [31] and then show the LU decoding algorithm for solving $\mathbf{u}$ from the Vandermonde linear system in Eq. (18).

Theorem 13. *[31] For a positive integer r, the $r \times r$ Vandermonde matrix $\mathbf{V}_{r\times r}(\mathbf{a})$ in Eq. (17) can be factorized into*

$$\mathbf{V}_{r\times r}(\mathbf{a}) = \mathbf{L}_r^{(1)}\mathbf{L}_r^{(2)}\ldots\mathbf{L}_r^{(r-1)}\mathbf{U}_r^{(r-1)}\mathbf{U}_r^{(r-2)}\ldots\mathbf{U}_r^{(1)}$$

where $\mathbf{U}_r^{(\ell)}$ is the upper triangular matrix

$$\mathbf{U}_r^{(\ell)} = \left[\begin{array}{c|cccccc} \mathbf{I}_{r-\ell-1} & \multicolumn{6}{c}{\mathbf{0}} \\ \hline & 1 & x^{a_1} & 0 & \cdots & 0 & 0 \\ & 0 & 1 & x^{a_2} & \cdots & 0 & 0 \\ \mathbf{0} & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ & 0 & 0 & 0 & \cdots & 1 & x^{a_l} \\ & 0 & 0 & 0 & \cdots & 0 & 1 \end{array}\right]$$

and $\mathbf{L}_r^{(\ell)}$ is the lower triangular matrix

$$\left[\begin{array}{c|ccc} \mathbf{I}_{r-\ell-1} & \multicolumn{3}{c}{\mathbf{0}} \\ \hline & 1 & 0 & \cdots & 0 \\ & 1 & x^{a_{r-\ell+1}} + x^{a_{r-\ell}} & \cdots & 0 \\ \mathbf{0} & \vdots & \vdots & \ddots & \vdots \\ & 0 & 0 & \cdots & x^{a_r} + x^{a_{r-\ell}} \end{array}\right]$$

for $\ell = 1, 2, \ldots, r-1$.

When $r = 3$, the 3×3 Vandermonde matrix $\mathbf{V}_{3\times3}(a_1, a_2, a_3)$ can be factorized as

$$\begin{aligned} &\mathbf{L}_3^{(1)}\mathbf{L}_3^{(2)}\mathbf{U}_3^{(2)}\mathbf{U}_3^{(1)} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & x^{a_3} + x^{a_2} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 1 & x^{a_2} + x^{a_1} & 0 \\ 0 & 1 & x^{a_3} + x^{a_1} \end{bmatrix} \cdot \\ &\quad \begin{bmatrix} 1 & x^{a_1} & 0 \\ 0 & 1 & x^{a_2} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & x^{a_1} \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned}$$

With the LU factorization of the Vandermonde matrix in Theorem 13, we can solve the Vandermonde linear system in Eq. (18) by solving the following linear equations

$$\mathbf{u}\mathbf{L}_r^{(1)}\mathbf{L}_r^{(2)}\ldots\mathbf{L}_r^{(r-1)}\mathbf{U}_r^{(r-1)}\mathbf{U}_r^{(r-2)}\ldots\mathbf{U}_r^{(1)} = \mathbf{v}.$$

Algorithm 1 Solving a Vandermonde Linear System

Input: positive integer r, prime number p, integers $a_1, a_2, \ldots, a_r$, and $\mathbf{v} = (v_1(x), v_2(x), \ldots, v_r(x)) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)^r$.
Output: $\mathbf{u} = (u_1(x), u_2(x), \ldots, u_r(x)) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)^r$ that satisfies Eq. (18).
Require: $x^{a_{i_1}} + x^{a_{i_2}}$ is relatively prime to $h(x)$ over $\mathbb{F}_q[x]$ for all $0 \le i_1 \le i_2 \le k + r - 1$.

1: $\mathbf{u} \leftarrow \mathbf{v}$
2: **for** i from 1 to $r-1$ **do**
3: **for** j from $r-i+1$ to r **do**
4: $u_j(x) \leftarrow u_j(x) + u_{j-1}(x)x^{a_{i+j-r}}$
5: **for** i from $r-1$ down to 1 **do**
6: Solve $g(x)$ from $(x^{a_r} + x^{a_{r-i}})g(x) = u_r(x)$ by Lemma 11 or Lemma 12
7: $u_r(x) \leftarrow g(x)$
8: **for** j from $r-1$ down to $r-i+1$ **do**
9: Solve $g(x)$ from $(x^{a_j} + x^{a_{r-i}})g(x) = (u_j(x) + u_{j+1}(x))$ by Lemma 11 or Lemma 12
10: $u_j(x) \leftarrow g(x)$
11: $u_{r-i}(x) \leftarrow u_{r-i}(x) + u_{r-i+1}(x)$
12: **return** $\mathbf{u} = (u_1(x), ..., u_r(x))$

The decoding algorithm for solving the Vandermonde linear system based on the LU factorization of a Vandermonde matrix is given in Algorithm 1. In Algorithm 1, Steps 2-4 require $r(r-1)/2$ additions and $r(r-1)/2$ multiplications and Steps 5-11 require $r(r-1)/2$ backward additions and $r(r-1)/2$ divisions by factors of the form $x^{a_j} + x^{a_{r-i}}$.

Example 4. *Continue from Example 1, where $\tau = p = 3$, $k = 6$ and $r = 3$. We have six information polynomials*

$$\begin{aligned} s_0(x) &= 1 + x + x^3 + x^4, \\ s_1(x) &= x + x^2 + x^4 + x^5, \\ s_2(x) &= x + x^4, \\ s_3(x) &= 1 + x^2 + x^3 + x^5, \\ s_4(x) &= x + x^2 + x^7 + x^8, \\ s_5(x) &= x + x^7, \end{aligned}$$

where each polynomial is in $\mathcal{C}_{3\cdot3}(1, 3, 2, 2)$. The parity-check matrix of the code is

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 & x^8 \\ 1 & x^2 & x^4 & x^6 & x^8 & x^{10} & x^{12} & x^{14} & x^{16} \end{bmatrix}.$$

Therefore, we can obtain

$$\begin{bmatrix} s_6(x) & s_7(x) & s_8(x) \end{bmatrix} \begin{bmatrix} 1 & x^6 & x^{12} \\ 1 & x^7 & x^{14} \\ 1 & x^8 & x^{16} \end{bmatrix} = \begin{bmatrix} x + x^2 + x^4 + x^8 \\ 1 + x + x^4 + x^5 + x^6 + x^8 \\ 1 + x^5 + x^6 + x^8 \end{bmatrix}^T.$$

According to Theorem 13, the above Vandermonde matrix can be factorized as

$$\begin{bmatrix}1 & x^6 & x^{12}\\ 1 & x^7 & x^{14}\\ 1 & x^8 & x^{16}\end{bmatrix} = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 1 & x^7+x^8\end{bmatrix}\cdot \begin{bmatrix}1 & 0 & 0\\ 1 & x^6+x^7 & 0\\ 0 & 1 & x^6+x^8\end{bmatrix}\cdot\begin{bmatrix}1 & x^6 & 0\\ 0 & 1 & x^7\\ 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & x^6\\ 0 & 0 & 1\end{bmatrix}.$$

By Algorithm 1, we can solve the three parity polynomials as follows. First, we solve the following linear system

$$\begin{bmatrix}s_6'''(x) & s_7'''(x) & s_8'''(x)\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & x^6\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}x+x^2+x^4+x^8\\ 1+x+x^4+x^5+x^6+x^8\\ 1+x^5+x^6+x^8\end{bmatrix}^T$$

to obtain

$$\begin{bmatrix}s_6'''(x)\\ s_7'''(x)\\ s_8'''(x))\end{bmatrix} = \begin{bmatrix}x+x^2+x^4+x^8\\ 1+x+x^4+x^5+x^6+x^8\\ 1+x+x^2+x^3+x^7+x^8\end{bmatrix}.$$

Then, we solve the following linear system

$$\begin{bmatrix}s_6''(x) & s_7''(x) & s_8''(x)\end{bmatrix}\begin{bmatrix}1 & x^6 & 0\\ 0 & 1 & x^7\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}x+x^2+x^4+x^8\\ 1+x+x^4+x^5+x^6+x^8\\ 1+x+x^2+x^3+x^7+x^8\end{bmatrix}^T$$

to obtain

$$\begin{bmatrix}s_6''(x)\\ s_7''(x)\\ s_8''(x))\end{bmatrix} = \begin{bmatrix}x+x^2+x^4+x^8\\ 1+x^4+x^6+x^7\\ 1+x+x^3+x^4+x^5+x^8\end{bmatrix}.$$

Next, we solve the following linear system

$$\begin{bmatrix}s_6'(x) & s_7'(x) & s_8'(x)\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 1 & x^6+x^7 & 0\\ 0 & 1 & x^6+x^8\end{bmatrix} = \begin{bmatrix}x+x^2+x^4+x^8\\ 1+x^4+x^6+x^7\\ 1+x+x^3+x^4+x^5+x^8\end{bmatrix}^T$$

to obtain

$$\begin{bmatrix}s_6'(x)\\ s_7'(x)\\ s_8'(x)\end{bmatrix} = \begin{bmatrix}x^4+x^5+x^7+x^8\\ x+x^2+x^5+x^7\\ x^2+x^3+x^5+x^6\end{bmatrix}.$$

Finally, we solve the following linear system

$$\begin{bmatrix}s_6(x) & s_7(x) & s_8(x)\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 1 & x^7+x^8\end{bmatrix} = \begin{bmatrix}x^4+x^5+x^7+x^8\\ x+x^2+x^5+x^7\\ x^2+x^3+x^5+x^6\end{bmatrix}^T$$

to obtain

$$\begin{bmatrix}s_6(x)\\ s_7(x)\\ s_8(x)\end{bmatrix} = \begin{bmatrix}x^4+x^5+x^7+x^8\\ x+x^2+x^4+x^5\\ x^4+x^7\end{bmatrix}.$$

Table I
EXAMPLE OF $\mathsf{GEBR}(3,3,6,3,2,1)$.

1	0	0	1	0	0	0	0	0
1	1	1	0	1	1	0	1	0
0	1	0	1	1	0	0	1	0
1	0	0	1	0	0	0	0	0
1	1	1	0	0	0	1	1	1
0	1	0	1	0	0	1	1	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	0	1
0	0	0	0	1	0	1	0	0

Table I shows the example of $\mathsf{GEBR}(3,3,6,3,2,1)$.

In the following, we only evaluate the encoding complexity, and we can obtain the decoding complexity similarly. We define the normalized encoding complexity as the ratio of the total number of XORs involved in the encoding procedure to the number of information symbols. In the encoding procedure of $\mathsf{GEBR}(p,\tau,k,r,q,1)$, we first compute τ parity symbols for the first k columns that takes $k\tau(p-2)$ XORs. Then, we compute the multiplication of k polynomials and the $r\times k$ Vandermonde matrix that requires $(k-1)rp\tau$ XORs, and solve the Vandermonde linear system. In solving the $r\times r$ Vandermonde linear system, there are $r(r-1)$ additions that require $r(r-1)p\tau$ XORs, $r(r-1)/2$ divisions that require $(r(r-1)/2)\cdot((3p\tau-\tau-4)/2)$ XORs.[1] Therefore, the normalized encoding complexity of $\mathsf{GEBR}(p,\tau,k,r,q,1)$ is

$$\frac{\frac{1}{4}r(r-1)(7p\tau-\tau-4)+(k-1)rp\tau+k\tau(p-2)}{k(p-1)\tau}.$$

The encoding/decoding method of EBR is given in [2], [16], and the normalized encoding complexity is

$$\frac{(\frac{1}{2}r^2-\frac{5}{2}r+2^r+rk-1)p+\frac{1}{4}r(r-1)(3p-5)+k(p-2)}{k(p-1)},$$

where $k=p-r$.

We give the comparison of EBR and our proposed codes about the encoding complexity in Table II. The results of Table II show that the proposed LU decoding method has less encoding complexity compared with the decoding methods in [2], [16].

Table II
COMPARISON OF ENCODING ALGORITHMS.

p	τ	r	$k=p-r$	EBR	GEBR	Improvement %
5	1	3	2	8.88	8.25	7.0
7	1	4	3	13.22	11.28	14.7
11	1	5	6	14.42	11.48	20.4
17	1	7	10	25.63	15.11	41.0
19	1	8	11	38.69	17.67	54.3
23	1	10	13	100.72	22.88	77.3

IV. GENERALIZED EXPANDED INDEPENDENT PARITY CODES

In this section, we present the construction of generalized expanded independent parity (GEIP) codes. The constructed

[1]Suppose that $\gcd(b,\tau)=1$.

GEIP code is denoted by $\mathsf{GEIP}(p,\tau,k,r,q,g(x))$ with encoding matrix given in Eq. (19).

$$\mathbf{P}_{k\times r}=\begin{bmatrix}1 & 1 & 1 & \cdots & 1\\ 1 & x & x^2 & \cdots & x^{r-1}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 1 & x^{k-1} & x^{2(k-1)} & \cdots & x^{(r-1)(k-1)}\end{bmatrix}. \quad (19)$$

As the first k polynomials are in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$, the computed r polynomials are also in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$. Note that the EIP code proposed in [2] is a special case as $\mathsf{GEIP}(p,\tau=1,k,r,q,g(x))$.

Table III
EXAMPLE OF $\mathsf{GEIP}(p=3,\tau=3,k=3,r=2,q,g(x)=1)$, WHERE $s_{i,j}=s_{i-6,j}+s_{i-3,j}$ FOR $i=6,7,8$ AND $j=0,1,2$.

0	1	2	3	4
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,0}+s_{0,1}+s_{0,2}$	$s_{0,0}+s_{8,1}+s_{7,2}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,0}+s_{1,1}+s_{1,2}$	$s_{1,0}+s_{0,1}+s_{8,2}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,0}+s_{2,1}+s_{2,2}$	$s_{2,0}+s_{1,1}+s_{0,2}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,0}+s_{3,1}+s_{3,2}$	$s_{3,0}+s_{2,1}+s_{1,2}$
$s_{4,0}$	$s_{4,1}$	$s_{4,2}$	$s_{4,0}+s_{4,1}+s_{4,2}$	$s_{4,0}+s_{3,1}+s_{2,2}$
$s_{5,0}$	$s_{5,1}$	$s_{5,2}$	$s_{5,0}+s_{5,1}+s_{5,2}$	$s_{5,0}+s_{4,1}+s_{3,2}$
$s_{6,0}$	$s_{6,1}$	$s_{6,2}$	$s_{6,0}+s_{6,1}+s_{6,2}$	$s_{6,0}+s_{5,1}+s_{4,2}$
$s_{7,0}$	$s_{7,1}$	$s_{7,2}$	$s_{7,0}+s_{7,1}+s_{7,2}$	$s_{7,0}+s_{6,1}+s_{5,2}$
$s_{8,0}$	$s_{8,1}$	$s_{8,2}$	$s_{8,0}+s_{8,1}+s_{8,2}$	$s_{8,0}+s_{7,1}+s_{6,2}$

Example 5. *Consider $\alpha=6,p=\tau=k=3,r=2$ and $g(x)=1$. The 18 information symbols are $s_{i,j}\in\mathbb{F}_q$ for $i=0,1,\ldots,5$ and $j=0,1,2$. The encoding matrix of $\mathsf{GEIP}(p=3,\tau=3,k=3,r=2,q,g(x)=1)$ is*

$$\mathbf{P}_{3\times 2}=\begin{bmatrix}1 & 1\\ 1 & x\\ 1 & x^2\end{bmatrix}.$$

Example 5 is illustrated in Table III, where $s_{i,j}=s_{i-6,j}+s_{i-3,j}$ for $i=6,7,8$ and $j=0,1,2$.

A. The (n,k) Recoverable Property

The codes $\mathsf{GEIP}(p,\tau,k,r,q,g(x))$ are (n,k) recoverable, if the determinant of any square sub-matrix of $\mathbf{P}_{k\times r}$ in Eq. (19) is invertible in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$. Recall that $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ is isomorphic to $\mathbb{F}_q[x]/(h(x))$ by Lemma 2, the (n,k) recoverable condition is reduced to that the determinant of any square sub-matrix of $\mathbf{P}_{k\times r}$ in Eq. (19), after reducing modulo $h(x)$, is invertible in $\mathbb{F}_q[x]/(h(x))$.

Theorem 14. *Let p be a prime number such that 2 is a primitive element in $\mathbb{Z}_p$ and τ be a power of p. If $(p-1)\tau$ is larger than*

$$\frac{1}{4}kr\min(k,r)-\frac{1}{12}(\min(k,r))^3-\frac{9}{4}\max(k,r)+\frac{25}{12}\min(k,r)-4, \quad (20)$$

then the codes $\mathsf{GEIP}(p,\tau,k,r,2,g(x)=1)$ are (n,k) recoverable for $r\geq 9$.

Proof: When 2 is a primitive element in $\mathbb{Z}_p$ and τ be a multiple of p, then $h(x)=1+x^\tau+x^{2\tau}+\ldots+x^{(p-1)\tau}$ is an irreducible polynomial [30]. If a polynomial whose degree is less than $(p-1)\tau$, then the polynomial is relatively prime to $h(x)$. It is sufficient to show that the maximum degree of the determinants or all the factors of the determinants of all square sub-matrix is less than $(p-1)\tau$. It is shown by Theorem 4 in [32] that the maximum degree of the determinants or all the factors of the determinants is equal to the value on the left side in Eq. (20) when $r\geq 9$. Therefore, $\mathsf{GEIP}(p,\tau,k,r,2,g(x)=1)$ are (n,k) recoverable for $r\geq 9$, if Eq. (20) holds. ■

If τ is a power of 2, then

$$1+x^\tau+x^{2\tau}+\ldots+x^{(p-1)\tau}=(1+x+x^2+\ldots+x^{p-1})^\tau.$$

Recall that, since 2 is a primitive element in $\mathbb{Z}_p$, $1+x+x^2+\ldots+x^{p-1}$ is irreducible over $\mathbb{F}_2$. It is sufficient to show that the maximum degree of the determinants or all the factors of the determinants of all square sub-matrix is less than $p-1$, and we can obtain the following theorem with a proof similar to that of Theorem 14.

Theorem 15. *If 2 is a primitive element in $\mathbb{Z}_p$, τ is a power of 2 and $p-1$ is large than the value in Eq.* (20)*, then the codes $\mathsf{GEIP}(p,\tau,k,r,2,g(x)=1)$ are (n,k) recoverable for $r\geq 9$.*

When $r\leq 3$, the determinant of any square sub-matrix can be written as a multiplication of factors $1+x^i$, where $i\in\{1,2,\ldots,k-1\}$. Therefore, $\mathsf{GEIP}(p,\tau,k,r,q,g(x)=1)$ are (n,k) recoverable for $r\leq 3$, if $1+x^i$ and $h(x)$ are relatively prime. When $4\leq r\leq 8$, we can list the prime numbers p for which $\mathsf{GEIP}(p,\tau,k,r,q,g(x)=1)$ are (n,k) recoverable with similar proof of the MDS condition in [27].

Note that EIP codes share the same (n,k) recoverable condition as IP codes (also called generalized EVENODD codes [4], [11] or Blaum-Bruck-Vardy (BBV) codes [32] in the literature). By letting τ be a power of p, our $\mathsf{GEIP}(p,\tau,k,r,q,g(x)=1)$ codes not only support much more parameters, but the codes may be (n,k) recoverable for some parameters with $p<k$, compared with EIP codes. Example 5 is an (n,k) recoverable property code, as $1+x^i$ is relatively prime to $1+x^3+x^6$ for $i=1,2,3$.

B. Encoding/Decoding Procedure

The encoding procedure of $\mathsf{GEIP}(p,\tau,k,r,q,g(x))$ is as follows. Given $k\alpha$ information symbols $s_{i,j}$ with $i=0,1,\ldots,\alpha-1$ and $j=0,1,\ldots,k-1$, we obtain $(m-\alpha)k$ parity symbols $s_{i,j}$ with $i=\alpha,\alpha+1,\ldots,m-1$ and $j=0,1,\ldots,k-1$ by systematically encoding such that $s_j(x)\in\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ for $j=0,1,\ldots,k-1$. Note that when $g(x)=1$, we can do the systematic encoding by Eq. (4); when $g(x)\neq 1$, the systematic encoding is similar to that of cyclic codes [33, Ch. 7.8]. Then, we compute the last r polynomials by Eq. (2) with encoding matrix $\mathbf{P}_{k\times r}$ in Eq. (19).

Next, we consider the encoding complexity of $\mathsf{GEIP}(p,\tau,k,r,q,g(x)=1)$. First, we compute $s_{(p-1)\tau+\mu,j}=\sum_{\ell=0}^{p-2}s_{\ell\tau+\mu,j}$ for $j=0,1,\ldots,k-1$ and $\mu=0,1,\ldots,\tau-1$, which takes $k\tau(p-2)$ XORs. Then, we compute the r parity polynomials by Eq. (2) with encoding matrix $\mathbf{P}_{k\times r}$ in Eq. (19), which takes $r(k-1)p\tau$ XORs. The encoding complexity is $k\tau(p-2)+r(k-1)p\tau$ XORs. Recall that the encoding complexity of EIP code with $g(x)=1$ is $k(p-2)+r(k-1)p$ XORs. Therefore, the normalized

encoding complexity of $\mathsf{GEIP}(p,\tau,k,r,q,g(x)=1)$ is equal to that of EIP code with $g(x)=1$.

Suppose that r information columns have failed and up to $d-1$ symbols or a burst of up to $\tau+\deg(g(x))$ symbols in each of other k columns have failed. We can first recover up to $d-1$ erased symbols in each of the k columns or a burst of up to $\tau+\deg(g(x))$ erased symbols. Suppose that $\mathsf{GEIP}(p,\tau,k,r,q,g(x))$ is (n,k) recoverable. Then, we can recover r failed columns by first subtracting the other $k-r$ non-failed information polynomials from each of r parity polynomials and then solving the $r\times r$ Vandermonde linear equations by applying the LU decoding method in Algorithm 1. Note that if some of the erased columns are among the r parity columns, we can not formulate the $r\times r$ Vandermonde linear equations and the LU decoding method is not applicable.

Example 6. *Suppose that columns 0 and 1 in Example 5 have failed. We can obtain three polynomials*

$$s_j(x)=s_{0,j}+s_{1,j}x+\cdots+s_{8,j}x^8,$$

for $j=2,3,4$. By subtracting $s_2(x)$ from each of $s_3(x)$ and $s_4(x)$, we have

$$\begin{bmatrix}s_3(x)-s_2(x) & s_4(x)-x^2s_2(x)\end{bmatrix}=\begin{bmatrix}s_0(x) & s_1(x)\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & x\end{bmatrix}.$$

Therefore, we can solve $s_0(x)$ as

$$s_0(x)=\frac{s_4(x)-x^2s_2(x)-x(s_3(x)-s_2(x))}{1-x},$$

by Lemma 11 and $s_1(x)=s_3(x)-s_2(x)-s_0(x)$.

V. Minimum Symbol Distance

In this section, we consider the symbol distance of the proposed array codes, which is the number of symbols in which two codewords differ. Minimum symbol distance is a measure of the maximum number of failed symbols that the codes can tolerate.

Theorem 16. *Let D be the minimum symbol distance of an (n,k) recoverable property array codes constructed by the coding method in Section II. Then, we have $D\geq d(r+1)$.*

Proof: The proof is similar to the one of Lemma 28 in [2]. For completeness, we present the proof. Since the code is (n,k) recoverable, there are at least $r+1$ non-zero columns, and since each non-zero column has weight at least d, we obtain $D\geq d(r+1)$. ∎

We first consider the (n,k) recoverable property array codes with each entry of the encoding matrix $\mathbf{P}_{k\times r}$ being a power of x.

Theorem 17. *If each entry of $\mathbf{P}_{k\times r}$ is a power of x, then $D=d(r+1)$.*

Proof: Let $k-1$ out of the k data polynomials be zero and the remaining data polynomial be a non-zero polynomial in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ with weight d. Note that the multiplication of x^i and a polynomial can be implemented by cyclic-shifting i positions of the polynomial. By encoding the k data polynomials, we have that the obtained r parity polynomials are all in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$ with weight d. Therefore, we obtain a code with symbol distance being $d(r+1)$ and we can obtain the result by Theorem 16. ∎

By Theorem 17, the minimum symbol distance of $\mathsf{GEIP}(p,\tau,k,r,q,g(x))$ is $d(r+1)$. Next, we consider the minimum symbol distance of $\mathsf{GEBR}(p,\tau,k,r,q,1)$.

Lemma 18. *Let $s(x)=x^i(1+x^{j\tau})\in\mathcal{C}_{p\tau}(1,\tau,q,2)$ with weight 2, where i,j are integers with $0\leq i<p\tau$ and $1\leq j<p\neq 2$. If $c(x)\in\mathcal{C}_{p\tau}(1,\tau,q,d)$ such that $(1+x^a)c(x)=(1+x^b)s(x) \bmod 1+x^{p\tau}$, where $0<a,b<\tau$ and $a\neq b$, then the weight of $c(x)$ is larger than 2.*

Proof: Note that $d>1$ because $(1+x^\tau)|f(x)$, where $f(x)$ is the generator polynomial of $\mathcal{C}_{p\tau}(1,\tau,q,d)$. We assume that the weight of $c(x)$ is 2, we can always obtain a contradiction as follows and therefore the weight of $c(x)$ is larger than 2. Let $c(x)=x^\ell(1+x^c)$ with $0\leq\ell<p\tau$. Since $c(x)\in\mathcal{C}_{p\tau}(1,\tau,q,d)$, $c\neq 0$ is a multiple of τ. From $(1+x^a)c(x)=(1+x^b)s(x)$, we have

$$x^{\ell-i}(1+x^a+x^c+x^{a+c})=1+x^b+x^{j\tau}+x^{b+j\tau}.$$

Let $e=\ell-i$. Then $0\leq e<p\tau$. When $e=0$, we have $\{0,a,c,a+c\}=\{0,b,j\tau,b+j\tau\} \bmod p\tau$. By the assumption, $a\neq b$, $a\neq 0$ and $a\neq j\tau$. Thus, $a=b+j\tau \bmod p\tau$. Since $b<\tau$ and $j<p$, we have $a=b+j\tau$ which is impossible due to $a<\tau$ and $j\geq 1$.

Next, we consider that $e\neq 0$ and we have $\{e,e+a,e+c,e+a+c\}=\{0,b,j\tau,b+j\tau\} \bmod p\tau$. Note that $\ell+c<p\tau$. Since $c\neq 0$ is a multiple of τ, we have $\ell<(p-1)\tau$. Assume that $e=0 \bmod p\tau$. This is impossible since $e<p\tau$. Assume that $e+a=0 \bmod p\tau$, i.e., $\ell-i+a=p\tau$. That is, $a-i=p\tau-\ell>\tau$ which contradicts to the fact $a<\tau$. Assume that $e+c=0 \bmod p\tau$, i.e., $\ell-i+c=p\tau$. Since $\ell+c<p\tau$, it is impossible. Finally, assume that $e+a+c=0 \bmod p\tau$. We have $\ell-i+a+c=p\tau$ due to $\ell+c<p\tau$ and $a-i<\tau$. Since $b<j\tau<b+j\tau$ and and $e<e+a<e+c$, $e=b<\tau$, $e+a=j\tau<2\tau$, and $e+c=b+j\tau$. Hence, $j=1$ and $c=\tau$. Therefore, $a+b=p\tau-\tau=(p-1)\tau$ and $e+a=b+a=j\tau=\tau$. Since $p>2$, we have a contradiction. Therefore, the weight of $c(x)$ is larger than 2. ∎

Theorem 19. *Suppose the codes $\mathsf{GEBR}(p,\tau,k,r,q,1)$ are (n,k) recoverable. When $r=2$ and $\tau\leq\lfloor\frac{k+1}{2}\rfloor$, the minimum symbol distance of $\mathsf{GEBR}(p,\tau,k,r,q,1)$ is $2(r+1)=6$. When $r=2$ and $\tau>k+1$, the minimum symbol distance of $\mathsf{GEBR}(p,\tau,k,r,q,1)$ is 8. When $r=3$, and $\tau\leq\lfloor\frac{k+2}{3}\rfloor$, the minimum symbol distance of $\mathsf{GEBR}(p,\tau,k,r,q,1)$ is $2(r+1)=8$.*

Proof: By Theorem 16, if we can find a codeword composed of $r+1$ non-zero polynomials each with weight 2 and $k-1$ zero polynomials, then the minimum symbol distance is $2(r+1)$.

When $r=2$, by Theorem 16, each non-zero codeword contains at least three non-zero polynomials. Without loss of generality, suppose that the three non-zero polynomials are $s_\alpha(x),s_\beta(x),s_\gamma(x)$ and the other $k-1$ polynomials are zero, where $0\leq\alpha<\beta<\gamma\leq k+1$. Suppose that the weight of

$s_\alpha(x)$ is 2. According to Eq. (5), we obtain that

$$\begin{bmatrix} s_\alpha(x) \\ x^\alpha s_\alpha(x) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ x^\beta & x^\gamma \end{bmatrix} \cdot \begin{bmatrix} s_\beta(x) \\ s_\gamma(x) \end{bmatrix}.$$

Therefore, we can compute that $(x^\beta + x^\gamma)s_\gamma(x) = (x^\alpha + x^\beta)s_\alpha(x)$ and $(x^\beta + x^\gamma)s_\beta(x) = (x^\alpha + x^\gamma)s_\alpha(x)$. If $\lfloor \frac{k+1}{2} \rfloor \geq \tau$, we have $k+1 \geq 2\tau$. Let $(\alpha, \beta, \gamma) = (0, \tau, 2\tau)$ and $s_0(x) = 1 + x^\tau \in \mathcal{C}_{p\tau}(1, \tau, q, 2)$ with weight 2, then we can obtain $s_\tau(x) = x^\tau + x^{p\tau-\tau}$ and $s_{2\tau}(x) = 1 + x^{p\tau-\tau}$, which are both with weight 2. Therefore, the minimum symbol distance of $\mathsf{GEBR}(p, \lfloor \frac{k+1}{2} \rfloor \geq \tau, k, r = 2, q, 1)$ is $2(r+1) = 6$.

If $k + 1 < \tau$, we have $0 < \gamma - \alpha \leq k + 1 < \tau$ and $0 < \gamma - \beta \leq k < \tau$. Suppose that the weight of $s_\alpha(x)$ is 2, by Lemma 18, the weight of s_β is larger than 2. Since $s_\beta(x) \in \mathcal{C}_{p\tau}(1, \tau, q, 2)$ (the number of non-zero coefficients of $s_\beta(x)$ is an even number by Lemma 4) and the weight of $s_\beta(x)$ is larger than 2, the weight of $s_\beta(x)$ is no less than 4. Consider the polynomial $s_\gamma(x)$. Since the weight of $s_\gamma(x)$ is at least 2, the minimum symbol distance of $\mathsf{GEBR}(p, \tau > k + 1, k, r = 2, q, 1)$ is at least 8. Let $(\alpha, \beta, \gamma) = (0, \beta, 2\beta)$ and $s_0 = 1 + x^\tau$, where $\beta \leq (k+1)/2 < \tau$. Then we obtain $s_{2\beta}(x) = x^{p\tau-\beta} + x^{(p\tau-\beta+\tau) \bmod p\tau}$, which has weight 2. Since $1 + x^{2\beta} = (1 + x^\beta)^2$, we can obtain $s_\beta(x) = x^{p\tau-\beta}(1 + x^\tau)(1 + x^\beta)$, i.e., $s_\beta(x) = 1 + x^\tau + x^{p\tau-\beta} + x^{(p\tau-\beta+\tau) \bmod p\tau}$, which is with weight 4. We can thus obtain that the minimum symbol distance of $\mathsf{GEBR}(p, \tau > k + 1, k, r = 2, q, 1)$ is 8.

When $r = 3$, by Theorem 16, we have at least four non-zero polynomials. Without loss of generality, suppose that the four non-zero polynomials are $s_\alpha(x), s_\beta(x), s_\gamma(x), s_\eta(x)$ and the other $k - 1$ polynomials are zero, where $0 \leq \alpha < \beta < \gamma < \eta \leq k+2$. We assume that the weight of $s_\alpha(x)$ is 2. By Eq. (5), we have

$$\begin{bmatrix} s_\alpha(x) \\ x^\alpha s_\alpha(x) \\ x^{2\alpha} s_\alpha(x) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ x^\beta & x^\gamma & x^\eta \\ x^{2\beta} & x^{2\gamma} & x^{2\eta} \end{bmatrix} \cdot \begin{bmatrix} s_\beta(x) \\ s_\gamma(x) \\ s_\eta(x) \end{bmatrix}.$$

Since $\lfloor \frac{k+2}{3} \rfloor \geq \tau$, we have $k + 2 \geq 3\tau$. Let $(\alpha, \beta, \gamma, \eta) = (0, \tau, 2\tau, 3\tau)$ and $s_0(x) = 1 + x^\tau$, then we can compute $s_\tau(x), s_{2\tau}(x), s_{3\tau}(x)$ as follows,

$$\begin{bmatrix} s_\tau(x) \\ s_{2\tau}(x) \\ s_{3\tau}(x) \end{bmatrix} = \begin{bmatrix} x^\tau + x^{p\tau-2\tau} \\ 1 + x^{p\tau-3\tau} \\ x^{p\tau-3\tau} + x^{p\tau-2\tau} \end{bmatrix},$$

which have all weight 2. Therefore, the minimum symbol distance is $2(r+1) = 8$ when $\lfloor \frac{k+2}{3} \rfloor \geq \tau$ and $r = 3$. ∎

By Theorem 19, the minimum symbol distance of $\mathsf{GEBR}(p, \tau > k + 1, k, r = 2, q, 1)$ is larger than that of $\mathsf{GEBR}(p, \tau = 1, k, r = 2, q, 1)$, i.e., EBR codes with $r = 2$.

Theorem 20. *Suppose the codes $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ are (n, k) recoverable. When $r = 4$, $g(x) = 1$ and $\tau \leq \lfloor \frac{k+3}{4} \rfloor$, the minimum symbol distance of $\mathsf{GEBR}(p, \tau, k, 4, q, 1)$ is no larger than 12.*

Proof: It is sufficient to find a codeword such that the symbol distance is 12 when $r = 4$. As the code $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ is (n, k) recoverable, each codeword contains at least $r + 1$ non-zero polynomials that are in $\mathcal{C}_{p\tau}(1, \tau, q, d)$, with the weight d of each non-zero polynomial being a multiple of 2 by Lemma 4.

Consider $r = 4$. Without loss of generality, suppose that the five non-zero polynomials are $s_\alpha(x), s_\beta(x), s_\gamma(x), s_\delta(x), s_\eta(x)$ and the other $k - 1$ polynomials are zero, where $0 \leq \alpha < \beta < \gamma < \delta < \eta \leq k + 3$. Suppose that the weight of $s_\alpha(x)$ is 2. According to Eq. (5), we have

$$\begin{aligned} &\begin{bmatrix} s_\alpha(x) & x^\alpha s_\alpha(x) & x^{2\alpha} s_\alpha(x) & x^{3\alpha} s_\alpha(x) \end{bmatrix} \\ &= \begin{bmatrix} s_\beta(x) & s_\gamma(x) & s_\delta(x) & s_\eta(x) \end{bmatrix} \cdot \begin{bmatrix} 1 & x^\beta & x^{2\beta} & x^{3\beta} \\ 1 & x^\gamma & x^{2\gamma} & x^{3\gamma} \\ 1 & x^\delta & x^{2\delta} & x^{3\delta} \\ 1 & x^\eta & x^{2\eta} & x^{3\eta} \end{bmatrix}. \end{aligned}$$

Since $\tau \leq \lfloor \frac{k+3}{4} \rfloor$, we have $4\tau \leq k + 3$. Let $(\alpha, \beta, \gamma, \delta, \eta) = (0, \tau, 2\tau, 3\tau, 4\tau)$ and $s_0(x) = 1 + x^\tau$, then we can take

$$\begin{aligned} s_\tau(x) =& x^\tau + x^{(p-3)\tau}, \\ s_{2\tau}(x) =& x^{(p-5)\tau} + x^{(p-3)\tau} + x^{(p-2)\tau} + 1, \\ s_{3\tau}(x) =& x^{(p-6)\tau} + x^{(p-2)\tau}, \\ s_{4\tau}(x) =& x^{(p-6)\tau} + x^{(p-5)\tau}. \end{aligned}$$

Therefore, the minimum symbol distance is no larger than 12 when $r = 4$ and $\tau \leq \lfloor \frac{k+3}{4} \rfloor$. ∎

Lemma 30 in [2] is a special case of Theorem 19 with $\tau = 1$ and $2 \leq r \leq 3$. To determine the minimum symbol distance of other parameters is an open problem.

VI. Recovery of Erased Lines of Slope i in $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ Codes

In this section, we assume $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ are (n, k) recoverable and want to show that $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ can recover some erased lines of slope i with $i = 0, 1, \ldots, r-1$, under some constraint. We first consider the code $\mathsf{GEBR}(p, 1, k, r, q, 1)$ with $\tau = 1$, that is, an EBR code. Recall that the $k + r$ symbols in line ℓ of slope i of the $m \times (k + r)$ codeword array are $s_{\ell,0}, s_{\ell-i,1}, s_{\ell-2i,2}, \cdots, s_{\ell-(k+r-1)i,k+r-1}$ for $\ell = 0, 1, \ldots, m - 1$ and $i = 0, 1, \ldots, r - 1$.

Theorem 21. *The code $\mathsf{GEBR}(p, 1, k, r, q, 1)$ can recover any r erased lines $e_1, e_2, \ldots, e_r$ of slope i for $0 \leq i \leq r - 1$, if and only if the following matrix*

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x^{e_1} & x^{e_2} & \cdots & x^{e_r} \\ x^{e_1 \cdot 2^{-1}} & x^{e_2 \cdot 2^{-1}} & \cdots & x^{e_r \cdot 2^{-1}} \\ \vdots & \vdots & \ddots & \vdots \\ x^{e_1 \cdot (r-i-1)^{-1}} & x^{e_2 \cdot (r-i-1)^{-1}} & \cdots & x^{e_r \cdot (r-i-1)^{-1}} \\ x^{e_1 \cdot (p-1)^{-1}} & x^{e_2 \cdot (p-1)^{-1}} & \cdots & x^{e_r \cdot (p-1)^{-1}} \\ x^{e_1 \cdot (p-2)^{-1}} & x^{e_2 \cdot (p-2)^{-1}} & \cdots & x^{e_r \cdot (p-2)^{-1}} \\ \vdots & \vdots & \ddots & \vdots \\ x^{e_1 \cdot (p-i)^{-1}} & x^{e_2 \cdot (p-i)^{-1}} & \cdots & x^{e_r \cdot (p-i)^{-1}} \end{bmatrix}$$

is invertible over $\mathbb{F}_q[x]/(1 + x + \ldots + x^{p-1})$, where $0 \leq e_1 < e_2 < \cdots < e_r \leq p - 1$ and $k + r \leq p$. Note that for any integer ℓ with $1 \leq \ell \leq p - 1$, ℓ^{-1} is the inverse of ℓ in $\mathbb{Z}_p$, i.e., $\ell^{-1}\ell = 1 \bmod p$.

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2022.3202140

Proof: When $\tau = 1$, we have $p \geq k + r$. Suppose that r lines $e_1, e_2, \ldots, e_r$ of slope 0 are erased, where $0 \leq e_1 < e_2 < \cdots < e_r \leq p-1$. For $\ell = 0, 1, \ldots, p-1$, we represent $k+r$ symbols $s_{\ell,0}, s_{\ell,1}, \ldots, s_{\ell,k+r-1}$ by the polynomial

$$\bar{s}_\ell(x) = s_{\ell,0} + s_{\ell,1}x + \ldots + s_{\ell,k+r-1}x^{k+r-1}$$

over $\mathbb{F}_q[x]/(1+x^p)$. As

$$s_{\ell,0} + s_{\ell,1} + \ldots + s_{\ell,k+r-1} = 0,$$

by Lemma 4, we have[2]

$$\bar{s}_\ell(x) \in \mathcal{C}_p(1, 1, q, d).$$

$\mathcal{C}_p(1,1,q,d)$ is an ideal of $\mathbb{F}_q[x]/(1+x^p)$ and is isomorphic to $\mathbb{F}_q[x]/(1+x+\ldots+x^{p-1})$ by Lemma 2. For $j = 0, 1, \ldots, k+r-1$, the summation of the p symbols in column j is zero, we can thus compute

$$s_{e_1,j} + s_{e_2,j} + \cdots + s_{e_r,j} = \sum_{\ell=1,\ell\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} s_{\ell+e_1,j},$$

i.e.,

$$\bar{s}_{e_1}(x)+\bar{s}_{e_2}(x)+\cdots+\bar{s}_{e_r}(x) = \sum_{\ell=1,\ell\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} \bar{s}_{\ell+e_1}(x). \tag{21}$$

According to row ℓ of Eq. (5), we have

$$\begin{aligned}
&0 = s_0(x) + x^\ell s_1(x) + \cdots + x^{(p-1)\ell}s_{p-1}(x)\\
&= (s_{0,0} + s_{1,0}x + \cdots + s_{p-1,0}x^{p-1})+\\
&x^\ell(s_{0,1} + s_{1,1}x + \cdots + s_{p-1,1}x^{p-1})+\\
&\cdots + x^{(p-1)\ell}(s_{0,p-1} + s_{1,p-1}x + \cdots + s_{p-1,p-1}x^{p-1})\\
&= (s_{0,0} + s_{0,1}x^\ell + \cdots + s_{0,p-1}x^{(p-1)\ell})+\\
&x(s_{1,0} + s_{1,1}x^\ell + \cdots + s_{1,p-1}x^{(p-1)\ell})+\\
&\cdots + x^{p-1}(s_{p-1,0} + s_{p-1,1}x^\ell + \cdots + s_{p-1,p-1}x^{(p-1)\ell})\\
&= (s_{0,0} + s_{1,p-\ell^{-1}} + \cdots + s_{p-1,p-(p-1)\ell^{-1}})+\\
&x^\ell(s_{0,1} + s_{1,p-\ell^{-1}+1} + \cdots + s_{p-1,p-(p-1)\ell^{-1}+1}) + \cdots +\\
&x^{(p-1)\ell}(s_{0,p-1} + s_{1,p-\ell^{-1}+p-1} + \cdots + s_{p-1,p-(p-1)\ell^{-1}+p-1}),
\end{aligned}$$

where $\ell^{-1}\ell = 1 \bmod p$. Note that $\ell \cdot i \neq \ell \cdot j \bmod p$ for $i \neq j \in \{0, 1, \ldots, p-1\}$ and $\{0, \ell, \cdots, (p-1)\ell\} = \{0, 1, \cdots, p-1\} \bmod p$, we have

$$s_{0,j} + s_{1,p-\ell^{-1}+j} + \cdots + s_{p-1,p-(p-1)\ell^{-1}+j} = 0$$

for $j = 0, 1, \ldots, p-1$. Therefore, we can obtain that

$$\bar{s}_0(x)+x^{\ell^{-1}}\bar{s}_1(x)+x^{2\ell^{-1}}\bar{s}_2(x)+\cdots+x^{(p-1)\ell^{-1}}\bar{s}_{p-1}(x) = 0,$$

where $\ell = 1, 2, \ldots, r-1$. From the above equation, we then can compute

$$\begin{aligned}
&x^{e_1\ell^{-1}}\bar{s}_{e_1}(x) + x^{e_2\ell^{-1}}\bar{s}_{e_2}(x) + \cdots + x^{e_r\ell^{-1}}\bar{s}_{e_r}(x)\\
&= \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)\ell^{-1}}\bar{s}_{u+e_1}(x).
\end{aligned} \tag{22}$$

[2]We can set $s_{\ell,k+r}, s_{\ell,k+r+1}, \ldots, s_{\ell,p-1}$ all zeros.

By Eq. (21) and Eq. (22), we can obtain

$$\begin{aligned}
&\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x^{e_1} & x^{e_2} & \cdots & x^{e_r} \\ \vdots & \vdots & \ddots & \vdots \\ x^{e_1(r-1)^{-1}} & x^{e_2(r-1)^{-1}} & \cdots & x^{e_r(r-1)^{-1}} \end{bmatrix} \begin{bmatrix} \bar{s}_{e_1}(x) \\ \bar{s}_{e_2}(x) \\ \vdots \\ \bar{s}_{e_r}(x) \end{bmatrix}\\
&= \begin{bmatrix} \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} \bar{s}_{u+e_1}(x) \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{u+e_1}\bar{s}_{u+e_1}(x) \\ \vdots \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)(r-1)^{-1}}\bar{s}_{u+e_1}(x) \end{bmatrix}.
\end{aligned} \tag{23}$$

By applying the isomorphism $\theta : \mathcal{C}_p(1,1,q,d) \to \mathbb{F}_q[x]/(1+x+\ldots+x^{p-1})$ in Lemma 2 for the above linear equations, we can show that, if the determinant of the above $r \times r$ matrix is invertible over $\mathbb{F}_q[x]/(1+x+\ldots+x^{p-1})$, then we can compute $\bar{s}_{e_1}(x), \bar{s}_{e_2}(x), \ldots, \bar{s}_{e_r}(x)$ by first solving the linear equations over $\mathbb{F}_q[x]/(1+x+\ldots+x^{p-1})$ and then applying the inverse isomorphism θ^{-1}.

Next, we consider that r lines $e_1, e_2, \ldots, e_r$ of slope i are erased, where $0 \leq e_1 < e_2 < \cdots < e_r \leq p-1$ and $i = 1, 2, \ldots, r-1$. For $\ell = 0, 1, \ldots, p-1$, we represent the $k+r$ symbols $s_{\ell,0}, s_{\ell-i,1}, s_{\ell-2i,2}, \ldots, s_{\ell-i(k+r-1),k+r-1}$ in the line of slope i by the polynomial

$$\bar{s}_\ell(x) = s_{\ell,0} + s_{\ell-i,1}x + \ldots + s_{\ell-i(k+r-1),k+r-1}x^{k+r-1},$$

which is in $\mathcal{C}_p(1,1,q,d)$, as the summation of the $k+r$ symbols is zero. According to row $i+1$ of Eq. (5), we have that the summation of the $k+r$ symbols in the line of slope $i+1$ is zero, i.e.,

$$s_{j,0}+s_{j-(i+1),1}+s_{j-2(i+1),2}+\cdots+s_{j-(k+r-1)(i+1),k+r-1} = 0 \tag{24}$$

for $j = 0, 1, \ldots, p-1$. Note that all the indices are taken modulo p in the proof and $s_{j,k+r}, s_{j,k+r+1}, \cdots, s_{j,p-1}$ are all zero. Then, we can obtain that

$$\begin{aligned}
&\bar{s}_0(x) + x\bar{s}_1(x) + x^2\bar{s}_2(x) + \cdots + x^{p-1}\bar{s}_{p-1}(x)\\
&= \sum_{j=0}^{p-1} s_{-ij,j}x^j + x(\sum_{j=0}^{p-1} s_{1-ij,j}x^j) + \cdots + x^{p-1}(\sum_{j=0}^{p-1} s_{p-1-ij,j}x^j)\\
&= (s_{0,0} + s_{p-1-i,1} + s_{p-2-2i,2} + \cdots + s_{1-(p-1)i,p-1})+\\
&\quad (s_{-i,1} + s_{1,0} + s_{2-(p-1)i,p-1} + \cdots + s_{p-1-2i,2})x\\
&\quad + (s_{-2i,2} + s_{1-i,1} + s_{2,0} + \cdots + s_{p-1-3i,3})x^2 + \cdots +\\
&\quad (s_{-i(p-1),p-1} + s_{1-i(p-2),p-2} + \cdots + s_{p-1,0})x^{p-1}\\
&= 0,
\end{aligned}$$

where the last equation comes from Eq. (24). Similarly, according to row $i+\ell$ of Eq. (5), we can compute that

$$\bar{s}_0(x) + x^{\ell^{-1}}\bar{s}_1(x) + \cdots + x^{(p-1)\ell^{-1}}\bar{s}_{p-1}(x) = 0,$$

where $\ell = 1, 2, \ldots, r-i-1, -1, -2, \ldots, -i$, then we can compute

$$\begin{aligned}
&x^{e_1\ell^{-1}}\bar{s}_{e_1}(x) + x^{e_2\ell^{-1}}\bar{s}_{e_2}(x) + \cdots + x^{e_r\ell^{-1}}\bar{s}_{e_r}(x)\\
&= \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)\ell^{-1}}\bar{s}_{u+e_1}(x).
\end{aligned} \tag{25}$$

Since the summation of the p symbols in each column is zero, we have

$$\bar{s}_0(x) + \bar{s}_1(x) + \ldots + \bar{s}_{p-1}(x) = 0,$$

together with Eq. (25), we can obtain

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x^{e_1} & x^{e_2} & \cdots & x^{e_r} \\ \vdots & \vdots & \ddots & \vdots \\ x^{e_1(r-i-1)^{-1}} & x^{e_2(r-i-1)^{-1}} & \cdots & x^{e_r(r-i-1)^{-1}} \\ x^{e_1(p-1)^{-1}} & x^{e_2(p-1)^{-1}} & \cdots & x^{e_r(p-1)^{-1}} \\ \vdots & \vdots & \ddots & \vdots \\ x^{e_1(p-i)^{-1}} & x^{e_2(p-i)^{-1}} & \cdots & x^{e_r(p-i)^{-1}} \end{bmatrix} \begin{bmatrix} \bar{s}_{e_1}(x) \\ \bar{s}_{e_2}(x) \\ \vdots \\ \bar{s}_{e_r}(x) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} \bar{s}_{u+e_1}(x) \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{u+e_1} \bar{s}_{u+e_1}(x) \\ \vdots \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)(r-i-1)^{-1}} \bar{s}_{u+e_1}(x) \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)(p-1)^{-1}} \bar{s}_{u+e_1}(x) \\ \vdots \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)(p-i)^{-1}} \bar{s}_{u+e_1}(x) \end{bmatrix}, \quad (26)$$

for $i = 1, 2, \ldots, r-2$ and

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x^{e_1(p-1)^{-1}} & x^{e_2(p-1)^{-1}} & \cdots & x^{e_r(p-1)^{-1}} \\ \vdots & \vdots & \ddots & \vdots \\ x^{e_1(p-r+1)^{-1}} & x^{e_2(p-r+1)^{-1}} & \cdots & x^{e_r(p-r+1)^{-1}} \end{bmatrix} \begin{bmatrix} \bar{s}_{e_1}(x) \\ \bar{s}_{e_2}(x) \\ \vdots \\ \bar{s}_{e_r}(x) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} \bar{s}_{u+e_1}(x) \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)(p-1)^{-1}} \bar{s}_{u+e_1}(x) \\ \vdots \\ \sum_{u=1,u\neq e_2-e_1,\ldots,e_r-e_1}^{p-1} x^{(u+e_1)(p-r+1)^{-1}} \bar{s}_{u+e_1}(x) \end{bmatrix}, \quad (27)$$

when $i = r-1$. If the left $r \times r$ matrices in Eq. (26) and Eq. (27) are invertible in $\mathbb{F}_q[x]/(1+x+\cdots+x^{p-1})$, then we can compute the erased r polynomials. Therefore, the necessary and sufficient condition for recovering any r erased lines of slope i is proved. ∎

In Theorem 21, we presented a necessary and sufficient condition for recovering any r erased lines of slope i for $0 \leq i \leq r-1$. We have p distinct slopes in the $p \times (k+r)$ array; however the method in Theorem 21 is not applicable to the slope i with $r \leq i \leq p-1$. The reason is as follows. When $0 \leq i \leq r-1$, we represent the erased $k+r$ symbols in each erased line by a polynomial which is in $\mathcal{C}_p(1,1,q,d)$. According to the parity-check matrix in Eq. (5), we can formulate r linear equations of the r erased polynomials in $\mathcal{C}_p(1,1,q,d)$. If the corresponding $r \times r$ matrix of the r linear equations is invertible in $\mathbb{F}_q[x]/(1+x+\cdots+x^{p-1})$, then we can recover the r erased polynomials, i.e., the r erased lines. When $r \leq i \leq p-1$, the erased r polynomials are in $\mathbb{F}_q[x]/(1+x^p)$, we are not sure whether each erased polynomial is in $\mathcal{C}_p(1,1,q,d)$ or not. Therefore, there are many solutions in solving the r linear equations of the r erased polynomials, even if the $r \times r$ matrix of the r linear equations is invertible in $\mathbb{F}_q[x]/(1+x+\cdots+x^{p-1})$. Finding necessary and sufficient conditions for recovering any r erased lines of slope i for $r \leq i \leq p-1$ is an open problem.

By Theorem 21, when $r = 1, 2, 3$, we can check that the codes $\mathsf{GEBR}(p, 1, k, r, q, 1)$ can recover any r erased lines of slope i for $0 \leq i \leq r-1$, which is also shown by Theorem 40 in [2]. When $r \geq 4$, we need to check that the matrix given in Theorem 21 is invertible over $\mathbb{F}_q[x]/(1+x+\ldots+x^{p-1})$. Note that when the r erased lines $e_1, e_2, \ldots, e_r$ are consecutive integers modulo p, i.e., $e_{i+1} = e_i + 1 \bmod p$ for $i = 1, 2, \ldots, r-1$, then the $r \times r$ matrix is a Vandermonde matrix and is invertible over $\mathbb{F}_q[x]/(1+x+\ldots+x^{p-1})$. We can directly obtain the following corollary from Theorem 21.

Corollary 22. *The codes $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ can recover any r erased lines $e_1, e_2, \ldots, e_r$ with $e_{i+1} = e_i + 1 \bmod p$ of slope i for $0 \leq i \leq r-1$, where $k + r \leq p$.*

Example 7. *Consider the code $\mathsf{GEBR}(p = 11, \tau = 1, k = 7, r = 4, q, 1)$. We have $k + r = 11$ polynomials*

$$s_j(x) = s_{0,j} + s_{1,j}x + s_{2,j}x^2 + \ldots + s_{10,j}x^{10},$$

where $j = 0, 1, \ldots, 10$ and

$$s_{10,j} = s_{0,j} + s_{1,j} + s_{2,j} + \ldots + s_{9,j}.$$

The parity-check matrix of the code is

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 & x^8 & x^9 & x^{10} \\ 1 & x^2 & x^4 & x^6 & x^8 & x^{10} & x^{12} & x^{14} & x^{16} & x^{18} & x^{20} \\ 1 & x^3 & x^6 & x^9 & x^{12} & x^{15} & x^{18} & x^{21} & x^{24} & x^{27} & x^{30} \end{bmatrix}.$$

According to row ℓ of the parity-check matrix, we have

$$s_{i,0} + s_{i-\ell,1} + s_{i-2\ell,2} + \cdots + s_{i-10\ell,10} = 0, \quad (28)$$

where $\ell = 0, 1, 2, 3$ and $i = 0, 1, \ldots, 10$. Note that the indices are operated modulo $p = 11$ in the example. Suppose that the following 44 symbols in four lines of slope 1

$$s_{0,0}, s_{10,1}, s_{9,2}, s_{8,3}, s_{7,4}, s_{6,5}, s_{5,6}, s_{4,7}, s_{3,8}, s_{2,9}, s_{1,10},$$
$$s_{1,0}, s_{0,1}, s_{10,2}, s_{9,3}, s_{8,4}, s_{7,5}, s_{6,6}, s_{5,7}, s_{4,8}, s_{3,9}, s_{2,10},$$
$$s_{2,0}, s_{1,1}, s_{0,2}, s_{10,3}, s_{9,4}, s_{8,5}, s_{7,6}, s_{6,7}, s_{5,8}, s_{4,9}, s_{3,10},$$
$$s_{3,0}, s_{2,1}, s_{1,2}, s_{0,3}, s_{10,4}, s_{9,5}, s_{8,6}, s_{7,7}, s_{6,8}, s_{5,9}, s_{4,10},$$

are erased. For $i = 0, 1, \ldots, 10$, we represent the following 11 symbols

$$s_{i,0}, s_{i-1,1}, s_{i-2,2}, s_{i-3,3}, s_{i-4,4}, s_{i-5,5},$$
$$s_{i-6,6}, s_{i-7,7}, s_{i-8,8}, s_{i-9,9}, s_{i-10,10},$$

in the line of slope 1 by the polynomial

$$\bar{s}_i(x) = s_{i,0} + s_{i-1,1}x + s_{i-2,2}x^2 + \cdots + s_{i-10,10}x^{10}. \quad (29)$$

By row $\ell = 1$ (the second row) of the parity-check matrix, we have that the summation of all the coefficients of $\bar{s}_i(x)$ is zero. Therefore, $\bar{s}_i(x) \in \mathcal{C}_p(1,1,q,d)$. We need to recover four polynomials $\bar{s}_0(x), \bar{s}_1(x), \bar{s}_2(x), \bar{s}_3(x)$ from the other 7 polynomials. By Eq. (28) *with $\ell = 0$, we have*

$$s_{i,0} + s_{i,1} + s_{i,2} + \cdots + s_{i,10} = 0,$$

where $i = 0, 1, \ldots, 10$. Recall that the polynomial $\bar{s}_i(x)$ is given in Eq. (29)*, we have*

$$\bar{s}_0(x) + x^{10}\bar{s}_1(x) + x^9\bar{s}_2(x) + \cdots + x\bar{s}_{10}(x) = 0.$$

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2022.3202140

When $\ell = 2, 3$, with the same argument, we have

$$\bar{s}_0(x) + x\bar{s}_1(x) + x^2\bar{s}_2(x) + \cdots + x^{10}\bar{s}_{10}(x) = 0,$$
$$\bar{s}_0(x) + x^6\bar{s}_1(x) + x^{12}\bar{s}_2(x) + \cdots + x^{60}\bar{s}_{10}(x) = 0.$$

Since the summation of the 11 symbols in each column is zero, we have

$$\bar{s}_0(x) + \bar{s}_1(x) + \bar{s}_2(x) + \cdots + \bar{s}_{10}(x) = 0.$$

Therefore, we obtain

$$\begin{bmatrix} 1 & x^{10} & x^9 & x^8 \\ 1 & 1 & 1 & 1 \\ 1 & x & x^2 & x^3 \\ 1 & x^6 & x^{12} & x^{18} \end{bmatrix} \cdot \begin{bmatrix} \bar{s}_0(x) \\ \bar{s}_1(x) \\ \bar{s}_2(x) \\ \bar{s}_3(x) \end{bmatrix} = \begin{bmatrix} \sum_{i=4}^{10} x^{11-i}\bar{s}_i(x) \\ \sum_{i=4}^{10} \bar{s}_i(x) \\ \sum_{i=4}^{10} x^{i}\bar{s}_i(x) \\ \sum_{i=4}^{10} x^{6i}\bar{s}_i(x) \end{bmatrix}.$$

Since

$$\det \begin{bmatrix} 1 & x^{10} & x^9 & x^8 \\ 1 & 1 & 1 & 1 \\ 1 & x & x^2 & x^3 \\ 1 & x^6 & x^{12} & x^{18} \end{bmatrix} \bmod (1 + x^{11})$$
$$= (1+x)(1+x^6)(1+x^{10})(x+x^6)(x+x^{10})(x^6+x^{10}),$$

which is relatively prime to $1+x+\ldots+x^{10}$. Therefore, we can solve $\bar{s}_0(x), \bar{s}_1(x), \bar{s}_2(x), \bar{s}_3(x)$. Specifically, we can compute the four polynomials by Algorithm 1.

We have checked that the codes $\mathsf{GEBR}(p, \tau, k, r = 4, q, 1)$ can recover any $r = 4$ erased lines e_1, e_2, e_3, e_4 with $0 \leq e_1 < e_2 < e_3 < e_4 \leq p-1$ of slope i for $0 \leq i \leq 3$, when $p = 7, 11, 13, 19$.

We can not employ the above method for the case of $\tau \geq 2$ in general, as the polynomial representing the $k + r$ erased symbols in the line of a slope is not in $\mathcal{C}_{p\tau}(1, \tau, q, d)$. We show in the next theorem that the code $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ can recover up to τ erased lines of a slope, when $\tau \geq 2$. Note that the erased τ lines are not arbitrary.

Theorem 23. *The code $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ can recover any τ erased lines $e_1, e_2, \ldots, e_\tau$ of slope i for $i = 0, 1, \ldots, r-1$, where $\tau \nmid (e_\alpha - e_\beta)$ for $\alpha \neq \beta \in \{1, 2, \ldots, \tau\}$.*

Proof: Suppose that τ lines $e_1, e_2, \ldots, e_\tau$ of slope i are erased, where $\tau \nmid (e_\alpha - e_\beta)$ for $\alpha \neq \beta \in \{1, 2, \ldots, \tau\}$. For $\ell = e_1, e_2, \ldots, e_\tau$, the erased $k+r$ symbols in the line of slope i are $s_{\ell,0}, s_{\ell-i,1}, s_{\ell-2i,2}, \ldots, s_{\ell-(k+r-1)i,k+r-1}$. By Lemma 4, we have

$$s_{\ell,j} = \sum_{\mu=1}^{p-1} s_{\mu\tau+\ell,j},$$

where $\ell = 0, 1, \ldots, p\tau - 1$ and $j = 0, 1, \ldots, k + r - 1$. For any $\ell = e_1, e_2, \ldots, e_\tau$ and $j = 0, 1, \ldots, k + r - 1$, the symbol $s_{\ell-ji,j}$ is erased and the other $p - 1$ symbols $s_{\ell-ji+\tau,j}, s_{\ell-ji+2\tau,j}, \ldots, s_{\ell-ji+(p-1)\tau,j}$ are not erased, as $\tau \nmid (e_\alpha - e_\beta)$ for $\alpha \neq \beta \in \{1, 2, \ldots, \tau\}$. Therefore, we can recover the erased symbol $s_{\ell-ji,j}$ by

$$s_{\ell-ji,j} = s_{\ell-ji+\tau,j} + s_{\ell-ji+2\tau,j} + \ldots + s_{\ell-ji+(p-1)\tau,j}.$$

■

By Theorem 23, $\mathsf{GEBR}(p, \tau, k, r, q, 1)$ can recover up to τ specified erased lines of slope i, for general parameter r. In the following, we consider the code with $r = 2$ and $\tau \geq 2$.

Theorem 24. *If $\tau \geq 2$ and $p\tau > 2(k + r - 1)$, then the code $\mathsf{GEBR}(p, \tau, k, r = 2, q, 1)$ can recover any two erased lines of slope i for $i = 0, 1$.*

Proof: Recall that $s_{i,j}$ is the entry in row i and column j of the array in $\mathsf{GEBR}(p, \tau, k, 2, q, 1)$, where $i = 0, 1, \ldots, p\tau - 1$ and $j = 0, 1, \ldots, k + r - 1$. Suppose that two lines of slope 0 are erased, i.e., rows α and β of the array are erased, where $0 \leq \alpha < \beta \leq p\tau - 1$. We need to recover $s_{\alpha,j}$ and $s_{\beta,j}$ for $j = 0, 1, \ldots, k + r - 1$ from the other symbols.

As $r = 2$, according to Eq. (5), we have that

$$\sum_{j=0}^{k+r-1} s_{i,j} = 0 \text{ for } i = 0, 1, \ldots, p\tau - 1, \tag{30}$$

and

$$\sum_{j=0}^{k+r-1} s_{i-j,j} = 0 \text{ for } i = 0, 1, \ldots, p\tau - 1. \tag{31}$$

If $\tau \nmid (\beta - \alpha)$, then we can recover $s_{\alpha,j}$ and $s_{\beta,j}$ by

$$\begin{aligned} s_{\alpha,j} &= \sum_{\ell=1}^{p-1} s_{\ell\tau+\alpha,j}, \\ s_{\beta,j} &= \sum_{\ell=1}^{p-1} s_{\ell\tau+\beta,j}, \end{aligned} \tag{32}$$

according to Eq. (4).

Next, we assume that $\tau \mid (\beta - \alpha)$. By Eq. (31) with $i = \alpha$ and $i = \beta$, we have

$$s_{\alpha,0} + s_{\alpha-1,1} + \ldots + s_{\alpha-(k+r-1),k+r-1} = 0,$$
$$s_{\beta,0} + s_{\beta-1,1} + \ldots + s_{\beta-(k+r-1),k+r-1} = 0.$$

If $0 \leq \alpha - (k + r - 1)$, or $0 \geq \alpha - (k + r - 1)$ and $\beta < p\tau + \alpha - (k + r - 1)$, then we can recover $s_{\alpha,0}$ by

$$s_{\alpha,0} = s_{\alpha-1,1} + s_{\alpha-2,2} + \ldots + s_{\alpha-(k+r-1),k+r-1}.$$

Otherwise, we have $\beta \geq p\tau + \alpha - (k + r - 1)$, then $\beta - (k + r - 1) \geq p\tau + \alpha - 2(k + r - 1) > \alpha$ by assumption and we can recover $s_{\beta,0}$ by

$$s_{\beta,0} = s_{\beta-1,1} + s_{\beta-2,2} + \ldots + s_{\beta-(k+r-1),k+r-1}.$$

Once $s_{\alpha,0}$ or $s_{\beta,0}$ is known, we can recover $s_{\beta,0}$ or $s_{\alpha,0}$ by Eq. (32) with $j = 0$. By repeating the above procedure for $i = \alpha+1, \alpha+2, \ldots, \alpha+k+r-1$ and $i = \beta+1, \beta+2, \ldots, \beta+k+r-1$, we can recover all $2n$ symbols $s_{\alpha,j}$ and $s_{\beta,j}$ for $j = 0, 1, \ldots, k + r - 1$.

Next, we assume that two lines of slope 1 are erased, i.e., $s_{\alpha,0}, s_{\alpha-1,1}, \ldots, s_{\alpha-(k+r-1),k+r-1}$ and $s_{\beta,0}, s_{\beta-1,1}, \ldots, s_{\beta-(k+r-1),k+r-1}$ are erased. If $\tau \nmid (\beta - \alpha)$, we can recover $s_{\alpha-j,j}$ and $s_{\beta-j,j}$ by

$$\begin{aligned} s_{\alpha-j,j} &= \sum_{\ell=1}^{p-1} s_{\ell\tau+\alpha-j,j}, \\ s_{\beta-j,j} &= \sum_{\ell=1}^{p-1} s_{\ell\tau+\beta-j,j}, \end{aligned} \tag{33}$$

within the column. When $\tau \mid (\beta - \alpha)$, we can recover the symbol $s_{\alpha-(k+r-1),k+r-1}$ by

$$s_{\alpha-(k+r-1),k+r-1} = \sum_{j=0}^{k+r-2} s_{\alpha-(k+r-1),j}$$

if $\alpha-(k+r-1)\geq 0$ or $\alpha-(k+r-1)<0$ and $p\tau+\alpha-(k+r-1)>\beta$, or recover the symbol $s_{\beta,0}$ by

$$s_{\beta,0}=s_{\beta,1}+s_{\beta,2}+\ldots+s_{\beta,k+r-1}$$

if $\beta>p\tau+\alpha-(k+r-1)$. Once $s_{\alpha-(k+r-1),k+r-1}$ or $s_{\beta,0}$ is known, we can recover $s_{\beta-(k+r-1),k+r-1}$ or $s_{\alpha,0}$ by Eq. (32) with $j=k+r-1$ or $j=0$. Similarly, we can recover all $2n$ symbols in the erased two lines of slope 1. ■

The next theorem shows that $\mathsf{GEBR}(p,\tau,k,r=2,q,1)$ can recover more than two erased lines of slope i, if τ is large.

Theorem 25. *If $\tau>k+r-1$, then the code $\mathsf{GEBR}(p,\tau,k,r=2,q,1)$ can recover any three erased lines of slope i for $i=0,1$.*

Proof: Suppose that rows α, β and γ of the array are erased, where $0\leq\alpha<\beta<\gamma\leq p\tau-1$. We want to recover $s_{\alpha,j}$, $s_{\beta,j}$ and $s_{\gamma,j}$ for $j=0,1,\ldots,k+r-1$ from the other symbols.

According to Eq. (5), we have that

$$\begin{array}{l}\sum_{j=0}^{k+r-1}s_{i,j}=0 \text{ for } i=0,1,\ldots,p\tau-1,\\ \sum_{j=0}^{k+r-1}s_{i-j,j}=0 \text{ for } i=0,1,\ldots,p\tau-1.\end{array}\tag{34}$$

If $\tau\nmid(\beta-\alpha)$ and $\tau\nmid(\gamma-\beta)$, then we can recover $s_{\alpha,j}$, $s_{\beta,j}$ and $s_{\gamma,j}$ by

$$\begin{array}{rl}s_{\alpha,j}=&\sum_{\ell=1}^{p-1}s_{\ell\tau+\alpha,j},\\ s_{\beta,j}=&\sum_{\ell=1}^{p-1}s_{\ell\tau+\beta,j},\\ s_{\gamma,j}=&\sum_{\ell=1}^{p-1}s_{\ell\tau+\gamma,j},\end{array}\tag{35}$$

according to Eq. (4). If $\tau\mid(\beta-\alpha)$ and $\tau\nmid(\gamma-\beta)$, then we can first recover $s_{\gamma,j}$ by Eq. (35) and then recover $s_{\alpha,j}$ and $s_{\beta,j}$ by Theorem 24. Similarly, we can recover the erased symbols if $\tau\nmid(\beta-\alpha)$ and $\tau\mid(\gamma-\beta)$.

In the following, we assume that $\tau\mid(\beta-\alpha)$ and $\tau\mid(\gamma-\beta)$. By Eq. (34), we have

$$\begin{aligned}&s_{\alpha+j,0}+s_{\alpha+j-1,1}+\ldots+s_{\alpha-(k+r-1)+j,k+r-1}=0,\\ &s_{\beta+j,0}+s_{\beta+j-1,1}+\ldots+s_{\beta-(k+r-1)+j,k+r-1}=0,\\ &s_{\gamma+j,0}+s_{\gamma+j-1,1}+\ldots+s_{\gamma-(k+r-1)+j,k+r-1}=0,\end{aligned}$$

where $j=0,1,\ldots,k+r-1$. As $\tau>k+r-1$, we have

$\alpha+j<\beta, 0<\alpha-(k+r-1)+j$ or
$0>\alpha-(k+r-1)+j$ and $p\tau+\alpha-(k+r-1)+j>\gamma$,
$\alpha<\beta-(k+r-1)+j$ and $\gamma>\beta+j$,
$\beta<\gamma-(k+r-1)+j$ and $p\tau+\alpha>\gamma+j$,

for $j=0,1,\ldots,k+r-1$ and we can recover $s_{\alpha,j},s_{\beta,j},s_{\gamma,j}$ by

$$\begin{aligned}s_{\alpha,j}&=\sum_{i=0,i\neq j}^{k+r-1}s_{\alpha+j-i,i},\\ s_{\beta,j}&=\sum_{i=0,i\neq j}^{k+r-1}s_{\beta+j-i,i},\\ s_{\gamma,j}&=\sum_{i=0,i\neq j}^{k+r-1}s_{\gamma+j-i,i}.\end{aligned}$$

Similarly, we can recover all $2n$ symbols in any erased three lines of slope 1. ■

The recovery of erased lines of slope i in $\mathsf{GEBR}(p,\tau,k,r,q,1)$ for general parameters τ and r is an open problem and is a subject of future work.

VII. Comparison with LRC and Product Codes

LRCs [17] and product codes [21] are two families of existing codes that can locally repair any single-symbol failure. The differences between our GEBR codes and the existing two codes are as follows.

Given $k\alpha$ information symbols, an $(m(k+r),\alpha k,k+r)$ LRC [17] creates $r\alpha$ global parity symbols by encoding all the information symbols, divides all $(k+r)\alpha$ symbols (including $k\alpha$ information symbols and $r\alpha$ global parity symbols) into $k+r$ groups which are placed into $k+r$ columns and obtains $m-\alpha$ local parity symbols for each group. Compared with LRC, our GEBR codes have two advantages. First, each symbol in our GEBR codes can be repaired by either some symbols in the same column or the symbols along each of r lines of slope, while a symbol in LRC can only be locally repaired within the group. Second, our codes have much lower decoding complexity. When there are r column failures, we can formulate r linear equations for the erased r columns with encoding matrix being Vandermonde matrix for GEBR codes and solve the erased r columns by the proposed fast LU decoding algorithm. While the r linear equations corresponding to the r erased columns (groups) for the LRC are not Vandermonde linear equations, there is no fast decoding algorithm designed for the LRC when r columns have failed. Moreover, although we can obtain some well-designed LRC that can recover some r erased lines of a slope, the underlying field size should be large enough.

Note that LRCs with availability [34], [35] are special LRCs such that each symbol can be repaired with multiple disjoint repair groups. However, the construction of LRCs with availability [34], [35] to achieve the known bound on symbol distance require a sufficiently large field, and therefore incur much more decoding complexity than the proposed codes when some lines are erased.

A product code with parameters k,r,α,m organizes the $k\alpha$ information symbols into an $\alpha\times k$ information array, first creates r local parity symbols for each row and then obtains $m-\alpha$ local parity symbols for each of the $k+r$ columns. Therefore, any symbol can be recovered by accessing some symbols in the same row or in the same column, but not in a line of a non-zero slope. Second, product code can only recover at most any $m-\alpha$ row failures but our GEBR codes can recover at most any $\max\{m-\alpha,r\}$ row failures. Finally, the minimum symbol distance of product code is at most $(m-\alpha+1)(r+1)$, while the minimum symbol distance of GEBR codes is strictly larger than $(m-\alpha+1)(r+1)$ for some parameters.

Table IV shows the comparison of our GEBR codes, LRCs and product codes, when $m=p$ and $\alpha=p-1$. It is easy to check that the three codes have the same storage overhead. When any single-symbol fails, GEBR codes have $r+1$ disjoint repair groups, while LRCs and product codes have one and two

Table IV
COMPARISON WITH $(p^2,(p-1)(p-r),p)$ LRCS AND PRODUCT CODES WITH $m=p,\alpha=p-1$.

Codes	single-symbol failures	r-column failures decoding	r consecutive-row failures	r consecutive-row failures decoding	minimum symbol distance
GEBR	each of $r+1$ lines	fast decoding	yes	fast decoding	$\geq 2r+2$
LRC	one line (the same group)	no fast decoding	maybe over large field	no	$(p-1)r+2$
Product	each of two lines	fast decoding	no	N/A	$2r+2$

repair groups, respectively. In decoding r-column failures, both GEBR codes and product codes have fast decoding algorithm, there is no fast decoding algorithm for LRCs. We can decode any r consecutive-row failures for GEBR codes by the fast LU decoding algorithm. Although it is possible to design LRCs over a large finite field to recover any r consecutive-row failures, there is no fast decoding algorithm for general parameters. LRCs have the largest minimum symbol distance among the three codes. Compared with LRCs, GEBR codes can be viewed as codes with larger recoverability for single-symbol failures, multi-column failures and multi-row failures, at a cost of minimum symbol distance reduction. When compared with product codes, GEBR codes not only have larger recoverability, but also possible have larger minimum symbol distance for some parameters.

Consider the code $\mathsf{GEBR}(p=11,\tau=1,k=7,r=4,q,1)$ in Example 1, we have $k(p-1)=70$ data symbols and $p^2-k(p-1)=51$ local parity symbols. Each symbol has $r+1=5$ disjoint repair groups. We can recover any $r=4$ erased lines e_1,e_2,e_3,e_4 with $0\leq e_1<e_2<e_3<e_4\leq 10$ of slope i for $0\leq i\leq 3$. We can also recover any four column failures by the fast LU decoding algorithm. While for the product code with the same parameters, we can only recover any symbol by two disjoint repair groups, and we can not recover any four erased lines.

VIII. CONCLUSION

In this paper, we propose a coding method of array codes that has local repair property. We present the constructions of GEBR codes and GEIP codes based on the proposed coding method that can support much more parameters, compared with EBR codes and EIP codes, respectively. We propose an efficient LU decoding method for GEBR codes and GEIP codes based on the LU factorization of Vandermonde matrix. When τ is large, we show that GEBR codes have both larger minimum symbol distance and larger recovery ability of erased lines for some parameters, compared with EBR codes. The (n,k) recoverable condition of GEBR codes for general $g(x)$ is one of our future work. How to propose a coding framework to unify GEBR codes, LRCs, and product codes is another future work. It is also interesting to explore some good properties by replacing each column of the proposed codes with regenerating codes.

APPENDIX A
PROOF OF LEMMA 11

We first show that $r_0=\sum_{u=1}^{\frac{p-1}{2}}\sum_{\ell=1}^{\tau}f_{(2u-1)\tau b+\ell b}$. According to Eq. (13), we have

$$r_{\tau b+\ell b}=r_{\tau b+(\ell-1)b}+f_{\tau b+\ell b}, \tag{36}$$

where $\ell=0,1,\ldots,p\tau-1$. Summing both sides of Eq. (36) from $\ell=0$ to $\ell=(i-1)\tau$, we have

$$r_{ib\tau}=r_{\tau b-b}+\sum_{\ell=0}^{(i-1)\tau}f_{\tau b+\ell b}, \tag{37}$$

where $i=1,2,\ldots,p-1$. Summing both sides of Eq. (37) from $i=1$ to $i=p-1$, we have

$$\sum_{i=1}^{p-1}r_{ib\tau}=\sum_{i=1}^{p-1}r_{i\tau}=(p-1)r_{\tau b-b}+\sum_{i=1}^{p-1}\sum_{\ell=0}^{(i-1)\tau}f_{\tau b+\ell b}, \tag{38}$$

where the first equation above comes from that $\gcd(b,p)=1$. By Eq. (3) in Lemma 3, we have $\sum_{i=1}^{p-1}r_{i\tau}=r_0$. Since p is an odd prime number, we have $(p-1)r_{\tau b-b}=0$. We can compute $\sum_{i=1}^{p-1}\sum_{\ell=0}^{(i-1)\tau}f_{\tau b+\ell b}$ as

$$\begin{aligned}&\sum_{i=1}^{p-1}\sum_{\ell=0}^{(i-1)\tau}f_{\tau b+\ell b}=(p-1)f_{\tau b}+(p-2)\sum_{\ell=1}^{\tau}f_{\tau b+\ell b}+\\&(p-3)\sum_{\ell=1}^{\tau}f_{2\tau b+\ell b}+\cdots+2\sum_{\ell=1}^{\tau}f_{(p-3)\tau b+\ell b}+\sum_{\ell=1}^{\tau}f_{(p-2)\tau b+\ell b}\\&=\sum_{u=1}^{\frac{p-1}{2}}\sum_{\ell=1}^{\tau}f_{(2u-1)\tau b+\ell b}.\end{aligned}$$

Therefore, we obtain that $r_0=\sum_{u=1}^{\frac{p-1}{2}}\sum_{\ell=1}^{\tau}f_{(2u-1)\tau b+\ell b}$. Similarly, we can show that Eq. (14) holds for $j=0,1,\ldots,a-1$. Once r_0 is known, we can compute other $\frac{p\tau}{a}-1$ coefficients recursively by Eq. (15) with $j=0$ and $\ell=1,2,\ldots,\frac{p\tau}{a}-1$. Similarly, we can compute $\frac{p\tau}{a}-1$ coefficients recursively by Eq. (15) with $\ell=1,2,\ldots,\frac{p\tau}{a}-1$ for $j=0,1,\ldots,a-1$, after solving r_j.

Next, we need to show that the solved $r(x)$ is in $\mathcal{C}_{p\tau}(g(x),\tau,q,d)$, i.e., $g(x)(1+x^\tau)$ divides $r(x)$. First, $(1+x^\tau)$ divides $r(x)$, as we can show that $\sum_{\ell=0}^{p-1}r_{\ell\tau+\mu}=0$ for $\mu=0,1,\ldots,\tau-1$. Second, since $g(x)$ divides $f(x)$, if $\gcd(1+x^b,g(x))=1$, then $g(x)$ divides $r(x)$. As $\gcd(g(x),1+x)=1$ and $\gcd(p,b)=1$, we have that $\gcd(1+x^b,g(x))=1$ by Lemma 19 in [2]. Therefore, $g(x)(1+x^\tau)$ divides $r(x)$ and the lemma is proved.

APPENDIX B
PROOF OF LEMMA 12

Since $\gcd(b,m)=\gcd(up^s,p^{\nu+1})=p^s$ and $\gcd(u,p)=1$, we have that $\gcd(u,p^{\nu+1})=1$. In the following, we show that

$$\begin{aligned}&\{0,up^s,2up^s,\cdots,u(p^{\nu+1}-2p^s)\}\bmod p^{\nu+1}\\=\;&\{0,p^s,2p^s,\cdots,p^{\nu+1}-2p^s\}.\end{aligned} \tag{39}$$

First, we prove that if $i \neq j \in \{0, p^s, 2p^s, \cdots, p^{\nu+1} - 2p^s\}$, then $u \cdot i \neq u \cdot j \bmod p^{\nu+1}$. Suppose that $u \cdot i = u \cdot j \bmod p^{\nu+1}$, then there exists an integer ℓ such that

$$u \cdot i = u \cdot j + \ell p^{\nu+1},$$

and we can further obtain that

$$u \cdot (i - j) = \ell p^{\nu+1}.$$

Since $\gcd(u, p^{\nu+1}) = 1$, we have $p^{\nu+1} \mid (i - j)$, which contradicts to that $i \neq j \in \{0, p^s, 2p^s, \cdots, p^{\nu+1} - 2p^s\}$. Similarly, we can show that

$$u \cdot i \neq p^{\nu+1} - p^s.$$

Therefore, Eq. (39) holds. According to Eq. (13), we have

$$f_{2iup^s+up^s} = r_{2iup^s+up^s} + r_{2iup^s}, \tag{40}$$

where $i = 0, 1, \ldots, p^{\nu-s+1}-1$. Summing both sides of Eq. (40) from $i = 0$ to $i = \frac{p^{\nu-s+1}-3}{2}$, we have

$$\begin{aligned}
\sum_{i=0}^{\frac{p^{\nu-s+1}-3}{2}} f_{2iup^s+up^s} &= \sum_{i=0}^{\frac{p^{\nu-s+1}-3}{2}} (r_{2iup^s+up^s} + r_{2iup^s}) \\
&= \sum_{i=0}^{p^{\nu-s+1}-2} r_{iup^s} \\
&= \sum_{i=0}^{p^{\nu-s+1}-2} r_{ip^s} \qquad (41) \\
&= r_{(p^{\nu-s+1}-1)p^s} = r_{p^{\nu+1}-p^s}, \qquad (42)
\end{aligned}$$

where Eq. (41) comes from Eq. (39), Eq. (42) comes from that

$$\{0, p^s, 2p^s, \cdots, p^{\nu+1} - 2p^s\} =$$
$$\begin{cases}
\{0, p^\nu, \cdots, (p-1)p^\nu\} \cup \{p^s, p^\nu + p^s, \cdots, (p-1)p^\nu + p^s\} \\
\cup \cdots \cup \{2p^\nu - p^s, 3p^\nu - p^s, \cdots, (p-1)p^\nu - p^s\}, \text{ if } \nu > s, \\
\{0, p^\nu, 2p^\nu, \cdots, (p-2)p^\nu\}, \text{ if } \nu = s.
\end{cases}$$

Similarly, we can show that Eq. (16) holds for $j = 0, 1, \ldots, m-1$. Once $r_{p^{\nu+1}-p^s+j}$ for $j = 0, 1, \ldots, p^s - 1$ are known, we can compute the other coefficients recursively.

Recall that $\sum_{i=0}^{\frac{p^{\nu-s+1}-3}{2}} f_{2iup^s+up^s+j} = r_{p^{\nu+1}-p^s+j}$ for $j = 0, 1, \ldots, m-1$ by Eq. (42), we have

$$r_i = f_{up^s+p^s+i} + f_{3up^s+p^s+i} + \cdots + f_{(p^{\nu-s+1}-2)up^s+p^s+i}$$

for $i = 0, 1, \ldots, m$. Recall that the indices are taken modulo $m = p^{\nu+1}$. We have

$$r(x) = (x^{p^{\nu+1}-up^s-p^s} + x^{p^{\nu+1}-3up^s-p^s} + \cdots + x^{2up^s-p^s}) f(x).$$

Since $f(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$, we have that $r(x) \in \mathcal{C}_{p\tau}(g(x), \tau, q, d)$ and the lemma is proved.

REFERENCES

[1] W. You, H. Hou, Y. S. Han, P. P. C. Lee, and G. Han, "Generalized Expanded-Blaum-Roth Codes and Their Efficient Encoding/Decoding," in *Proc. IEEE GLOBECOM*, 2020, pp. 1–6.

[2] M. Blaum and S. R. Hetzler, "Array Codes with Local Properties," *IEEE Trans. Information Theory*, vol. 66, no. 6, pp. 3675–3690, 2020.

[3] D. A. Patterson, P. Chen, G. Gibson, and R. H. Katz, "Introduction to Redundant Arrays of Inexpensive Disks (RAID)," in *Digest of Papers. COMPCON Spring 89. Thirty-Fourth IEEE Computer Society International Conference: Intellectual Leverage*, 1989, pp. 112–117.

[4] M. Blaum, J. Brady, J. Bruck, and Jai Menon, "EVENODD: An Efficient Scheme for Tolerating Double Disk Failures in RAID Architectures," *IEEE Trans. on Computers*, vol. 44, no. 2, pp. 192–202, 1995.

[5] H. Hou and P. P. C. Lee, "A New Construction of EVENODD Codes With Lower Computational Complexity," *IEEE Communications Letters*, vol. 22, no. 6, pp. 1120–1123, 2018.

[6] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, "Row-Diagonal Parity for Double Disk Failure Correction," in *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*. San Francisco, CA, 2004, pp. 1–14.

[7] C. Huang and L. Xu, "STAR: An Efficient Coding Scheme for Correcting Triple Storage Node Failures," *IEEE Trans. on Computers*, vol. 57, no. 7, pp. 889–901, 2008.

[8] H. Hou and P. P. C. Lee, "STAR+ Codes: Triple-Fault-Tolerant Codes with Asymptotically Optimal Updates and Efficient Encoding/Decoding," in *Proceedings of the 2021 IEEE Information Theory Workshop (ITW 2021)*, 2021.

[9] H. Hou, P. P. C. Lee, Y. S. Han, and Y. Hu, "Triple-Fault-Tolerant Binary MDS Array Codes with Asymptotically Optimal Repair," in *2017 IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 839–843.

[10] M. Blaum, "A Family of MDS Array Codes with Minimal Number of Encoding Operations," in *IEEE International Symposium on Information Theory*, 2006.

[11] M. Blaum, J. Brady, J. Bruck, J. Jai Menon, and A. Vardy, "The EVENODD Code and its Generalization: An Effcient Scheme for Tolerating Multiple Disk Failures in RAID Architectures," in *High Performance Mass Storage and Parallel I/O*. Wiley-IEEE Press, 2002, ch. 8, pp. 187–208.

[12] M. Blaum and R. M. Roth, "New Array Codes for Multiple Phased Burst Correction," *IEEE Trans. Information Theory*, vol. 39, no. 1, pp. 66–77, 1993.

[13] H. Hou, K. W. Shum, M. Chen, and H. Li, "New MDS Array Code Correcting Multiple Disk Failures," in *Proc. IEEE GLOBECOM*, 2014, pp. 2369–2374.

[14] G. L. Feng, R. H. Deng, F. Bao, and J.-C. Shen, "New Efficient MDS Array Codes for RAID. Part II. Rabin-Like Codes for Tolerating Multiple (≥ 4) Disk Failures," *IEEE Trans. on Computers*, vol. 54, no. 12, pp. 1473–1483, 2005.

[15] H. Hou and Y. S. Han, "A New Construction and an Efficient Decoding Method for Rabin-Like Codes," *IEEE Trans. Communications*, vol. 66, no. 2, pp. 521–533, 2018.

[16] M. Blaum, V. Deenadhayalan, and S. Hetzler, "Expanded Blaum-Roth Codes With Efficient Encoding and Decoding Algorithms," *IEEE Communications Letters*, vol. 23, no. 6, pp. 954–957, 2019.

[17] I. Tamo and A. Barg, "A Family of Optimal Locally Recoverable Codes," *IEEE Trans. Information Theory*, vol. 60, no. 8, pp. 4661–4676, 2014.

[18] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the Locality of Codeword Symbols," *IEEE Trans. Information Theory*, vol. 58, no. 11, pp. 6925–6934, 2012.

[19] C. Huang, M. Chen, and J. Li, "Pyramid Codes: Flexible Schemes to Trade Space for Access Efficiency in Reliable Data Storage Systems," *ACM Transactions on Storage (TOS)*, vol. 9, no. 1, pp. 1–28, 2013.

[20] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "XORing Elephants: Novel Erasure Codes for Big Data," in *Proceedings of the 39th international conference on Very Large Data Bases*. VLDB Endowment, 2013, pp. 325–336.

[21] P. Gopalan, G. Hu, S. Kopparty, S. Saraf, C. Wang, and S. Yekhanin, "Maximally Recoverable Codes for Grid-like Topologies," in *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2017, pp. 2092–2108.

[22] A. S. Rawat, D. S. Papailiopoulos, A. G. Dimakis, and S. Vishwanath, "Locality and Availability in Distributed Storage," *IEEE Trans. Information Theory*, vol. 62, no. 8, pp. 4481–4493, 2016.

[23] A. Barg, I. Tamo, and S. Vlăduţ, "Locally Recoverable Codes on Algebraic Curves," *IEEE Trans. Information Theory*, vol. 63, no. 8, pp. 4928–4939, 2017.

[24] X. Kong, X. Wang, and G. Ge, "New Constructions of Optimal Locally Repairable Codes with Super-Linear Length," *IEEE Trans. Information Theory*, vol. 67, no. 10, pp. 6491–6506, 2021.

[25] H. Hou, Y. S. Han, P. P. C. Lee, Y. Hu, and H. Li, "A New Design of Binary MDS Array Codes with Asymptotically Weak-Optimal Repair," *IEEE Trans. Information Theory*, vol. 65, no. 11, pp. 7095–7113, 2019.

[26] H. Hou, Y. S. Han, P. P. C. Lee, and Q. Zhou, "New Regenerating Codes over Binary Cyclic Codes," in *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 216–220.
[27] M. Blaum, J. Bruck, and A. Vardy, "MDS Array Codes with Independent Parity Symbols," *IEEE Trans. Information Theory*, vol. 42, no. 2, pp. 529–542, 1996.
[28] H. Hou, K. W. Shum., M. Chen, and H. Li, "BASIC Codes: Low-Complexity Regenerating Codes for Distributed Storage Systems," *IEEE Trans. Information Theory*, vol. 62, no. 6, pp. 3053–3069, 2016.
[29] H. Hou, Y. S. Han, K. W. Shum, and H. Li, "A Unified Form of EVENODD and RDP Codes and Their Efficient Decoding," *IEEE Trans. Communications*, vol. 66, no. 11, pp. 5053–5066, 2018.
[30] T. Itoh, "Characterization for a Family of Infinitely Many Irreducible Equally Spaced Polynomials," *Information Processing Letters*, vol. 37, no. 5, pp. 273–277, 1991.
[31] S.-L. Yang, "On The LU factorization of The Vandermonde Matrix," *Discrete Applied Mathematics*, vol. 146, no. 1, pp. 102–105, 2005.
[32] H. Hou, K. W. Shum, and H. Li, "On the MDS Condition of Blaum-Bruck-Vardy Codes With Large Number Parity Columns," *IEEE Communications Letters*, vol. 20, no. 4, pp. 644–647, 2016.
[33] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Elsevier, 1977, vol. 16.
[34] A. Wang and Z. Zhang, "Repair Locality with Multiple Erasure Tolerance," *IEEE Trans. Information Theory*, vol. 60, no. 11, pp. 6979–6987, 2013.
[35] I. Tamo, A. Barg, and A. Frolov, "Bounds on the Parameters of Locally Recoverable Codes," *IEEE Trans. Information Theory*, vol. 62, no. 6, pp. 3070–3083, 2016.

Hanxu Hou received the B.Eng. degree in Information Security from Xidian University, Xian, China, in 2010, and Ph.D. degrees in the Dept. of Information Engineering from the Chinese University of Hong Kong in 2015 and in the School of Electronic and Computer Engineering from Peking University in 2016. He is now an Associate Professor with Dongguan University of Technology. He was a recipient of the 2020 Chinese Information Theory Young Rising Star Award by China Information Theory Society. He was recognized as an Exemplary Reviewer 2020 in IEEE Transactions on Communications. His research interests include erasure coding and coding for distributed storage systems.

Yunghsiang S. Han (S'90-M'93-SM'08-F'11) was born in Taipei, Taiwan, 1962. He received B.Sc. and M.Sc. degrees in electrical engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 1984 and 1986, respectively, and a Ph.D. degree from the School of Computer and Information Science, Syracuse University, Syracuse, NY, in 1993. He was from 1986 to 1988 a lecturer at Ming-Hsin Engineering College, Hsinchu, Taiwan. He was a teaching assistant from 1989 to 1992, and a research associate in the School of Computer and Information Science, Syracuse University from 1992 to 1993. He was, from 1993 to 1997, an Associate Professor in the Department of Electronic Engineering at Hua Fan College of Humanities and Technology, Taipei Hsien, Taiwan. He was with the Department of Computer Science and Information Engineering at National Chi Nan University, Nantou, Taiwan from 1997 to 2004. He was promoted to Professor in 1998. He was a visiting scholar in the Department of Electrical Engineering at University of Hawaii at Manoa, HI from June to October 2001, the SUPRIA visiting research scholar in the Department of Electrical Engineering and Computer Science and CASE center at Syracuse University, NY from September 2002 to January 2004 and July 2012 to June 2013, and the visiting scholar in the Department of Electrical and Computer Engineering at University of Texas at Austin, TX from August 2008 to June 2009. He was with the Graduate Institute of Communication Engineering at National Taipei University, Taipei, Taiwan from August 2004 to July 2010. From August 2010 to January 2017, he was with the Department of Electrical Engineering at National Taiwan University of Science and Technology as Chair Professor. From February 2017 to February 2021, he was with School of Electrical Engineering & Intelligentization at Dongguan University of Technology, China. Now he is with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China. He is also a Chair Professor at National Taipei University from February 2015. His research interests are in error-control coding, wireless networks, and security.

Dr. Han was a winner of the 1994 Syracuse University Doctoral Prize and a Fellow of IEEE. One of his papers won the prestigious 2013 ACM CCS Test-of-Time Award in cybersecurity.

Patrick P. C. Lee received the B.Eng. degree (first class honors) in Information Engineering from the Chinese University of Hong Kong in 2001, the M.Phil. degree in Computer Science and Engineering from the Chinese University of Hong Kong in 2003, and the Ph.D. degree in Computer Science from Columbia University in 2008. He is now a Professor of the Department of Computer Science and Engineering at the Chinese University of Hong Kong. His research interests are in various applied/systems topics including storage systems, distributed systems and networks, operating systems, dependability, and security.

You Wu received the M.Phil. degree from Guangdong University of Technology in 2021. She is now in Beijing Didi Infinity Technology and Development Co., Ltd. Her research interests include the coding for distributed storage systems.

Guojun Han received the M.E. degree from South China University of Technology, Guangzhou, China, and the Ph.D. degree from Sun Yatsen University, Guangzhou, China. From March 2011 to August 2013, he was a Research Fellow at the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. From October 2013 to April 2014, he was a Research Associate at the Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology. He is now a Full Professor and Executive Dean at the School of Information Engineering, Guangdong University of Technology, Guangzhou, China.

He has been a Senior Member of IEEE since 2014. His research interests are in the areas of wireless communications, signal processing, coding and information theory. He has more than 15 years experience on research and development of advanced channel coding and signal processing algorithms and techniques for various data storage and communication systems.

Mario Blaum (Life Fellow, IEEE) was born in Buenos Aires, Argentina. He received the Licenciado degree from the University of Buenos Aires in 1977, the M.Sc. degree from the TechnionIsrael Institute of Technology in 1981, and the Ph.D. degree from the California Institute of Technology (Caltech) in 1984, all in mathematics.

In 1985, he was a Research Fellow at the Department of Electrical Engineering, Caltech. In 1985, he joined the IBM Research Division, Almaden Research Center. In 2003, his division was transferred to Hitachi Global Storage Technologies, where he was a Research Staff Member until 2009, in which he rejoined the IBM Almaden Research Center. Since 2001, he has been an Academic Advisor at the Universidad Complutense of Madrid, Spain. He retired in 2021. His research interest includes all aspects of coding for storage technology.