Variant Codes Based on A Special Polynomial Ring and Their Fast Computations

Leilei Yu, Yunghsiang S. Han, *Fellow, IEEE*, Jiasheng Yuan, and Zhongpei Zhang

***Abstract*—Binary array codes are widely used in storage systems to prevent data loss, such as the Redundant Array of Independent Disks (RAID). Most designs for such codes, such as Blaum-Roth (BR) codes and Independent-Parity (IP) codes, are carried out on the polynomial ring $\mathbb{F}_2[x]/\langle\sum_{i=0}^{p-1} x^i\rangle$, where $\mathbb{F}_2$ is a binary field, and p is a prime number. In this paper, we consider the polynomial ring $\mathbb{F}_2[x]/\langle\sum_{i=0}^{p-1} x^{i\tau}\rangle$, where $p>1$ is an odd number and $\tau \geq 1$ is any power of two, and explore variant codes from codes over this polynomial ring. Particularly, the variant codes are derived by mapping parity-check matrices over the polynomial ring to binary parity-check matrices.**

Specifically, we first propose two classes of variant codes, termed V-ETBR and V-ESIP codes. To make these variant codes binary maximum distance separable (MDS) array codes that achieve optimal storage efficiency, this paper then derives the connections between them and their counterparts over polynomial rings. These connections are general, making it easy to construct variant MDS array codes from various forms of matrices over polynomial rings. Subsequently, some instances are explicitly constructed based on Cauchy and Vandermonde matrices. In the proposed constructions, both V-ETBR and V-ESIP MDS array codes can have any number of parity columns and have the total number of data columns of exponential order with respect to p. In contrast, previous binary MDS array codes only have a total number of data columns of linear order with respect to p. This makes the codes proposed in this paper more suitable for application to large-scale storage systems. In terms of computation, two fast syndrome computations are proposed for the Vandermonde-based V-ETBR and V-ESIP MDS array codes, both meeting the lowest known asymptotic complexity among MDS codes. Due to the fact that all variant codes are constructed from parity-check matrices over simple binary fields instead of polynomial rings, they are attractive in practice.

***Index Terms*—Storage systems, binary array code, binary parity-check matrix, syndrome computation.**

I. Introduction

Modern distributed storage systems require data redundancy to maintain data reliability and durability in the presence of unpredictable failures. Replications and erasure codes are two typical redundancy mechanisms [1], [2]. Compared to the former, erasure codes only need less data redundancy to attain the same level of data protection [3]. One well-known class of erasure codes is *binary array codes* [4]–[7]. Their coding procedures involve only XOR (exclusive OR) and cyclic shift operations, which enables simple and efficient implementations in both software and hardware [8]. This paper focuses on such codes.

Binary array codes have been widely used in storage systems, such as RAID (Redundant Array of Independent Disks) [9]. With the development of distributed storage systems in recent years, they have also been used as the basis for developing other erasure codes, such as locally repairable codes [2], [8], [10], [11] and regenerating codes [12]–[14]. For an $\ell \times (k+r)$ binary array code, any codeword can be viewed as an $\ell \times (k+r)$ array of bits, where k columns store all information bits to form k information columns, and the remaining columns store all the parity bits encoded from information bits to form r parity columns. The row size ℓ generally depends on the code construction. In coding theory, maximum distance separable (MDS) codes reach optimal storage efficiency [15], and each of their codewords consists of information and parity symbols, such that any subset of symbols in the codeword with the same number as information symbols can recover the entire codeword. *Binary MDS array codes* have the same property by treating each column as a symbol. More precisely, for an $\ell \times (k+r)$ binary MDS array code, any k out of $k+r$ columns suffice to decode (reconstruct) all columns. Some well-known examples of binary array codes are EVENODD [16], row-diagonal parity (RDP) [17], STAR [18], and triple-fault-tolerance codes [19]. These codes are all binary MDS array codes for the case of two or three parity columns. Examples of binary array codes with more parity columns are Blaum-Roth (BR) [4], Independent-Parity (IP) [5], generalized RDP codes [6], and the codes in [20]. Although they are not always binary MDS array codes, the conditions that render them such codes can be found in the corresponding literature.

The new binary array codes proposed in this paper target an arbitrary number of parity columns, and their constructions are closely related to the BR, IP, and generalized RDP codes mentioned above. Specifically, BR and IP codes are both constructed by parity-check matrices over the polynomial ring $\mathbb{F}_2[x]/\langle\sum_{i=0}^{p-1} x^i\rangle$, where $\mathbb{F}_2$ denotes a binary field and p is a prime number [4], [5]. Generalized RDP codes can be regarded as a variant of shortened IP codes [6], and they possess lower computational complexity [21]. In this paper, we reformulate the generalized RDP codes, and then one can intuitively understand the essence of the generalized RDP codes being more computationally superior. Briefly, when computing syndromes, the codes over $\mathbb{F}_2[x]/\langle\sum_{i=0}^{p-1} x^i\rangle$ are first calculated in an auxiliary polynomial ring $\mathbb{F}_2[x]/\langle x^p+1\rangle$, where multiplying x only requires performing a simple cyclic shift operation. Then all results are returned to the original ring [4], [5]. As

L. Yu, Y. S. Han, J. Yuan and Z. Zhang are with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen, China (e-mail: yuleilei@uestc.edu.cn, yunghsiangh@gmail.com, 202312281024@std.uestc.edu.cn, Zhangzp@uestc.edu.cn). This work was supported by the National Key Research and Development Program of China under Grant 2022YFA1004902.

a variant, the generalized RDP codes have a similar process to the shortened IP codes in computing syndromes, with the only difference being that they do not process the extra bits of the auxiliary polynomial ring compared to the original ring. Thus, the generalized RDP codes eliminate two operations in the shortened IP codes when computing syndromes. One is the processing for one fixed bit in each symbol over the auxiliary ring, and the other is the modulo operation for returning to the original ring. A binary parity-check matrix for the generalized RDP codes is explicitly provided in this paper (Please refer to (12)).

In fact, this paper generalizes the above variant technique so that new codes based on binary parity-check matrices can be easily obtained from codes over the polynomial ring $\mathbb{F}_2[x]/\langle\sum_{i=0}^{p-1} x^{i\tau}\rangle$, where p is an odd number and τ is any power of two. In our setup, the parity-check matrices of codes over the polynomial ring can be determined not only by the Vandermonde matrices containing only monomials (e.g. BR, IP codes) but also by matrices with more forms (e.g. Cauchy matrices, etc.) and wider parameter ranges. In this paper, two classes of codes defined in $\mathbb{F}_2[x]/\langle\sum_{i=0}^{p-1} x^{i\tau}\rangle$ are referred to as ETBR and ESIP codes, which can be regarded as extensions of BR and shortened IP codes, respectively. Correspondingly, the variants of ETBR and ESIP codes are referred to as V-ETBR and V-ESIP codes, respectively. The main contributions of this paper are enumerated as follows:

1) This paper proposes two new classes of binary array codes (i.e., V-ETBR and V-ESIP codes), which are both based on binary parity-check matrices (see Sec. III). We show that the well-known generalized RDP codes are a special case of the V-ESIP codes.
2) This paper presents the conditions for the new codes to be binary MDS array codes by exploring the connections between them and their counterparts over the polynomial ring (see Sec. IV). In particular, these connections are built on the foundation that all parity-check matrices have a sufficiently flexible form. This provides convenience for constructing V-ETBR/V-ESIP MDS array codes with various forms.
3) Based on Vandermonde and Cauchy matrices, this paper explicitly provides the constructions for the V-ETBR and V-ESIP MDS array codes, both with any number of parity columns r (see Sec. V). Compared to previous binary MDS array codes over the polynomial ring, the constructed codes have significantly more data columns for a given design parameter p, as well as a more flexible row size ℓ.
4) This paper also proposes two fast syndrome computations, which respectively correspond to the V-ETBR MDS array codes with any $r \geq 2$ (see Sec. V-B1) and the V-ESIP MDS array codes with $r = 4$ (see Sec. V-B2). Both of them meet the lowest known asymptotic computational complexity among MDS codes [1], i.e., each data bit requires $\lfloor \lg r \rfloor + 1$ XORs as the total number of data columns approaches infinity.

In this paper, the proposed fast syndrome computations can be seen as an extension of the syndrome computation in Reed-Solomon (RS) codes over finite fields [1] to the variant codes. In [1], the computation involved in RS codes can generate a large amount of intermediate data through the Reed-Muller (RM) transform to reduce the total number of operations. Some variant codes constructed in this paper are based on Vandermonde matrices (over polynomial rings) with a similar structure as in [1], and the fast computation in RS codes is compatible with these constructed variant codes. In this paper, the fast computations proposed for variant codes can be easily adjusted to be suitable for the corresponding codes over the polynomial ring. To avoid tediousness, we will not repeat the presentation. Note that the variant codes are based on binary parity-check matrices, leading to easy implementation through the use of existing open-source libraries for matrix operations over $\mathbb{F}_2$, such as M4RI [22]. This means that engineers can use them without needing to have much knowledge of algebra. At the end of this paper, we also compared the specific number of XORs required for encoding and decoding of the variant codes with other alternative binary MDS array codes, i.e., Circulant Cauchy code [23], Rabin-like code [24], and BR code [4], [25]. When the total number of data columns is 251, and the number of parity columns ranges from 4 to 7, the average encoding/decoding improvements of variant codes compared to them are 69%/69%, 63%/61%, and 26%/22%, respectively. Since the variant codes are based on simple binary parity-check matrices, there is still a great potential to further improve computational efficiency by using scheduling algorithms for binary matrix multiplication, such as [26], [27], etc.

Recently, [8], [28], and [29] proposed some new binary MDS array codes. Their idea is to construct binary parity-check matrices by truncating circulant matrices of elements over polynomial rings. The resulting binary MDS array codes are essentially V-ETBR/V-ESIP codes, and this paper can be seen as a generalization of their works. This generalization extends parity-check matrices restricted to Vandermonde forms to having arbitrary matrix forms, as well as extends the Vandermonde-based syndrome computation in their works, which is only applicable to $2 \leq r \leq 3$, to supporting arbitrary $r \geq 2$. Furthermore, one of the main contributions of this paper is to propose the intrinsic connections between codes over the polynomial ring and V-ETBR/V-ESIP codes. This was not considered in the previous work. Particularly, these connections provide a powerful tool for constructing binary MDS array codes over binary fields. The detailed differences between the previous work and this paper are enumerated as follows:

1) This paper clearly reveals the relationship between V-ETBR/V-ESIP codes and the well-known generalized RDP codes, as the former is a generalization of the variant technique implied by the latter. This was not pointed out in the previous work.
2) In the previous work, the V-ETBR/V-ESIP codes consider only binary parity-check matrices determined by Vandermonde matrices. In contrast, the matrices used in this paper have a more flexible form, of which the Vandermonde matrix is just a special instance. This can

facilitate the construction of more variant codes.

3) The previous work focuses only on V-ETBR/V-ESIP codes without discussing their connections with the corresponding codes over polynomial rings. In this paper, we consider these connections and show that, based on them, new MDS codes over polynomial rings can be directly obtained as by-products.
4) In terms of construction, all MDS array codes proposed in the previous work and this paper can have a total number of data columns far exceeding the design parameter p. However, the feasible number of parity columns for the V-ESIP MDS array codes in the previous work is three, while that in this paper is any size.
5) In terms of computation, fast syndrome computation in the previous work is for $2 \leq r \leq 3$, whereas that proposed in this paper is for arbitrary $r \geq 2$. The former is a special case of the latter.

The remainder of this paper is organized as follows. Section II introduces all necessary preliminaries, including some existing well-known binary array codes and important notations. Section III provides the specific definitions of ESIP/ESIP and V-ETBR/V-ESIP codes. By exploring the general connections between V-ETBR/V-ESIP codes and their counterparts over polynomial rings (i.e., ESIP/ESIP codes), Section IV proposes the conditions that make variant codes binary MDS array codes. In Section V, some explicit constructions for V-ETBR and V-ESIP MDS array codes are proposed, along with their fast syndrome computations. Section VI concludes this paper.

II. Preliminaries

This section describes some existing well-known classes of array codes, i.e., BR codes [4], IP codes [5], and generalized RDP codes [6]. To begin with, let

$$\mathbb{R}_{p,\tau} := \frac{\mathbb{F}_2[x]}{\langle f_{p,\tau}(x) \rangle} \tag{1}$$

denote a binary polynomial ring, where

$$f_{p,\tau}(x) = 1 + x^{\tau} + \cdots + x^{(p-1)\tau} \tag{2}$$

with two positive integers p, τ. The identity that $x^{p\tau} + 1 = (x^{\tau} + 1) \cdot f_{p,\tau}(x)$ leads to operations in $\mathbb{R}_{p,\tau}$ that can be performed first in polynomial ring

$$\mathbb{R} := \frac{\mathbb{F}_2[x]}{\langle x^{p\tau} + 1 \rangle} , \tag{3}$$

and then, all results should be reduced modulo $f_{p,\tau}(x)$. Since multiplying by x in $\mathbb{R}$ is equivalent to performing a one-bit cyclic shift on a vector with $p\tau$ bits, the above realization for the operations in $\mathbb{R}_{p,\tau}$ is simple and efficient [4], [5].

A. BR codes

BR codes are constructed in polynomial ring $\mathbb{R}_{p,1}$ [4], where p is a prime number. Given the value of p, the BR$(p, r < p)$ is defined as the set of $(p-1)\times p$ arrays (denoted by $[x_{i,j}]$, where $x_{i,j} \in \{0, 1\}$, the first $p - r$ data columns are information columns and others are parity columns). For $\ell = 0, 1, ..., p-1$, the ℓ-th column of a $(p-1)\times p$ array can be viewed as a binary polynomial $D_\ell = \sum_{i=0}^{p-2} x_{i,\ell} \cdot x^i \in \mathbb{R}_{p,1}$. The BR$(p, r)$ requires that $\mathbf{0}^{\mathrm{T}} = H_{BR} \cdot (D_0, D_1, ..., D_{p-1})^{\mathrm{T}}$, where $H_{BR} \in \mathbb{R}_{p,1}^{r\times p}$ is the Vandermonde parity-check matrix given by

$$H_{BR} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & x & x^2 & \cdots & x^{p-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x^{r-1} & x^{2(r-1)} & \cdots & x^{(r-1)(p-1)} \end{pmatrix}, \tag{4}$$

and $\mathbf{0}$ is a zero-row vector.

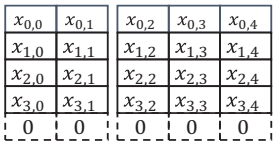

Fig. 1. Diagram of the BR code with $p = 5$ and $r = 3$.

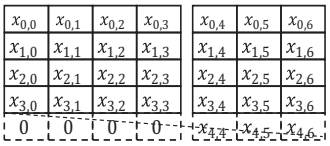

Fig. 2. Diagram of the generalized RDP code with $p = 5$ and $r = 3$.

BR codes have an intuitive graphical representation and Fig. 1 provides an example of BR$(5, 3)$ to demonstrate it. In Fig. 1, the last row is imaginary to facilitate operations, the leftmost two data columns are information columns of the BR$(5, 3)$, and the rightmost three data columns are all parity columns. According to the identity $\mathbf{0}^{\mathrm{T}} = H_{BR} \cdot (D_0, D_1, ..., D_{p-1})^{\mathrm{T}}$, the result obtained by bit-wise XORing all data columns is an all-zero column. If each column has been subjected to down-cyclic shifts according to the corresponding column index size, the above result is either an all-zero column or an all-one column. This satisfies the need for realization in $\mathbb{R}_{p,1}$, which involves first performing operations in $\mathbb{R}$ and then reducing to $\mathbb{R}_{p,1}$. The above result is also true if the number of down-cyclic shifts is twice the size of the corresponding column index. One can know from [4] that BR codes are always binary MDS array codes.

B. IP codes

IP codes are also constructed in $\mathbb{R}_{p,1}$, but all parity columns are independent of each other, leading to a minimization of the number of parity updates when a data bit is updated [5], [10], [16]. Precisely, given the prime number p and a positive integer r, the IP$(p + r, r)$ is defined as the set of $(p - 1) \times (p + r)$ arrays of bits. In the same way as the BR codes, each column of the array forms a binary polynomial, then the parity-check matrix of the IP$(p + r, r)$ is $H_{IP} = (H_{BR}|I_r)$, where H_{BR} is shown in (4) and I_r is an $r \times r$ identity matrix. The matrix H_{IP} implies that IP codes also have an intuitive graphical representation similar to that shown in BR codes. Contrary to BR codes, IP codes are not always binary MDS array codes. The conditions for making IP codes to be binary MDS array codes can be found in [5], [7].

C. Generalized RDP codes

In [17], the authors presented a binary MDS array code with two parity columns, i.e., RDP codes. This code was generalized to support more parity columns in [6]. Generalized

RDP codes are not directly constructed by parity-check matrices over $\mathbb{R}_{p,1}$ like the two codes introduced above. Given a prime number p and a positive integer r, the generalized RDP$(p+r-1,r)$ code is defined as the set of $(p-1)\times(p+r-1)$ arrays (denoted by $[x_{i,j}]$, where $x_{i,j} \in \{0,1\}$, the first $p-1$ data columns are information columns, and others are parity columns). From [6], it satisfies the following encoding equations:

$$x_{i,p-1} = \sum_{j=0}^{p-2} x_{i,j} \text{ for } 0 \le i \le p-2 \,, \tag{5}$$

$$\text{and } \; x_{i,p-1+j} = \sum_{\ell=0}^{p-1} x_{i-j\ell,\ell} \text{ for } \begin{matrix} 0 \le i \le p-2 \\ 1 \le j \le r-1 \end{matrix} \,, \tag{6}$$

where addition is performed through XOR, all subscripts in the right-hand side of equal signs are modulo p, and $x_{p-1,j} = 0$ for $j = 0, 1, ..., p-1$.

Similar to BR and IP codes, the generalized RDP codes have an intuitive graphical representation. Fig. 2 shows an example of $p = 5$ and $r = 3$, where the leftmost four data columns are information columns and the last row is imaginary. Clearly, the first parity column, i.e., $\{x_{i,4}\}_{i=0}^{4}$, is obtained by bit-wise XORing the first 4 columns. The second parity column, i.e., $\{x_{i,5}\}_{i=0}^{4}$, is obtained by bit-wise XORing the first 5 columns after each column has been subjected to down-cyclic shifts according to the corresponding column index size. The third parity column is similar to the second, but the number of down-cyclic shifts in each column becomes twice the corresponding column index size. The three parity columns of the generalized RDP$(7,3)$ code are obtained by directly deleting the imaginary row.

Generalized RDP codes are not always binary MDS array codes [6], as are IP codes. Conditions that make generalized RDP codes to be binary MDS array codes can be found in [6]. In particular, there is a connection between generalized RDP and IP codes as follows:

Theorem 1. *([6]) The generalized RDP$(p+r-1,r)$ is a binary MDS array code if the shortened IP$(p+r-1,r)$ with the following parity-check matrix over $\mathbb{R}_{p,1}$ is a binary MDS array code*

$$H_{SIP} = \left(\begin{array}{c|ccc} & 0 & 0 & \cdots & 0 \\ & 1 & 0 & \cdots & 0 \\ H_{BR} & 0 & 1 & \cdots & 0 \\ & \vdots & \vdots & \ddots & \vdots \\ & 0 & 0 & \cdots & 1 \end{array} \right). \tag{7}$$

The encoding of the shortened IP code with (7) can be analogized from the BR code in Section II-A. It is easy to see that the process is similar to that in the generalized RDP$(p+r-1,r)$. The only difference is that the latter does not need to calculate the last bit in each parity column and modulo $f_{p,1}(x)$.

D. Notations

Throughout this paper, the set $\{0,1,2,3,...\}$ is denoted by $\mathbb{N}$ and the set $\{i, i+1, ..., j-1\}$ is denoted by $[i,j)$, where $i \in \mathbb{N}, j \in \mathbb{N}$ with $i < j$. The transpose of a matrix or vector is marked with the notation T in the upper right-hand corner. Unless otherwise stated, suppose that

$$m = p\tau, \qquad \tau = 2^s, \tag{8}$$

where $p > 1$ is an odd number and $s \in \mathbb{N}$. Note that p and τ are determined if m is given. In addition, $f_{p,\tau}(x) = f_{p,1}^{\tau}(x)$.

Some special mappings are defined below. For any $i, j \in \mathbb{N}$ and $a = \sum_{i=0}^{m-1} a_i \cdot x^i \in \mathbb{R}$, define a mapping $\mathcal{A}_{i,j} : \mathbb{R} \to \mathbb{F}_2^{(m-i)\times(m-j)}$ by letting $\mathcal{A}_{i,j}(a)$ be the resultant $(m-i) \times (m-j)$ binary matrix after deleting the last i rows and last j columns of the following $m \times m$ binary circulant matrix

$$\begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{m-1} \\ a_{m-1} & a_0 & a_1 & \cdots & a_{m-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & a_3 & \cdots & a_0 \end{pmatrix}. \tag{9}$$

That is,

$$\mathcal{A}_{i,j}(a) = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{m-1-j} \\ a_{m-1} & a_0 & a_1 & \cdots & a_{m-2-j} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{1+i} & a_{2+i} & a_{3+i} & \cdots & a_{m-j+i} \end{pmatrix}, \tag{10}$$

where each subscript is modulo m. From [30], one can see that $\mathcal{A}_{0,0}$ is an isomorphic mapping. Moreover, for $i \in \mathbb{N}$, $\mathcal{A}_{i,i}(0)$ is the $(m-i) \times (m-i)$ zero matrix and $\mathcal{A}_{i,i}(1)$ is an $(m-i) \times (m-i)$ identity matrix. Furthermore, for any $\ell_0, \ell_1 \in \mathbb{N}$, we define a mapping from the set consisting of all $\ell_0 \times \ell_1$ matrices over $\mathbb{R}$ to the set consisting of all $\ell_0(m-\tau) \times \ell_1(m-\tau)$ matrices over $\mathbb{F}_2$, i.e.,

$$\mathcal{T}_{\ell_0,\ell_1,m} : M_{\ell_0\times\ell_1}(\mathbb{R}) \to M_{\ell_0(m-\tau)\times\ell_1(m-\tau)}(\mathbb{F}_2) \tag{11}$$

by letting $\mathcal{T}_{\ell_0,\ell_1,m}(B) = \overline{B}$, where $B = [b_{i,j}] \in \mathbb{R}^{\ell_0\times\ell_1}$ and $\overline{B} = [\mathcal{A}_{\tau,\tau}(b_{i,j})] \in \mathbb{F}_2^{\ell_0(m-\tau)\times\ell_1(m-\tau)}$.

In this paper, the code with $\mathcal{T}_{\ell_0,\ell_1,m}(B)$ as the parity-check matrix has a binary codeword of size $\ell_1 \cdot (m-\tau)$, where $\ell_0 < \ell_1$. By default, the codeword is arranged in an $(m-\tau) \times \ell_1$ array of bits in column-first order, and we refer to this code as a binary array code. In addition, we refer to this code as a binary MDS array code if each codeword array can be restored by any $\ell_1 - \ell_0$ columns.

III. Definitions of Variant codes

This section defines two new classes of binary array codes (i.e., V-ETBR and V-ESIP codes). One can see that the generalized RDP codes introduced in Section II-C are a special case of the V-ESIP codes.

To begin with, we define two codes over the polynomial ring $\mathbb{R}_{p,\tau}$ as follows:

Definition 1. ***(ETBR Codes)*** *Let $2 \le r < n$, and $H = [h_{i,j}]_{0\le i<r, 0\le j<n} \in \mathbb{R}^{r\times n}$. Define ETBR$(n, r, m = p\tau, H)$ as a code over $\mathbb{R}_{p,\tau}$ determined by the parity-check matrix H that is reduced to over $\mathbb{R}_{p,\tau}$, where each element in H is modulo $f_{p,\tau}(x)$ to be an element over $\mathbb{R}_{p,\tau}$.*

$$\mathcal{T}_{r,p+r-1,p}(H') = \begin{pmatrix} I_{p-1} & I_{p-1} & I_{p-1} & \cdots & I_{p-1} & & \\ I_{p-1} & \mathcal{A}_{1,1}(x^{p-1}) & \mathcal{A}_{1,1}(x^{p-2}) & \cdots & \mathcal{A}_{1,1}(x) & I_{p-1} & \\ \vdots & \vdots & \vdots & \ddots & \vdots & & \ddots \\ I_{p-1} & \mathcal{A}_{1,1}(x^{(p-1)(r-1)}) & \mathcal{A}_{1,1}(x^{(p-2)(r-1)}) & \cdots & \mathcal{A}_{1,1}(x^{r-1}) & & I_{p-1} \end{pmatrix}. \quad (12)$$

Remark 1. *In Definition 1, the form of H is not fixed and covers H_{BR} in* (4)*, so we refer to ETBR(n, r, m, H) as an extended BR code.*

Definition 2. *(***ESIP Codes***) Let $n \geq 2, r \geq 2$, and $H' = [H|\widehat{I}] \in \mathbb{R}^{r\times(n+r-1)}$, where the definition of H is the same as in Definition 1 and $\widehat{I}$ is the matrix after removing the first column of the $r \times r$ identity matrix. Define ESIP$(n, r, m = p\tau, H')$ as a code over $\mathbb{R}_{p,\tau}$ determined by the parity-check matrix H' that is reduced to over $\mathbb{R}_{p,\tau}$, where each element in H is modulo $f_{p,\tau}(x)$ to be an element over $\mathbb{R}_{p,\tau}$.*

Remark 2. *In Definition 2, the ESIP$(n, r, m = p, H')$ is exactly the shortened IP code given by* (7) *if $H = H_{BR}$. Obviously, H' has a wider range of parameters so that the ESIP codes can be regarded as an extension of shortened IP codes.*

The variant codes corresponding to ETBR and ESIP codes, i.e., V-ETBR and V-ESIP codes, are defined below. When we refer to ETBR/ESIP codes and V-ETBR/V-ESIP codes as corresponding, it means that they are determined by the same matrix H or H' over $\mathbb{R}$.

Definition 3. *(***V-ETBR Codes***) Define V-ETBR$(n, r, m = p\tau, H)$ as a binary array code whose parity-check matrix is $\mathcal{T}_{r,n,m}(H)$, where $2 \leq r < n$, $\mathcal{T}_{r,n,m}$ is defined in* (11)*, and the definition of H is the same as that in Definition 1.*

Definition 4. *(***V-ESIP Codes***) Define V-ESIP$(n, r, m = p\tau, H')$ as a binary array code whose parity-check matrix is $\mathcal{T}_{r,n+r-1,m}(H')$, where $n \geq 2, r \geq 2$, $\mathcal{T}_{r,n+r-1,m}$ is defined in* (11)*, and the definition of H' is the same as that in Definition 2.*

Conventionally, the last r columns of the array corresponding to the codeword in the above codes are referred to as parity columns and all other columns are referred to as information columns. We have the following relationship.

Lemma 1. *Let H' in Definition 4 be determined by a Vandermonde matrix H such that $h_{1,j} = x^{p-j}$ and $h_{i,j} = h_{1,j}^i$ for $2 \leq i < r, 0 \leq j < p$, and p is a prime number. Then V-ESIP$(p, r, m = p, H')$ is exactly the generalized RDP$(p + r - 1, r)$ described in Sec. II-C.*

Proof. From Definition 4, $\mathcal{T}_{r,p+r-1,p}(H')$ is the parity-check matrix of the V-ESIP$(p, r, m = p, H')$ and is given by (12) at the top of this page, where all unspecified entries are zero.

Let $\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_{p-2} \in \mathbb{F}_2^{p-1}$ denote all $p-1$ information columns in the codewrod. We next show that any parity column generated by the V-ESIP code is the same as that in the generalized RDP code described in Sec. II-C.

Let $\mathbf{b}_{p-1}, \mathbf{b}_p, ..., \mathbf{b}_{p+r-2} \in \mathbb{F}_2^{p-1}$ denote all r parity columns of the V-ESIP code. One can easily know from (12) that $\mathbf{b}_{p-1}$ is obtained by bit-wise XORing of all $p-1$ information columns. For any $i \in [1, r)$, the i-th parity column of the V-ESIP code is obtained by

$$\mathbf{b}_{p-1+i}^{\mathrm{T}} = \sum_{j=0}^{p-1} \mathcal{A}_{1,1}(x^{(p-j)i}) \cdot \mathbf{b}_j^{\mathrm{T}} = \sum_{j=0}^{p-1} \mathcal{A}_{1,0}(x^{(p-j)i}) \cdot (\mathbf{b}_j, 0)^{\mathrm{T}}. \quad (13)$$

Note that calculating $\mathcal{A}_{1,0}(x^{(p-j)i}) \cdot (\mathbf{b}_j, 0)^{\mathrm{T}}$ is equivalent to removing the last element from the result of $\mathcal{A}_{0,0}(x^{(p-j)i}) \cdot (\mathbf{b}, 0)^{\mathrm{T}}$. Furthermore, $\mathcal{A}_{0,0}(x^{(p-j)i}) = \left(\mathcal{A}_{0,0}(x^{p-j})\right)^i$, where $\mathcal{A}_{0,0}(x^{p-j})$ can be regarded as the operator of performing j times down-cyclic shift on a vector. Assume that each data column has an imaginary bit attached at the end, thus, (13) indicates that each $\mathbf{b}_{p-1+i}, i \in [1, r)$ can be obtained by bit-wise XORing of the first p columns after each column has been subjected to down-cyclic shifts according to i times the corresponding column index size. Each parity column needs to remove the last bit in the result. The above process is consistent with the graphical representation of the generalized RDP code, as shown in Fig. 2. This completes the proof. □

Lemma 1 explicitly provides a binary parity-check matrix for the generalized RDP$(p+r-1, r)$. From the perspective of the binary parity-check matrix, all fast computations about the generalized RDP codes, such as those proposed in [21], [31], can thus be regarded as scheduling schemes for matrix operations over binary fields. Furthermore, any existing scheduling algorithm for general matrix operations over binary fields may be used to accelerate the computation of generalized RDP codes, such as [26], [27].

Recall that Theorem 1 established a connection between generalized RDP codes and shortened IP codes, which can be viewed as a special case of the connection between V-ESIP codes and ESIP codes. It remains an open problem whether there exists a general connection between V-ESIP and ESIP codes, as well as between V-ETBR and ETBR codes. The next section is devoted to these issues.

IV. Conditions that make variant codes binary MDS array codes

This section proposes the conditions that make the variant codes (see Definitions 3 and 4) binary MDS array codes by exploring the general connections between them and their counterparts over polynomial rings (see Definitions 1 and 2). TABLE I defines some important symbols to be used later.

A. Rank of the square matrix $\mathcal{T}_{\ell,\ell,m}(V)$

We first explore the the rank of $\mathcal{T}_{\ell,\ell,m}(V)$. The following lemmas are useful.

TABLE I
IMPORTANT SYMBOLS USED IN SECTION IV

Symbol	Definition
ℓ	a positive number not less than two.
V	$V=[v_{i,j}]$ is an $\ell\times\ell$ square matrix over $\mathbb{R}$.
$\mathcal{B}(i,j)$	$\mathcal{B}(i,j)=(\mathcal{A}_{\tau,0}(v_{i,0}),...,\mathcal{A}_{\tau,0}(v_{i,j-1}))$, $i\in[0,\ell), j\in[1,\ell+1)$.
$\mathcal{B}_\tau(i,j)$	$\mathcal{B}_\tau(i,j)=(\mathcal{A}_{\tau,\tau}(v_{i,0}),...,\mathcal{A}_{\tau,\tau}(v_{i,j-1}))$, $i\in[0,\ell), j\in[1,\ell+1)$.
$\overline{\mathbf{v}}_{i,j}$	a non-zero codeword with vector form generated by generator matrix $\mathcal{B}_\tau(i,j)$.
$\mathbf{v}_{i,j}$	the non-zero codeword with vector form generated by generator matrix $\mathcal{B}(i,j)$ and corresponds to $\overline{\mathbf{v}}_{i,j}$.

Lemma 2. *Assume that $\ell_0\geq 2,\ell_1\geq 2$, $x^\tau+1|a_{i,j}+a_{i,k}$ with $a_{i,j},a_{i,k}\in\mathbb{R}, i\in[0,\ell_0), j,k\in[0,\ell_1)$. Let each $\mathbf{a}_{i,j}$ denote the binary coefficient vector of $a_{i,j}$, i.e., $a_{i,j}=\mathbf{a}_{i,j}\cdot(1,x,...,x^{m-1})^{\mathrm{T}}$, and let $\overline{\mathbf{a}}_{i,j}$ denote the vector after $\mathbf{a}_{i,j}$ deletes the last τ elements. If all vectors in the set $\{(\overline{\mathbf{a}}_{i,0},...,\overline{\mathbf{a}}_{i,\ell_1-1})\in\mathbb{F}_2^{1\times(m-\tau)\ell_1}\}_{i=0}^{\ell_0-1}$ are $\mathbb{F}_2$-linearly dependent, i.e., $\sum_{i=0}^{\ell_0-1}c_i\cdot(\overline{\mathbf{a}}_{i,0},...,\overline{\mathbf{a}}_{i,\ell_1-1})=\mathbf{0}_{1\times(m-\tau)\ell_1}$, where each $c_i\in\mathbb{F}_2$ and $c_0,c_1,...,c_{\ell_0-1}$ are not all zero, then the result of $\sum_{i=0}^{\ell_0-1}c_i\cdot(\mathbf{a}_{i,0},...,\mathbf{a}_{i,\ell_1-1})$ must have the form of $(\mathbf{0}_{1\times(m-\tau)},\mathbf{u}|\mathbf{0}_{1\times(m-\tau)},\mathbf{u},|...|\mathbf{0}_{1\times(m-\tau)},\mathbf{u})$, where $\mathbf{u}\in\mathbb{F}_2^{1\times\tau}$.*

Proof. From the condition, we immediately have

$$\begin{aligned}&\sum_{i=0}^{\ell_0-1}c_i\cdot(\mathbf{a}_{i,0},...,\mathbf{a}_{i,\ell_1-1})\\ =&(\mathbf{0}_{1\times(m-\tau)},\mathbf{u}_0|\mathbf{0}_{1\times(m-\tau)},\mathbf{u}_1,|...|\mathbf{0}_{1\times(m-\tau)},\mathbf{u}_{\ell_1-1})\end{aligned}\tag{14}$$

where each $\mathbf{u}_i\in\mathbb{F}_2^{1\times\tau}$. In the above formula, the sum of any two parts is $(\mathbf{0}_{1\times(m-\tau)},\mathbf{u}_j+\mathbf{u}_k)=\sum_{i=0}^{\ell_0-1}c_i\cdot(\mathbf{a}_{i,j}+\mathbf{a}_{i,k})$, where $0\leq j<k<\ell_1$. Since $x^\tau+1|a_{i,j}+a_{i,k}$, then the sum $(\mathbf{0}_{1\times(m-\tau)},\mathbf{u}_j+\mathbf{u}_k)$ is a binary coefficient vector of the polynomial that is a multiple of $x^\tau+1$. However, $\mathbf{u}_j+\mathbf{u}_k$ contains only τ elements. This results in $\mathbf{u}_j+\mathbf{u}_k$ having to be a zero vector. Therefore, we have $\mathbf{u}_j=\mathbf{u}_k$ with $j\neq k$. This completes the proof. □

Remark 3. *From the proof of Lemma 2, one can readily know that $\mathbf{u}=\mathbf{0}_{1\times\tau}$ in Lemma 2, if $x^\tau+1|a_{i,j}, i\in[0,\ell_0), j\in[0,\ell_1)$.*

Lemma 3. *The square matrix $\mathcal{T}_{\ell,\ell,m}(V)$ has full rank if the following conditions are satisfied:*

1) *V has full rank over $\mathbb{R}_{p,\tau}$, i.e., $\gcd(|V|,f_{p,\tau}(x))=1$, where $|V|$ is the determinant of V.*
2) *For any $0\leq i<\ell$, then $\mathcal{B}_\tau(i,\ell)$ in TABLE I has full row rank over $\mathbb{F}_2$.*
3) *For any $0\leq i<\ell, 0\leq j<\ell$, then $x^\tau+1|v_{i,j}$.*

When $v_{0,j}=1,\forall j\in[0,\ell)$, 3) is relaxed to

3') For any $1\leq i<\ell, 0\leq j<k<\ell$, then $x^\tau+1|v_{i,j}+v_{i,k}$.

Proof. According to TABLE I, $\mathcal{T}_{\ell,\ell,m}(V)$ is composed of $\mathcal{B}_\tau(0,\ell),\mathcal{B}_\tau(1,\ell),...,\mathcal{B}_\tau(\ell-1,\ell)$. Since each $\mathcal{B}_\tau(i,\ell), i\in[0,\ell)$, has full row rank, we only need to prove that there is no $\overline{\mathbf{v}}_{0,\ell},\overline{\mathbf{v}}_{1,\ell},...,\overline{\mathbf{v}}_{\ell-1,\ell}$, which are $\mathbb{F}_2$-linearly dependent. By contradiction, assume that there exists $\overline{\mathbf{v}}_{0,\ell},\overline{\mathbf{v}}_{1,\ell},...,\overline{\mathbf{v}}_{\ell-1,\ell}$ such that they are $\mathbb{F}_2$-linearly dependent, i.e., $\sum_{i=0}^{\ell-1}c_i\overline{\mathbf{v}}_{i,\ell}=\mathbf{0}_{1\times(m-\tau)\ell}$ where each $c_i\in\mathbb{F}_2$ and $c_0,c_1,...,c_{\ell-1}$ are not all zero.

We first consider the third condition of $x^\tau+1|v_{i,j}$. According to Remark 3 and the facts that $x^\tau+1|v_{i,j}$, each $\mathbf{v}_{i,\ell}$ is the vector consisting of the binary coefficient vectors in $q_{i,\ell}\cdot(v_{i,0},v_{i,1},...,v_{i,\ell-1})$, where $q_{i,\ell}\in\mathbb{R}\setminus\{0\}$ and $\deg(q_{i,\ell})<m-\tau$, then $\sum_{i=0}^{\ell-1}c_i\mathbf{v}_{i,\ell}=(\mathbf{0}_{1\times m},...,\mathbf{0}_{1\times m})$. Therefore, we have $\sum_{i=0}^{\ell-1}c_iq_{i,\ell}\cdot v_{i,j}=0 \mod x^m+1$ for $j\in[0,\ell)$. By taking $j=0,1,...,\ell-1$, the above equations can be converted into

$$\Gamma_0\cdot\begin{pmatrix}c_0\cdot q_{0,\ell}\\ c_1\cdot q_{1,\ell}\\ \vdots\\ c_{\ell-1}\cdot q_{\ell-1,\ell}\end{pmatrix}=\mathbf{0}^{\mathrm{T}},\tag{15}$$

where $\mathbf{0}$ is a zero-row vector and

$$\Gamma_0=\begin{pmatrix}v_{0,0}&v_{1,0}&\cdots&v_{\ell-1,0}\\ v_{0,1}&v_{1,1}&\cdots&v_{\ell-1,1}\\ \vdots&\vdots&\ddots&\vdots\\ v_{0,\ell-1}&v_{1,\ell-1}&\cdots&v_{\ell-1,\ell-1}\end{pmatrix}.\tag{16}$$

In (15), all operations are performed in $\mathbb{R}$. Note that each $c_i\in\mathbb{F}_2$ and $\deg(q_{i,\ell})<m-\tau$, we can solve the above linear equations in $\mathbb{R}_{p,\tau}$. Since $|\Gamma_0|=|V|$ is invertible over $\mathbb{R}_{p,\tau}$, then $c_0q_{0,\ell},...,c_{\ell-1}q_{\ell-1,\ell}$ in (15) must all be zero according to Cramer's rule. Moreover, each $q_{i,\ell}\neq 0$ with $\deg(q_{i,\ell})<m-\tau$, so that $c_0=c_1=\cdots=c_{\ell-1}=0$. This contradicts the assumption at the beginning.

Consider the third condition of $x^\tau+1|v_{i,j}+v_{i,k}$ instead, then Lemma 3 gives $\sum_{i=0}^{\ell-1}c_i\mathbf{v}_{i,\ell}=(\mathbf{0}_{1\times(m-\tau)},\mathbf{u}|...|\mathbf{0}_{1\times(m-\tau)},\mathbf{u})$, where $\mathbf{u}\in\mathbb{F}_2^{m-\tau}$. Since $v_{0,j}=1,\forall j\in[0,\ell)$, we have

$$\begin{aligned}&\sum_{i=1}^{\ell-1}c_i\mathbf{v}_{i,\ell}\\ =&c_0\cdot\mathbf{v}_{0,\ell}+(\mathbf{0}_{1\times(m-\tau)},\mathbf{u}|...|\mathbf{0}_{1\times(m-\tau)},\mathbf{u})=(\mathbf{u}'|...|\mathbf{u}'),\end{aligned}\tag{17}$$

where $\mathbf{u}'\in\mathbb{F}_2^m$. Based on the fact that any two part of the form in (17) sum to zero, we have $\sum_{i=1}^{\ell-1}c_iq_{i,\ell}\cdot(v_{i,j}+v_{i,k})=0 \mod x^m+1$ for $0\leq j<k<\ell$. By taking $(j,k)=(0,1),(0,2),...,(0,\ell-1)$, the above equations can be converted into

$$\Gamma_1\cdot\begin{pmatrix}c_1\cdot q_{1,\ell}\\ c_2\cdot q_{2,\ell}\\ \vdots\\ c_{\ell-1}\cdot q_{\ell-1,\ell}\end{pmatrix}=\mathbf{0}^{\mathrm{T}},\tag{18}$$

where $\mathbf{0}$ is a zero-row vector and

$$\Gamma_1 = \begin{pmatrix} v_{1,0}+v_{1,1} & \cdots & v_{\ell-1,0}+v_{\ell-1,1} \\ v_{1,0}+v_{1,2} & \cdots & v_{\ell-1,0}+v_{\ell-1,2} \\ \vdots & \ddots & \vdots \\ v_{1,0}+v_{1,\ell-1} & \cdots & v_{\ell-1,0}+v_{\ell-1,\ell-1} \end{pmatrix}. \quad (19)$$

In (18), all operations are performed in $\mathbb{R}$. Note that each $c_i \in \mathbb{F}_2$ and $\deg(q_{i,\ell}) < m-\tau$, we can solve the above linear equations in $\mathbb{R}_{p,\tau}$. One can know that the determinant of Γ_1 is equal

$$\begin{aligned} &\begin{vmatrix} 1 & 0 & \cdots & 0 \\ 1 & v_{1,0}+v_{1,1} & \cdots & v_{\ell-1,0}+v_{\ell-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & v_{1,0}+v_{1,\ell-1} & \cdots & v_{\ell-1,0}+v_{\ell-1,\ell-1} \end{vmatrix} \quad (20) \\ =&\begin{vmatrix} 1 & v_{1,0} & \cdots & v_{\ell-1,0} \\ 1 & v_{1,1} & \cdots & v_{\ell-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & v_{1,\ell-1} & \cdots & v_{\ell-1,\ell-1} \end{vmatrix} = |V|, \end{aligned}$$

where the first equality is obtained by subtracting the appropriate multiple of the first column from all other columns. Thus, $|\Gamma_1| = |V|$ is also invertible over $\mathbb{R}_{p,\tau}$. Then $c_1 q_{1,\ell}, ..., c_{\ell-1} q_{\ell-1,\ell}$ in (15) must all be zero according to Cramer's rule. Similarly, note that each $q_{i,\ell} \neq 0$ with $\deg(q_{i,\ell}) < m-\tau$, so we must have that $c_1 = \cdots = c_{\ell-1} = 0$, then $c_0 = 0$. This contradicts the assumption at the beginning. This completes the proof. □

The above lemma reveals the connection between the ranks of V and $\mathcal{T}_{\ell,\ell,m}(V)$. In Lemma 3, the latter two conditions are easily met. More precisely, the third condition only requires that any $v_{i,j}$ is a multiple of $x^\tau+1$, and the second condition can be satisfied by the following lemma, which is easily obtained through the proof of Proposition 6 in [29].

Lemma 4. *([29]) Let $a, b \in \mathbb{R}$, then $\mathcal{A}_{\tau,\tau}(a)$ has full row rank over $\mathbb{F}_2$ if $\gcd(a, x^m+1) = x^\tau+1$; $\mathcal{A}_{\tau,\tau}(a,b)$ has full row rank over $\mathbb{F}_2$ if $\gcd(a+b, x^m+1) = x^\tau+1$.*

B. Conditions for binary MDS array codes

We now present the conditions that make the variant codes binary MDS array codes, by establishing the connections between them and the corresponding codes over polynomial rings. To begin with, the following theorem on V-ETBR codes can be obtained.

Theorem 2. *V-ETBR$(n, r, m = p\tau, H)$ is a binary MDS array code if*

1) *The corresponding ETBR(n, r, m, H) is an MDS code over $\mathbb{R}_{p,\tau}$.*
2) *For any $0 \le i < r, 0 \le j < n$, then $\gcd(h_{i,j}, x^m+1) = x^\tau+1$.*

When $h_{0,j} = 1, \forall j \in [0, n)$, the above last condition is replaced with

2') *For any $1 \le i < r, 0 \le j, k < n$ and $j \neq k$, then $\gcd(h_{i,j}, x^m+1) = x^\tau+1$ or $\gcd(h_{i,j}+h_{i,k}, x^m+1) = x^\tau+1$.*

Proof. We only need to prove that any $\mathcal{T}_{\ell\times\ell}(V)$ for $\ell = r$ has full rank, where all elements in V are determined by H. This is easily derived from Lemmas 3 and 4. □

The following theorems on V-ESIP codes can be obtained.

Theorem 3. *When the first row of H in H' is an all-one row, the V-ESIP$(n, r, m = p\tau, H')$ is a binary MDS array code if*

1) *The corresponding ESIP(n, r, m, H') is an MDS code over $\mathbb{R}_{p,\tau}$.*
2) *For any $1 \le i < r$ and $0 \le j < k < n$, then $x^\tau+1 | h_{i,j}+h_{i,k}$.*

Proof. Without loss of generality, we only need to prove $\mathcal{T}_{\ell,\ell,m}(V)$ for any $1 < \ell \le r$ has full rank, where elements in $\mathcal{T}_{\ell,\ell,m}(V)$ are determined by H' and $\{v_{0,j} = 1\}_{j=0}^{\ell-1}$. We prove this via Lemma 3. First, the first condition of Lemma 3 is satisfied since the corresponding ESIP(n, r, m, H') is an MDS code over $\mathbb{R}_{p,\tau}$. Furthermore, the fact that V with $\ell = 2$ have full rank over $\mathbb{R}_{p,\tau}$ leads to $\gcd(h_{i,j}+h_{i,k}, f_{p,\tau}(x)) = 1, 1 \le i < r, 0 \le j < k < n$. Recall that the condition of $x^\tau+1 | h_{i,j}+h_{i,k}$, then we have $\gcd(h_{i,j}+h_{i,k}, x^m+1) = x^\tau+1$. One can easily see from Lemma 4 that the second condition of Lemma 3 is thus satisfied. The latter third condition of Lemma 3 is obviously satisfied. This completes the proof. □

Remark 4. *Now, the correctness of Theorem 1 can be readily proven by Theorem 3, just by setting $\tau = 1$. Theorem 1 requires p to be an odd prime number for shortened IP codes. Theorem 3 provides additional clarification by demonstrating that p only needs to be odd.*

In Theorem 3, the rightmost end of H' is not necessarily an identity matrix. Since the existence of an identity matrix can simplify encoding, we consider the following case that does not require the first row of H to be an all-one row (only the last column to be constrained).

Theorem 4. *When the rightmost end of H' is an $r \times r$ identity matrix, i.e., the last column of H is $(1, 0, 0, ..., 0)^{\mathrm{T}}$, then the V-ESIP$(n, r, m = p\tau, H')$ is a binary MDS array code if*

1) *The corresponding ESIP(n, r, m, H') is an MDS code over $\mathbb{R}_{p,\tau}$.*
2) *For any $0 \le i < r$ and $0 \le j < n-1$, then $\gcd(h_{i,j}, x^m+1) = x^\tau+1$.*

Proof. Without loss of generality, we only need to prove $\mathcal{T}_{\ell,\ell,m}(V)$ for any $1 \le \ell \le r$ has full rank, where elements in $\mathcal{T}_{\ell,\ell,m}(V)$ are determined by H after removing the last column. Similarly, we prove this via Lemma 3. First, the first condition of Lemma 3 is satisfied since the corresponding ESIP(n, r, m, H') is an MDS code over $\mathbb{R}_{p,\tau}$. Lemma 4 and $\gcd(h_{i,j}, x^m+1) = x^\tau+1$ lead to that the second condition of Lemma 3 holds. Finally, the former third condition of Lemma 3 obviously holds. This completes the proof. □

V. Explicit constructions & fast computations

Based on the conditions given in Sec. IV-B, we next present some explicit constructions for the V-ETBR/V-ESIP binary MDS array codes. In particular, Vandermonde matrices and

Cauchy matrices are two classes of matrices commonly used in the construction of MDS codes. They both have a regular structure, and their determinants can be easily calculated. By setting appropriate entries, one can easily make sub-matrices of the Cauchy-based/Vandermonde-based parity-check matrix having full rank. For more details, please refer to [1], [23], [32]. This paper also explores the use of the two matrices in our constructions.

A. Constructions

To begin with, suppose that $f_{p,1}(x)$ in (2) can be completely factorized into $f_{p,1}(x) = f_0(x) \cdot f_1(x) \cdots f_{\mu-1}(x)$, where each $f_i(x)$ is an irreducible polynomial over $\mathbb{F}_2[x]$ and $\lambda = deg(f_0(x)) \leq deg(f_1(x)) \leq \cdots \leq deg(f_{\mu-1}(x))$. Note that $\lambda = p-1$ if 2 is a primitive element in p-ary finite field $\mathbb{F}_p$ [5]. Then, we have the following construction for the V-ESIP MDS array codes with any number of parity columns, based on Cauchy matrices.

Construction 1. *(***V-ESIP MDS** *array codes with $r \geq 2$) Let $\{a_0, ..., a_{r-1}\}$ and $\{b_0, ..., b_{n-2}\}$ are two sets of elements from $\mathbb{R}$, where $\deg(a_i) < \lambda, \deg(b_j) < \lambda$ and $a_i \neq b_j$ for any i, j, then the V-ESIP$(n, r \geq 2, m = p\tau, H' = [H_I | I_{r\times r}])$ is a binary MDS array code, where $H_I = [(x^\tau + 1) \cdot g_{i,j}] \in \mathbb{R}^{r\times(n-1)}$ and $g_{i,j}$ denotes the inverse of $a_i + b_j$ over $\mathbb{R}_{p,\tau}$ that always exists due to the degree of $a_i + b_j$ less than λ.*

Proof. We prove this via Theorem 4. Let $H'_I = [g_{i,j} = \frac{1}{a_i+b_j}] \in \mathbb{R}_{p,\tau}^{r\times(n-1)}$ that is a Cauchy matrix. Obviously, the determinant of any square sub-matrix of H'_I is invertible over $\mathbb{R}_{p,\tau}$, since it is the product of some elements in the sets $\{a_i + a_j\}_{i\neq j}, \{b_i + b_j\}_{i\neq j}, \{\frac{1}{a_i+b_j}\}$ [32], where any a_i, b_j has the degree less than λ. Note that the determinant of the corresponding square sub-matrix of H_I is $(x^\tau+1)^\xi$ times the above result for some $\xi \geq 1$, and $\gcd(x^\tau + 1, f_{p,\tau}(x)) = 1$. This results in the determinant of any square sub-matrix of H_I having to be invertible over $\mathbb{R}_{p,\tau}$. Thus, the first condition of Theorem 4 is satisfied. Furthermore, we have $\gcd((x^\tau + 1)g_{i,j}, f_{p,\tau}(x)) = 1$, leading to the second condition of Theorem 4 holds. This completes the proof. □

Remark 5. *To our knowledge, many works on MDS codes seek efficient computation by mapping Cauchy-based parity-check matrices over finite fields to binary matrices [32]–[35]. This enables the use of scheduling algorithms of binary matrix-vector multiplication, reducing the number of operations. Construction 1 changes finite fields to polynomial rings and also provides a binary mapping. Existing scheduling algorithms of binary matrix-vector multiplication may be applicable to the variant codes in Construction 1, such as those proposed in [33]–[35]. Notably, our mapping is more convenient than the previous one to design and analyze the number of 1s in the resulting matrix after mapping, as it is obtained through circulant matrices, while the other is by taking the modulus of an irreducible polynomial. The new mapping offers a new idea for developing efficient scheduling algorithms for Cauchy-based codes.*

Based on Vandermonde matrices, Construction 2 provides the construction of the V-ETBR MDS array codes with any number of parity columns. This construction can also be found in [29], but a different proof is provided. Since the proof is based on Theorem 2, which reveals the general connection between variant codes and codes over polynomial rings (not specified in [29]), we only need to check whether the associated matrices over polynomial rings satisfy two simple conditions. This results in a proof process that is more concise and efficient compared to the one in [29] (which focuses directly on binary parity-check matrices). In addition, this proof shows the wide applicability of Theorem 2. From the proof of any construction proposed in this paper, one can easily see that the codes over $\mathbb{R}_{p,\tau}$ corresponding to the variant codes are also MDS codes.

To simplify the representation of elements in the Vandermonde matrix H, we let $h_{0,i} = 1, \forall i \in [0, n)$, and $h_i := h_{1,i}, \forall i \in [0, n)$, such that $h_{j,i} = h_i^j, \forall j \in [1, r), i \in [0, n)$.

Construction 2. *(***V-ETBR MDS** *array codes with $r \geq 2$) Let $H \in \mathbb{R}^{r\times n}$ be a Vandermonde matrix, $n = 2^{n_0}, n_0 \leq \lambda$, and $h_i = (1 + x^\tau) \cdot h'_i, \forall i \in [0, n)$, where $\{h'_i\}_{0\leq i<n}$ is given by $h'_0 = 0$ and $h'_{i+2^j} = h'_i + x^j, 0 \leq j < n_0, 0 \leq i < 2^j$. Then, the V-ETBR$(n, 2 \leq r < n, m = p\tau, H)$ is a binary MDS array code.*

Proof. We prove this via Theorem 2. Since the degree of any h'_i is less than λ, we have $gcd(h_i^j, x^m + 1) = (x^\tau + 1) \cdot gcd((h'_i)^j, f_{p,1}^\tau(x)) = x^\tau + 1$, where $1 \leq i < r$ and $0 \leq j < k < n$. This results in that the latter second condition of Theorem 2 holds. For the first condition of Theorem 2, we have $gcd(h_j + h_k, f_{p,\tau}(x)) = gcd(h'_j + h'_k, f_{p,1}^\tau(x)) = 1$, where $0 \leq j < k < n$. Then any $r \times r$ Vandermonde sub-matrix of H is invertible over $\mathbb{R}_{p,\tau}$, leading to the ETBR$(n, r, m = p\tau, H)$ being MDS code over $\mathbb{R}_{p,\tau}$. This completes the proof. □

Remark 6. *In [29], the authors provided a fast scheduling scheme for the syndrome computation of Construction 2 with $2 \leq r \leq 3$. The next subsection (i.e., Section V-B1) will propose its generalization to be suitable for any $r \geq 2$.*

According to Theorem 3, it is not difficult to check that the V-ESIP$(n, r = 3, m = p\tau, H')$ is a binary MDS array code if H' has the same H as Construction 2. The following provides the Vandermonde-based construction for the V-ESIP MDS array code with $r = 4$.

Construction 3. *(***V-ESIP MDS** *array codes with $r = 4$) Let $H \in \mathbb{R}^{r\times n}$ be a Vandermonde matrix, $n = 2^{n_1} + 1, n_1 \leq w = \lfloor \frac{\lambda-1}{2} \rfloor$, $h_{n-1} = 0$, and $h_i = (h'_i + x^w) \cdot (1 + x^\tau)$, where $i \in [0, 2^{n_1})$ and $\{h'_i\}_{0\leq i<2^{n_1}}$ is given by $h'_0 = 0, h'_{i+2^j} = h'_i + x^j, 0 \leq j < n_1, 0 \leq i < 2^j$. Then, the V-ESIP$(n, r = 4, m = p\tau, H')$ is a systematic binary MDS array code.*

Proof. We prove this via Theorem 3. The second condition in Theorem 3 obviously holds. For the first condition in Theorem 3, we only need to prove that any 4×4 sub-matrix of H' is invertible over $\mathbb{R}_{p,\tau}$. Specifically, we first consider any 4×4 sub-matrix of H in H', which is a Vandermonde square matrix. Clearly, for any $0 \leq i < j < n-1$, we have that $gcd(h_i + h_j, f_{p,\tau}(x)) = gcd(h'_i + h'_j, f_{p,1}^\tau(x))$ and

$gcd(h_i + h_{n-1}, f_{p,1}^{\tau}(x)) = gcd(h_i' + x^w, f_{p,1}^{\tau}(x))$. Since each $deg(h_i') < w < \lambda$, then $gcd(h_i + h_j, f_{p,\tau}(x)) = 1, \forall 0 \leq i < j < n$. This indicates that any 4×4 sub-matrix of H is invertible over $\mathbb{R}_{p,\tau}$. Next, we focus on the remaining cases. We only need to determine if the following matrices are invertible over $\mathbb{R}_{p,\tau}$,

$$\begin{pmatrix} 1 & 1 \\ h_i^3 & h_j^3 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 \\ h_i & h_j & h_k \\ h_i^3 & h_j^3 & h_k^3 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 \\ h_i^2 & h_j^2 & h_k^2 \\ h_i^3 & h_j^3 & h_k^3 \end{pmatrix}, \quad (21)$$

where $0 \leq i < j < k < n$. According to generalized Vandermonde determinants [36], the determinants of the above three matrices are respectively (let $h_{n-1}' = 0$)

$$\begin{aligned} &h_i^3 + h_j^3 = (h_i + h_j)(1 + x^{\tau})^2 \cdot \left((h_i')^2 + h_i' h_j' + (h_j')^2 \right. \\ &\qquad \left. + (h_i' + h_j')x^w + x^{2w}\right), \\ &h_i + h_j + h_k = (1 + x^{\tau})(h_i' + h_j' + h_k' + x^w), \\ &h_i h_j + h_i h_k + h_j h_k = (1 + x^{\tau})^2 \\ &\qquad \cdot (h_i' h_j' + h_i' h_k' + h_j' h_k' + x^{2w}), \end{aligned} \quad (22)$$

where $0 \leq i < j < k < n$. Since all h_i', h_j', h_k' have degrees less than w, the above three values are not zero. Furthermore, due to $2w < \lambda$, they are all coprime with $f_{p,\tau}(x)$. Then all the matrices in (21) are invertible over $\mathbb{R}_{p,\tau}$. This completes the proof. □

Remark 7. *It is clear that all the codes in Construction 1, 2, and 3 allow the total number of data columns to reach the exponential size with respect to the design parameter p. This is suitable for the needs of large-scale storage systems [29]. In addition, it is possible to construct the new codes using other matrices, such as Moore matrices [37] and some matrices searched by computers. All proposed conditions in Section IV-B offer great flexibility in constructing the variant codes.*

B. Fast Computations

To begin with, one can know from coding theory that the product of any parity-check matrix and its corresponding codeword is zero [38]. Formally, $\mathbf{0}^{\mathrm{T}} = \widehat{H} \cdot \widehat{\mathbf{x}}^{\mathrm{T}}$, where $\mathbf{0}$ denotes a zero vector, $\widehat{H}$ denotes a binary parity-check matrix, and $\widehat{\mathbf{x}}$ denotes the corresponding codeword. It follows that $\widehat{H} \cdot \mathbf{x}^{\mathrm{T}} = \widehat{H}_e \cdot \mathbf{e}^{\mathrm{T}}$, where $\mathbf{x}$ denotes the codeword after all erased symbols are set to zero, $\mathbf{e}$ denotes the vector consisting of all erased symbols, and $\widehat{H}_e$ denotes the sub-matrix of $\widehat{H}$ corresponding to $\mathbf{e}$. The above leads to the following common framework for encoding and decoding procedures [1], [39]: (when encoding, all parity symbols can be regarded as erased symbols.)

Step 1. Compute syndrome $\mathbf{s}^{\mathrm{T}} := \widehat{H} \cdot \mathbf{x}^{\mathrm{T}}$.
Step 2. Solve linear equations $\mathbf{s}^{\mathrm{T}} = \widehat{H}_e \cdot \mathbf{e}^{\mathrm{T}}$.

Note that in Step 2, $\mathbf{e}$ can be calculated by $\mathbf{e}^{\mathrm{T}} = \widehat{H}_e^{-1} \cdot \mathbf{s}^{\mathrm{T}}$. In practice, each storage node holds a massive amount of data. Once all erased nodes (from power outages, downtime, etc.) are identified, the inverse of $\hat{H}_e$ needs to be computed only once to recover all data stored in erased nodes. This results in the computational complexity of Step 2 being dominated by matrix-vector multiplication, which requires at most $c \cdot r^2(m-\tau)^2$ XOR,[1] where c is a very large constant determined by the capacity of storage nodes. If r, τ are constants and $p = \Theta(\lg n)$, we have $\lim_{n\to\infty} \frac{c \cdot r^2 (m-\tau)^2}{c \cdot (m-\tau) n} = 0$, where $m = p\tau$. This means that the asymptotic computational complexity of encoding/decoding is dominated by syndrome computation. This subsection proposes fast syndrome computations for the constructed Vandermonde-based variant codes.

1) Syndrome computation for Construction 2: Here, $\widehat{H} = \mathcal{T}_{r,n,m}(H)$, then the syndrome computation is $\mathbf{s}^{\mathrm{T}} = \mathcal{T}_{r,n,m}(H) \cdot \mathbf{x}^{\mathrm{T}}$. Let $\mathbf{x} = (\mathbf{x}_0, ..., \mathbf{x}_{n-1})$ and $\mathbf{s} = (\mathbf{s}_0, ..., \mathbf{s}_{r-1})$ with each $\mathbf{x}_i \in \mathbb{F}_2^{m-\tau}, \mathbf{s}_i \in \mathbb{F}_2^{m-\tau}$. For any $i \in [0, r)$, we have

$$\begin{aligned} \mathbf{s}_i^{\mathrm{T}} &= \sum_{j=0}^{2^{n_0}-1} \mathcal{A}_{\tau,\tau}(h_j^i) \cdot \mathbf{x}_j^{\mathrm{T}} \\ &= \sum_{j=0}^{2^{n_0}-1} \mathcal{A}_{\tau,\tau}\left((h_j')^i \cdot (1 + x^{\tau})^i\right) \cdot \mathbf{x}_j^{\mathrm{T}}. \end{aligned} \quad (23)$$

The following is dedicated to demonstrating that $\mathbf{s}$ can be calculated with the asymptotic complexity of $\lfloor \lg r \rfloor + 1$ XORs per data bit as n_0 increases.

We first focus on the auxiliary calculation, i.e., $(\mathbf{s}_i^*)^{\mathrm{T}} = \sum_{j=0}^{2^{n_0}-1} \mathcal{A}_{0,0}\left((h_j')^i \cdot (1 + x^{\tau})^i\right) \cdot (\mathbf{x}_j^*)^{\mathrm{T}}$, where $i \in [0, r)$, $\mathbf{s}_i^* \in \mathbb{F}_2^m$ and $\mathbf{x}_i^* = (\mathbf{x}_i, 0, 0, \cdots, 0) \in \mathbb{F}_2^m$. Obviously, for any $i \in [0, r)$, the first $m - \tau$ symbols in $\mathbf{s}_i^*$ exactly form $\mathbf{s}_i$. Note that the auxiliary calculation can be converted into

$$(\mathbf{s}_i^*)^{\mathrm{T}} = \mathcal{A}_{0,0}\left((1 + x^{\tau})^i\right) \cdot \sum_{j=0}^{2^{n_0}-1} \mathcal{A}_{0,0}\left((h_j')^i\right) \cdot (\mathbf{x}_j^*)^{\mathrm{T}}, \quad (24)$$

since $\mathcal{A}_{0,0}$ is an isomorphic mapping. In the above formula, the result of multiplying $\mathcal{A}_{0,0}\left((h_j')^i\right)$ by $(\mathbf{x}_j^*)^{\mathrm{T}}$ is in fact the reverse coefficient vector of the resultant polynomial from multiplying $(h_j')^i$ by

$$\mathbf{x}_j^*(x) := \mathbf{x}_j^* \cdot (x^{m-1}, ..., x, 1)^{\mathrm{T}}. \quad (25)$$

Hence, (24) can be easily obtained after calculating the following polynomial multiplication

$$P(i, \{\mathbf{x}_j^*(x)\}_{j=0}^{2^{n_0}-1}) := \sum_{j=0}^{2^{n_0}-1} (h_j')^i \cdot \mathbf{x}_j^*(x), \quad i \in [0, r). \quad (26)$$

It can be seen from the setting of $\{h_j'\}_{j=0}^{2^{n_0}-1}$ that the calculation in (26) is similar to the syndrome computation in [1]. The only difference is that the calculation is performed in the polynomial ring $\mathbb{R}$, while [1] is in a binary extension field. Fast syndrome computation in [1] can be easily extended to the case of polynomial rings. From [1], we have the following lemma for computing (26).

Lemma 5. *Let $\mathbf{y}^{\mathrm{T}} = (\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_{2^{n_0}-1})^{\mathrm{T}} = R_{n_0} \cdot \left(\mathbf{x}_0^*(x), ..., \mathbf{x}_{2^{n_0}-1}^*(x)\right)^{\mathrm{T}}$, where each $\mathbf{y}_0 \in \mathbb{R}$, R_{n_0} is a Reed-Muller matrix defined by $R_0 = (1)$ and*

$$R_{i+1} = \begin{pmatrix} R_i & R_i \\ \mathbf{0}_i & R_i \end{pmatrix}, \quad (27)$$

[1] In fact, this computational complexity can be reduced by scheduling algorithms for matrix-vector multiplication in the binary field, such as "four Russians" algorithm [40] or other heuristic algorithms in [26], [27].

TABLE II
COMPUTATIONAL COMPLEXITIES OF SYNDROME COMPUTATIONS IN THE VANDERMONDE-BASED VARIANT CODES (# OF XORS PER DATA BIT)

Configurations								
p	11 ($\lambda=10$)			13($\lambda=12$)			17($\lambda=8$)	Theoretical
n_0 or n_1	8	9	10	8	9	10	8	Results
V-ETBR($n=2^{n_0}, r, m=p, H$)								
$r=3$	2.026	2.015	2.008	2.027	2.015	2.008	2.028	2
$r=4$	3.112	3.070	3.043	3.117	3.073	3.045	3.123	3
$r=5$	3.145	3.088	3.053	3.150	3.091	3.055	3.156	3
$r=6$	3.376	3.234	3.143	3.384	3.240	3.146	3.395	3
$r=7$	3.607	3.380	3.232	3.619	3.387	3.237	3.635	3
$r=8$	5.795	5.223	4.807	5.874	5.283	4.848	5.995	4
V-ESIP($n=2^{n_1}+1, r=4, m=p, H'$)								
$r=4$	3.118	3.073	3.044	3.191	3.075	3.046	3.126	3

where $i \in \mathbb{N}$ and $\mathbf{0}_i$ denotes the $2^i \times 2^i$ all-zero matrix. Then for any $i \in [0, r)$,

$$P(i, \{\mathbf{x}_j^*(x)\}_{j=0}^{2^{n_0}-1}) = \begin{cases} \mathbf{y}_0, & \text{if } b(i)=0, \\ \sum_{j=0}^{n_0-1} x^{ij} \cdot \mathbf{y}_{2^j} + \sum_{\substack{0<j<2^{n_0} \\ 1<b(j)\le b(i)}} f(i,j)\cdot \mathbf{y}_j, & \text{if } b(i)\ge 1, \end{cases} \tag{28}$$

where $b(i)$ is the number of 1s in the binary representation of i, and $f(i,j)$ is a function that depends only on the indices i and j. In particular, when $b(i)=2$, each $f(i,j)$ in (28) is a polynomial containing two terms.

Proof. The proof can be easily obtained by analogy with that in [1]. □

From the above, the syndrome computation in (23) can be completed through the following steps (given m, r and n_0):

Step 1. From the input vector $(\mathbf{x}_0^*(x), ..., \mathbf{x}_{2^{n_0}-1}^*(x))$, calculate all required $\mathbf{y}_i$ in (28).
Step 2. From (28), calculate $\{P(i, \{\mathbf{x}_j^*(x)\}_{j=0}^{2^{n_0}-1})\}_{i=0}^{r-1}$.
Step 3. Calculate $\{\mathbf{s}_i^*\}_{i=0}^{r-1}$ according to (24), and then extract $\{\mathbf{s}_i\}_{i=0}^{r-1}$.

In Step 1, many operations involving zeros can be eliminated, as each $\mathbf{x}_i^*$ is obtained by filling zeros with $\mathbf{x}_i$. In Step 2, if $r<8$, all involved multiplications can be calculated using at most one vector addition and one circular shift. This is due to the fact that each multiplication factor is a polynomial containing no more than two terms. If $r \ge 8$, it is best to use matrix-vector multiplication for this operation (the multiplication of two polynomials over $\mathbb{R}$ can be converted into multiplying a circulant matrix by a coefficient vector of a polynomial). This is because $f(i,j)$ in (28) contains too many terms that need to be summed. In contrast, when implemented using matrix-vector multiplication, there exist general scheduling algorithms that can reduce the computational complexity. In Step 3, the involved two operations can be merged into

$$\mathbf{s}_i^{\mathrm{T}} = \mathcal{A}_{\tau,0}\left((1+x^\tau)^i\right) \cdot \sum_{j=0}^{2^{n_0}-1} \mathcal{A}_{0,0}\left((h_j')^i\right) \cdot (\mathbf{x}_j^*)^{\mathrm{T}}, \tag{29}$$

where $i \in [0, r)$.

In terms of complexity, Step 1 requires only a portion of the RM transform, and one can know from [1] that it produces XORs with the number of $(m-\tau)\cdot((\lfloor \lg r \rfloor + 1)n + o(n))$ [1], where little-o notation is used to describe an upper bound that cannot be tight. Step 2 produces multiplications and additions that are both $\sum_{i=1}^{r-1}\sum_{t=1}^{b(i)}\binom{n_0}{t} - r + 1$. When r is a constant, it is not difficult to check that $\lim_{n_0\to\infty} \frac{\sum_{i=1}^{r-1}\sum_{t=1}^{b(i)}\binom{n_0}{t}}{2^{n_0}/n_0} = 0$. Thus, the total number of XORs required for Step 2 is $m^2 \cdot o(2^{n_0}/n_0)$. Step 3 produces $r-1$ matrix-vector multiplications. In summary, when r and τ are constants and $n = 2^{n_0}$ approaches infinity, the asymptotic complexity of the above syndrome computation is $\lfloor \lg r \rfloor + 1$ XORs per data bit. Note that $m = p\tau$ and $p = \Theta(n_0)$, where big-Θ notation is used to describe a bound within a constant factor. For visualization, TABLE II lists the computational complexities required for the proposed syndrome computation with different parameters. It can be observed that the numerical results are close to the theoretical ones, especially when n_0 is large enough. Indeed, the syndrome computation proposed in [29], which reaches an asymptotic complexity of two XORs per data bit, is a special case of the above scheme at $r=3$.

2) Syndrome computation for Construction 3: Here, let $\mathbf{x} = (\mathbf{x}_0, ..., \mathbf{x}_{n+3})$ of each $\mathbf{x}_i \in \mathbb{F}_2^{m-\tau}$ be a codeword, and $\mathbf{s} = (\mathbf{s}_0, ..., \mathbf{s}_3)$ of each $\mathbf{s}_i \in \mathbb{F}_2^{m-\tau}$ the corresponding syndrome. Note that in Construction 3, $n = 2^{n_1}+1$ and the parity-check matrix $\mathcal{T}_{r,n,m}(H')$ is systematic. For any $i \in [0, 4)$, we have

$$\begin{aligned} \mathbf{s}_i^{\mathrm{T}} &= \mathbf{x}_{2^{n_1}+i}^{\mathrm{T}} + \sum_{j=0}^{2^{n_1}-1} \mathcal{A}_{\tau,\tau}(h_j^i)\cdot \mathbf{x}_j^{\mathrm{T}} \\ &= \mathbf{x}_{2^{n_1}+i}^{\mathrm{T}} + \sum_{j=0}^{2^{n_1}-1} \mathcal{A}_{\tau,0}\left((h_j'+x^w)^i(1+x^\tau)^i\right)\cdot(\mathbf{x}_j^*)^{\mathrm{T}} \\ &= \mathbf{x}_{2^{n_1}+i}^{\mathrm{T}} + \mathcal{A}_{\tau,0}\left((1+x^\tau)^i\right)\sum_{j=0}^{2^{n_1}-1}\mathcal{A}_{0,0}\left((h_j'+x^w)^i\right)\cdot(\mathbf{x}_j^*)^{\mathrm{T}}, \end{aligned} \tag{30}$$

where each $\mathbf{x}_j^* = (\mathbf{x}_i, 0, 0, ..., 0) \in \mathbb{F}_2^m$. In the above formula, the result of multiplying $\mathcal{A}_{0,0}\left((h_j'+x^w)^i\right)$ by $(\mathbf{x}_j^*)^{\mathrm{T}}$ is in fact the reverse coefficient vector of the resultant polynomial from multiplying $(h_j'+x^w)^i$ by $\mathbf{x}_j^*(x)$, where $\mathbf{x}_j^*(x)$ is shown in (25). Then, (30) can be easily obtained after calculating the

TABLE III
ASYMPTOTIC COMPLEXITIES OF ENCODING/DECODING WHEN r, τ ARE CONSTANTS AND THE TOTAL NUMBER OF DATA COLUMNS APPROACHES INFINITY (PER DATA BIT).

MDS array codes	Row size	Parity columns	Data columns	# of XORs	Note
BR code [4], [25]	$p-1$	$2 \le r < p$	p	r	p odd prime
IP code [5], [31]	$p-1$	$r \ge 2$	$p+r$	r	p odd prime
Generalized RDP code [6], [31]	$p-1$	$r \ge 2$	$p+r-1$	r	p odd prime
Rabin-like code [24]	$p-1$	$2 \le r < p$	p	$2r$	p odd prime
Circulant Cauchy code [23]	$p-1$	$2 \le r \le p$	$p+1$	$3r-2$	2 primitive element in $\mathbb{F}_p$
The Vandermonde-based V-ETBR code	$(p-1)\tau$	$2 \le r < 2^\lambda$	2^λ	$\lfloor \lg r \rfloor + 1$	p odd number
The Vandermonde-based V-ESIP code	$(p-1)\tau$	$r = 4$	$2^{\lfloor \frac{\lambda-1}{2} \rfloor} + 4$	3	p odd number

following polynomial multiplication

$$Q(i, \{\mathbf{x}_j^*(x)\}_{j=0}^{2^{n_1}-1}) := \sum_{j=0}^{2^{n_1}-1} (h'_j + x^w)^i \cdot \mathbf{x}_j^*(x), i \in [0, r). \tag{31}$$

The above formula can be simplified as (32), which is shown at the bottom of this page. This indicates that the syndrome computation can also be accelerated by (28). From the above, the syndrome computation can be completed through the following steps:

Step 1. From the input vector $(\mathbf{x}_0^*(x), ..., \mathbf{x}_{2^{n_1}-1}^*(x))$, calculate all required $\mathbf{y}_i$ in (32).
Step 2. Calculate $\{Q(i, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1})\}_{i=0}^3$ according to (32).
Step 3. Calculate (30).

In terms of complexity, Step 3 only requires a few vector additions and cyclic shifts. When r, τ are constants and n_0 approaches infinity, the asymptotic complexity of the above is dominated by the first two steps, and it is obviously the same as that in Sec. V-B1, i.e., $\lfloor \lg r \rfloor + 1 = 3$ XORs per data bit. TABLE II also lists the computational complexities for this syndrome computation with different parameters. Note that the total number of data columns at this time is $n+r-1$.

C. Comparison

TABLE III lists the asymptotic complexities of different binary MDS array codes. The fourth column shows the maximum number of data columns for each code, and the fifth column shows the asymptotic complexities of encoding and decoding, both of which are equal. It can be observed that the constructed Vandermonde-based variant codes not only have a more flexible row size and design parameter p but also have an exponentially growing total number of data columns with respect to p and minimal asymptotic encoding/decoding complexity.

To better demonstrate the impact of asymptotic computational complexity in practice, Fig. 3 and 4 also show the average number of XORs required for different binary MDS array codes with the total number of data columns being 127 and 251, respectively. Note that the average number of XORs is obtained by dividing the total number of XORs by the total number of bits in the data array, and that "Proposed 1" and "Proposed 2" in Fig. 3 and 4 correspond to the Vandermonde-based V-ETBR and V-ESIP codes in TABLE III, respectively. In our setup, the parameters p and τ of the variant codes are fixed to $p = 11$ and $\tau = 1$, while the parameter p of the other codes are the same as the total number of data columns ($p = 127$ in Fig. 3, $p = 251$ in Fig. 4). Each code has a row size of $p-1$ in the data array. This means that the row size in the data array of the variant codes is much smaller than that of other codes. In other words, the proposed variant codes require significantly less capacity per node in storage systems.

Let the variant codes use "Proposed 2" in the case of four parity columns and "Proposed 1" in the other cases. Fig. 3 shows that the average improvements in encoding/decoding for the variant codes compared to the Circulant Cauchy code [23], Rabin-like code [24], and BR code [4], [25] are 60%/61%, 51%/49%, and 12%/5%, respectively. The average improvements in Fig. 4 are 69%/69%, 63%/61%, and 26%/22%, respectively. With a fixed number of parity columns, the performance advantage of the variant codes in Fig. 4 is more obvious than that in Fig. 3.

It is worth noting that the practical performance of the variant codes constructed in this paper converges to the theoretical results when the number of data columns is much larger than that of parity columns. When the total number of data columns is not large enough, the proposed syndrome computation does not dominate the overall computational complexity, causing the efficiency of binary matrix-vector multiplication to be crucial. In our simulations, no scheduling algorithms for binary matrix-vector multiplication was used in the variant codes. Thus, there is a great potential to further improve the performance of the variant codes, which is also one of our future work.

VI. CONCLUSION

In this paper, we explore variant codes from codes over the polynomial ring $\mathbb{F}_2[x]/\langle \sum_{i=0}^{p-1} x^{i\tau} \rangle$, and then propose two new

$$Q(i, \{\mathbf{x}_j^*(x)\}_{j=0}^{2^{n_1}-1}) = \begin{cases} P(0, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}), & i = 0, \\ P(i, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}) + x^{iw} \cdot P(0, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}), & i = 1, 2, \\ P(3, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}) + x^w \cdot P(2, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}) + x^{2w} \cdot P(1, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}) \\ \quad + x^{3w} \cdot P(0, \{\mathbf{x}_j^*\}_{j=0}^{2^{n_1}-1}), & i = 3. \end{cases} \tag{32}$$

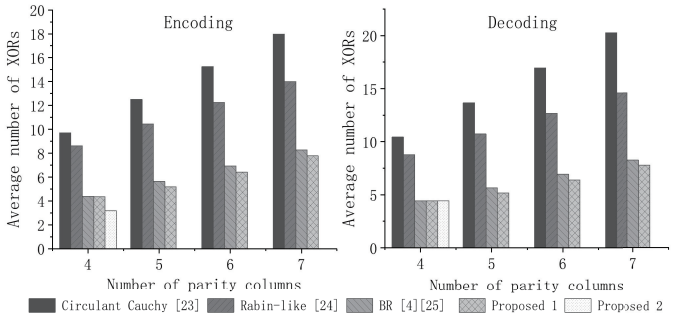

Fig. 3. Computational complexities of different binary MDS array codes (when the total number of data columns is 127).

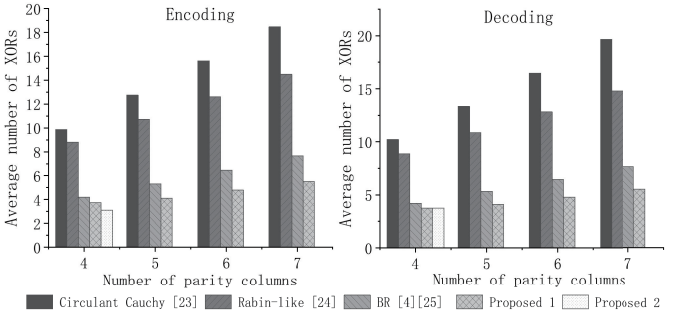

Fig. 4. Computational complexities of different binary MDS array codes (when the total number of data columns is 251).

classes of binary array codes, termed V-ETBR and V-ESIP codes. These variant codes are derived by mapping parity-check matrices over the polynomial ring to binary parity-check matrices. We show that the well-known generalized RDP code is a special case of the variant codes. To make this mapping a powerful tool in the construction of binary array codes, we explore in detail the connections between the variant codes and their counterparts over the polynomial ring, and provide conditions that make them binary MDS array codes. Based on these conditions, some new binary MDS array codes are explicitly constructed based on Cauchy and Vandermonde matrices. In addition, two fast syndrome computations for the constructed Vandermonde-based codes are proposed, both of which meet the lowest known asymptotic complexity among MDS codes [1]. Since the constructed codes have significantly more data columns than previous binary MDS array codes, the known lowest asymptotic computational complexity, and they are constructed from simpler binary parity-check matrices, they are attractive in practice.

ACKNOWLEDGMENTS

The authors would like to thank the associate editor, and the anonymous reviewers for their valuable comments and suggestions that helped us in improving this paper. In addition, the authors would like to acknowledge the support of the National Key Research and Development Program of China under Grant 2022YFA1004902.

REFERENCES

[1] L. Yu, S.-J. Lin, H. Hou, and Z. Li, "Reed-Solomon coding algorithms based on Reed-Muller transform for any number of parities," *IEEE Transactions on Computers*, vol. 72, no. 9, pp. 2677–2688, 2023.

[2] H. Hou, Y. S. Han, P. P. Lee, Y. Wu, G. Han, and M. Blaum, "A generalization of array codes with local properties and efficient encoding/decoding," *IEEE Transactions on Information Theory*, vol. 69, no. 1, pp. 107–125, 2022.

[3] J. D. Cook, R. Primmer, and A. de Kwant, "Compare cost and performance of replication and erasure coding," *hitachi Review*, vol. 63, p. 304, 2014.

[4] M. Blaum and R. M. Roth, "New array codes for multiple phased burst correction," *IEEE Transactions on Information Theory*, vol. 39, no. 1, pp. 66–77, 1993.

[5] M. Blaum, J. Bruck, and A. Vardy, "MDS array codes with independent parity symbols," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 529–542, 1996.

[6] M. Blaum, "A family of MDS array codes with minimal number of encoding operations," in *2006 IEEE International Symposium on Information Theory*, 2006, pp. 2784–2788.

[7] 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.

[8] J. Lv, W. Fang, B. Chen, S.-T. Xia, and X. Chen, "New constructions of binary MDS array codes and locally repairable array codes," in *2022 IEEE International Symposium on Information Theory (ISIT)*, 2022, pp. 2184–2189.

[9] D. A. Patterson, P. Chen, G. Gibson, and R. H. Katz, "Introduction to redundant arrays of inexpensive disks (RAID)," in *COMPCON Spring 89*. IEEE Computer Society, 1989, pp. 112–113.

[10] M. Blaum and S. R. Hetzler, "Array codes with local properties," *IEEE Transactions on Information Theory*, vol. 66, no. 6, pp. 3675–3690, 2019.

[11] M. Blaum, J. L. Hafner, and S. Hetzler, "Partial-MDS codes and their application to RAID type of architectures," *IEEE Transactions on Information Theory*, vol. 59, no. 7, pp. 4510–4519, 2013.

[12] K. W. Shum, H. Hou, M. Chen, H. Xu, and H. Li, "Regenerating codes over a binary cyclic code," in *2014 IEEE International Symposium on Information Theory*, 2014, pp. 1046–1050.

[13] M. Ye and A. Barg, "Explicit constructions of MDS array codes and RS codes with optimal repair bandwidth," in *2016 IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 1202–1206.

[14] H. Hou, Y. S. Han, B. Bai, and G. Zhang, "Towards efficient repair and coding of binary MDS array codes with small sub-packetization," in *2022 IEEE International Symposium on Information Theory (ISIT)*, 2022, pp. 3132–3137.

[15] Z. Shen and J. Shu, "Hv code: An all-around MDS code to improve efficiency and reliability of Raid-6 systems," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2014, pp. 550–561.

[16] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures," *IEEE Transactions on computers*, vol. 44, no. 2, pp. 192–202, 1995.

[17] 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.

[18] C. Huang and L. Xu, "STAR: An efficient coding scheme for correcting triple storage node failures," *IEEE Transactions on Computers*, vol. 57, no. 7, pp. 889–901, 2008.

[19] H. Hou, P. P. 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.

[20] H. Hou, K. W. Shum, M. Chen, and H. Li, "New MDS array code correcting multiple disk failures," in *2014 IEEE Global Communications Conference*, 2014, pp. 2369–2374.

[21] Z. Huang, H. Jiang, and K. Zhou, "An improved decoding algorithm for generalized RDP codes," *IEEE Communications Letters*, vol. 20, no. 4, pp. 632–635, 2016.

[22] M. Albrecht and G. Bard, "The M4RI library–version 20121224," *The M4RI Team*, vol. 105, p. 109, 2012.

[23] C. Schindelhauer and C. Ortolf, "Maximum distance separable codes based on circulant Cauchy matrices," in *International Colloquium on Structural Information and Communication Complexity*. Springer, 2013, pp. 334–345.

[24] H. Hou and Y. S. Han, "A new construction and an efficient decoding method for Rabin-like codes," *IEEE Transactions on Communications*, vol. 66, no. 2, pp. 521–533, 2017.
[25] P. Subedi and X. He, "A comprehensive analysis of XOR-based erasure codes tolerating 3 or more concurrent failures," in *2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum*, 2013, pp. 1528–1537.
[26] J. S. Plank, C. D. Schuman, and B. D. Robison, "Heuristics for optimizing matrix-based erasure codes for fault-tolerant storage systems," in *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012)*, 2012, pp. 1–12.
[27] C. Huang, J. Li, and M. Chen, "On optimizing XOR-based codes for fault-tolerant storage applications," in *2007 IEEE Information Theory Workshop*. IEEE, 2007, pp. 218–223.
[28] J. Lv, W. Fang, B. Chen, S.-T. Xia, and X. Chen, "Binary MDS array codes with flexible array dimensions and their fast encoding," in *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023, pp. 1249–1254.
[29] J. Lv, W. Fang, X. Chen, J. Yang, and S.-T. Xia, "New constructions of q-ary MDS array codes with multiple parities and their effective decoding," *IEEE Transactions on Information Theory*, vol. 69, no. 11, pp. 7082–7098, 2023.
[30] R. C. Subroto, "An algebraic approach to symmetric linear layers in cryptographic primitives," *Cryptography and Communications*, pp. 1–15, 2023.
[31] H. Hou, Y. S. Han, K. W. Shum, and H. Li, "A unified form of EVEN-ODD and RDP codes and their efficient decoding," *IEEE Transactions on Communications*, vol. 66, no. 11, pp. 5053–5066, 2018.
[32] J. Blomer, "An XOR-based erasure-resilient coding scheme," *Technical report at ICSI*, 1995.
[33] J. S. Plank and L. Xu, "Optimizing Cauchy Reed-Solomon codes for fault-tolerant network storage applications," in *Fifth IEEE International Symposium on Network Computing and Applications (NCA'06)*, 2006, pp. 173–180.
[34] J. S. Plank, S. Simmerman, and C. D. Schuman, "Jerasure: A library in c/c++ facilitating erasure coding for storage applications," *Technical Report CS-07–603, University of Tennessee*, 2007.
[35] Y. J. Tang and X. Zhang, "Fast en/decoding of Reed-Solomon codes for failure recovery," *IEEE Transactions on Computers*, vol. 71, no. 3, pp. 724–735, 2021.
[36] N. Kolokotronis, K. Limniotis, and N. Kalouptsidis, "Lower bounds on sequence complexity via generalised Vandermonde determinants," in *SETA*. Springer, 2006, pp. 271–284.
[37] U. Martínez-Peñas, "A general family of MSRD codes and PMDS codes with smaller field sizes from extended Moore matrices," *SIAM Journal on Discrete Mathematics*, vol. 36, no. 3, pp. 1868–1886, 2022.
[38] R. M. Roth, "Introduction to coding theory," *IET Communications*, vol. 47, no. 18-19, p. 4, 2006.
[39] L. Yu, Z. Lin, S.-J. Lin, Y. S. Han, and N. Yu, "Fast encoding algorithms for Reed–Solomon codes with between four and seven parity symbols," *IEEE Transactions on Computers*, vol. 69, no. 5, pp. 699–705, 2020.
[40] T. M. Chan, "Speeding up the four Russians algorithm by about one more logarithmic factor," in *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 212–217.

Leilei Yu received the B.Eng. degree in electronic information engineering from the Tianjin University of Technology, Tianjin, China, in 2015, and the Ph.D. degree in cyberspace security from the University of Science and Technology of China, Hefei, China, in 2021. From 2021 to 2022, he was a cybersecurity researcher with the Purple Mountain Laboratories, Nanjing, China. He is currently a post-doctoral research fellow with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China. His research focuses on coding theory and high-performance computing.

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 and as a consultant of Huawei Technology company. 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.

Jiasheng Yuan received the B.Sc. degree in electronic information science and technology and the M.Sc. degree in communication and information systems from Sun Yat-sen University, Guangzhou, China, in 2019 and 2021, respectively. He is currently pursuing the Ph.D. degree in Shenzhen Institute for Advanced Study, University of Electroinc Science and Technology of China. His research interests include coding theory and data communications.

Zhongpei Zhang received the B.S. and M.S. degrees from the Department of Physics, Sichuan Normal University, in 1990 and 1993, respectively, and the Ph.D. degree from the School of Computer and Communication Engineering, Southwest Jiaotong University, in 2000. From 2001 to 2003, he was a Post-Doctoral Fellow at the National Key Laboratory of Microwave and Digital Communication, Tsinghua University. From 2004 to 2005, he was a Post-Doctoral Fellow at the University of Oulu. He is currently a Professor and a Doctoral Tutor with the University of Electronic Science and Technology of China and the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China. He has participated in many research projects and chaired the National High-Tech Research and Development Program of China (863 Program) on Coordinated Multiple Points Transmission, and the National Natural Science Foundation of China on Massive MIMO Channel Acquisition. He has authored or coauthored more than 90 journal articles and conference papers. His research interests include channel coding, coordinated multiple points transmission, information theory, channel estimation based on compressive sensing, reconfigurable-reflecting-surface aided communications, and joint sensing and communication.