Two Classes of Binary MDS Array Codes with Asymptotically Optimal Repair for Any Single Column

Hanxu Hou, *Member, IEEE*, Yunghsiang S. Han, *Fellow, IEEE* and Patrick P. C. Lee, *Senior Member, IEEE*

***Abstract*—An $m\times(k+r)$ binary maximum distance separable (MDS) array code contains k information columns and r parity columns with each entry being a bit, where any k out of $k+r$ columns can recover the k information columns. When there is a failed column, it is critical to minimize the repair bandwidth that is the total number of bits downloaded from d out of $k+r-1$ surviving columns in repairing the failed column. In this paper, we first propose two explicit constructions of binary MDS array codes that have asymptotically optimal repair bandwidth for any information column, where $r\geq 2$ and $d=k+r-1$ for the first construction, and $r\geq 4$ is an even number and $d=k+\frac{r}{2}-1$ for the second construction. By applying a generic transformation for the proposed two classes of binary MDS array codes, we then obtain two classes of new binary MDS array codes that also have optimal repair bandwidth for any parity column and asymptotically optimal repair bandwidth for any information column.**

***Index Terms*—MDS codes, binary MDS array codes, asymptotically optimal repair bandwidth.**

I. Introduction

Distributed storage systems achieve high fault tolerance by deploying erasure codes to maintain data availability against failures of storage nodes. Binary maximum distance separable (MDS) array codes are a special class of erasure codes, where only XOR operations are involved in encoding and decoding procedures. An (n,k) binary MDS array code encodes k *information columns* of m bits into additional $r=n-k$ *parity columns* of the same size, such that any k out of n columns can retrieve all km information bits of k information columns. The m bits in each column are stored in the same storage node. The number of bits stored in each column, m, is called *sub-packetization*. We refer to a disk as a column or a storage node interchangeably, and an entry in the array as a bit. Typical constructions of binary MDS array codes are X-code [2], RDP codes [3] and EVENODD codes [4] with two parity columns ($r=2$), and STAR codes [5], generalized RDP codes [6] and generalized EVENODD codes [7] with more than two parity columns ($r>2$).

This paper was presented in part in [1] at the 28th International Conference on Computer Communications and Networks (ICCCN 2019), Valencia, Spain, July, 2019. H. Hou and Y. S. Han are with the School of Electrical Engineering & Intelligentization, Dongguan University of Technology (E-mail: hhx@dgut.edu.cn, hanyx@dgut.edu.cn). P. P. C. Lee is with Department of Computer Science and Engineering, The Chinese University of Hong Kong (E-mail: pclee@cse.cuhk.edu.hk). This work was partially supported by the National Natural Science Foundation of China (No. 61701115, 61671007), Start Fund of Dongguan University of Technology (No. GB200902-19, KCYXM2017025), Research Grants Council of HKSAR (GRF 14216316 and AoE/P-404/18), Innovation and Technology Fund (ITS/315/18FX), the Quality Engineering Project of 2018 DGUT (No. 201802016), 2019 Guangdong Province Teaching Quality and Teaching Reform (No. 2063016020).

As node failures are prevalent in a practical distributed storage system [8], we should repair the failed node in order to maintain the same level of data reliability. It is important to minimize the *repair bandwidth*, defined as the amount of bits downloaded during a repair operation, in distributed storage in which network transfer is the bottleneck. It is shown in [9] that we can repair a failed node by downloading $\frac{m}{d-k+1}$ bits or more from each of the d surviving nodes, where $k\leq d\leq n-1$. The minimum repair bandwidth of a failed node is thus

$$\frac{dm}{d-k+1}. \tag{1}$$

A failed node of binary MDS array codes is said to has *optimal repair* if the repair bandwidth of the failed node is equal to the lower bound in (1). Many constructions of MDS codes over large finite fields [9]–[12] have been proposed to achieve the minimum repair bandwidth. Although we can represent a field element by a binary vector and all field computations can be converted into XOR operations, recent results [13] show that directly vectorizing the XOR operations can achieve better performance than vectorizing finite field operations. In this paper, we propose two classes of binary MDS array codes that can achieve asymptotically or exactly optimal repair bandwidth for any column. Here "asymptotic" means that the repair bandwidth achieves the minimum value when d approaches infinity and $d-k$ is fixed.[1]

A. Related Work

We can conventionally repair a failed column by downloading all m bits from any k surviving columns. However, the repair bandwidth is k times of the failed m bits. There have been several constructions of binary MDS array codes [15]–[19] to achieve asymptotically or exactly optimal repair bandwidth for any information column. MDR codes [15], [20] and ButterFly codes [16], [21] are binary MDS array codes with only two parity columns that can achieve optimal repair bandwidth for any information column. Binary MDS array codes with three or more parity columns in [17], [19] have asymptotically optimal repair bandwidth for any information column; however, they do not provide efficient repair method for parity column.

[1]This is equivalent to the definition given in [14] where n approaches infinity and $n-k$ is fixed.

Note that the second construction of binary MDS array codes in [18] has asymptotically optimal repair bandwidth for any column. The construction is based on parity-check matrix and the encoding complexity is high. BASIC regenerating codes [22]–[24] also achieve the optimal repair bandwidth and only XOR operations are involved in coding and repairing procedures. However, they build on the product-matrix coding framework [25] with a low *code rate* (i.e., the ratio of the file size to the total storage space is less than 0.5). Two transformations for binary MDS array codes to enable optimal repair for any of the chosen $d-k+1$ columns are proposed in [26], [27]. One general transformation to enable optimal repair for any of the chosen $(d-k+1)\lfloor\frac{r-1}{d-k}\rfloor$ columns is given in [28]. In general, to enable optimal repair for any column of a binary MDS array code, one needs to recursively employ the transformations [26]–[28] for the binary MDS array code with many times, and both the encoding and decoding complexities of the resulting transformed codes are increased.

B. Contributions

The contributions of this work are summarized as follows. First, we propose two classes of binary MDS array codes that achieve asymptotically optimal repair bandwidth for any information column, where "asymptotic" means that the repair bandwidth achieves the minimum value asymptotically in d. In the first construction, we have $r \geq 2$ and $d = k + r - 1$. In the second construction, the number of parity column is an even number with $r \geq 4$ and $d = k + \frac{r}{2} - 1$. Note that the two encoding matrices of our two binary MDS array codes are new. Second, by applying the transformation in [26], [28] once for the first construction of the proposed binary MDS array codes, we can obtain a class of new codes that have asymptotically optimal repair bandwidth for any information column and optimal repair bandwidth for any parity column, where $d = k + r - 1$. We can also obtain another class of new codes by applying the transformation in [28] once for the second construction that have asymptotically optimal repair bandwidth for any information column and optimal repair bandwidth for any parity column, where $d = k + \frac{r}{2} - 1$. Note that we cannot directly apply the transformation in [26], [28] for the proposed codes and we need to carefully choose the encoding coefficient x^{e_j} (see Section IV for details) in order to ensure that the repair bandwidth of each information column of the transformed codes is the same as that of the underlying binary MDS array codes. The transformation in [26] is designed for EVENODD codes and we can choose any non-zero value for e_j. In the proposed first transformed codes, we should choose a suitable value for e_j to ensure that the repair bandwidth of each information column of the first transformed codes is the same as that of the first array codes in Section II-B. When $x^{e_j} = x^{\tau}$, we show that the repair bandwidth of any information column is not increased (see Theorem 6). In fact, we can show that only when e_j is a multiple of τ but not $p\tau$, the repair bandwidth of any information column is not increased. In the second transformed codes, we also design the encoding coefficient as $x^{e_j} = x^{\tau}$ and show that the second transformed codes have the same repair bandwidth for any information column as that of the second array codes in Section II-C.

The key differences between the proposed codes and the existing binary MDS array codes with asymptotically or exactly optimal repair bandwidth are as follows. First, in contrast to existing constructions with two parity columns in [15], [16], the quotient ring $\mathbb{F}_2[x]/(1+x^{p\tau})$ with cyclic structure is employed in our construction. Second, although both our construction and the existing binary MDS array codes in [17]–[19] are based on the quotient ring $\mathbb{F}_2[x]/(1+x^{p\tau})$, the detailed constructions are different. The work in [17] only considers the construction of binary MDS array codes with three parity columns. The work in [18] presents a general coding framework of constructing binary array codes over $\mathbb{F}_2[x]/(1+x^{p\tau})$ and proposes two explicit constructions based on the coding framework. The first construction of [18] has asymptotically optimal repair only for any information column. Although the second construction of [18] has asymptotically optimal repair for any column, it needs to solve linear equations over $\mathbb{F}_2[x]/(1+x^{p\tau})$ in the encoding process and thus incurs high encoding complexity. The work in [19] explores the property of the encoding matrix that can enable asymptotically optimal repair for any information column with $d = k + 1$. In this paper, we not only propose two new constructions of binary MDS array codes based on the coding framework in [18], but also apply the transformations in [26], [28] for the proposed codes. The transformed array codes have both the advantage enabled by the cyclic structure in the ring $\mathbb{F}_2[x]/(1+x^{p\tau})$ and the advantage enabled by the transformations in [26], [28]. Because of the above two differences, our transformed codes have lower encoding complexity compared with the existing binary MDS array codes [18], [26] with asymptotically or exactly optimal repair bandwidth for all columns (see Section V for details).

Note that if we directly apply the transformation in [26], [28] once for the codes in [17], [19], then we can obtain the transformed codes that have optimal repair bandwidth for any parity column. However, the repair bandwidth of each information column of the transformed codes is larger than that of codes in [17], [19], since some parity bits downloaded in repairing the information column are mixed with other parity bits which are not needed in the repair procedure.

Even though the proposed binary MDS array codes and the MDS codes with optimal repair [10]–[12], [29]–[32] are all based on constructing generator matrices or parity matrices, the proposed codes are constructed over binary field and the encoding matrices are designed on the ring with a cyclic structure.

The rest of the paper is organized as follows. Section II reviews the coding framework of binary MDS array codes [18] from the point of view of generator matrix, and presents two constructions of binary MDS array codes that have asymptotically optimal repair for any information column. Section III presents the repair algorithm for the proposed codes in Section II for any information column. Section IV shows two classes of new MDS array codes that have asymptotically optimal repair for any single column by applying a transformation for two classes of codes given in Section II. Section VI concludes the paper.

II. Two Constructions of Binary MDS Array Codes with Asymptotically Optimal Repair for Any Information Column

In this section, we first review the coding framework of binary MDS array codes [18] from the point of view of generator matrix, and then present two new constructions of binary MDS array codes with general parameters k and r that have asymptotically optimal repair for each information column for both constructions.

A. Coding Framework of Binary MDS Array Codes

The coding framework in [18] presents a method to construct a binary array code of size $(p-1)\tau \times (k+r)$, where $k \geq 2$ is the number of information columns, $r \geq 2$ is the number of parity columns, p is a prime number and τ is an integer that will be specified in the explicit construction.

The array codes contain $k+r$ columns, where the first k columns are information columns and the last r columns are parity columns. We label the index of $k+r$ columns from 0 to $k+r-1$. Let $s_{i,j}$ be the entry in row i and column j of the $(p-1)\tau \times (k+r)$ array, where $0 \leq i \leq (p-1)\tau - 1$, $0 \leq j \leq k+r-1$ and $s_{i,j} \in \mathbb{F}_2$. We want to represent each column by a polynomial in $\mathbb{F}_2[x]/(1+x^{p\tau})$ in order to employ the cyclic structure of $\mathbb{F}_2[x]/(1+x^{p\tau})$. We thus need to append τ extra bits to each column and each column now has $(p-1)\tau$ bits. Hence, we can represent each column by a polynomial in $\mathbb{F}_2[x]/(1+x^{p\tau})$. The cyclic structure is crucial for reducing the repair bandwidth. We can choose some values for τ to achieve asymptotically optimal repair bandwidth and some values for p to make the codes to be MDS. Specifically, the extra bits $s_{(p-1)\tau+\mu,j}$ for $\mu = 0,1,\ldots,\tau-1$ associated with $(p-1)\tau$ bits stored in column j are computed as

$$s_{(p-1)\tau+\mu,j} = \sum_{i=0}^{p-2} s_{i\tau+\mu,j}, \tag{2}$$

where $j = 0,1,\ldots,k-1$. By defining the extra bits as in (2), we can operate the polynomial representing a column over a sub-ring of $\mathbb{F}_2[x]/(1+x^{p\tau})$ that is useful to have the MDS property. Up to now, we can only append the extra bits for k information columns, as we do not know the value of parity bits before the encoding procedure. In the encoding procedure, we will show that the appended τ extra bits $s_{(p-1)\tau+\mu,j}$ for $\mu = 0,1,\ldots,\tau-1$ also satisfy (2) for $j = k, k+1, \ldots, k+r-1$.

We can represent $(p-1)\tau$ bits $s_{0,j}, s_{1,j}, \ldots, s_{(p-1)\tau-1,j}$ in column j and τ associated extra bits $s_{(p-1)\tau,j}$, $s_{(p-1)\tau+1,j}, \ldots, s_{p\tau-1,j}$ by the polynomial

$$s_j(x) = s_{0,j} + s_{1,j}x + \ldots + s_{p\tau-1,j}x^{p\tau-1}, \tag{3}$$

where $j = 0,1,\ldots,k+r-1$. The above polynomial $s_j(x)$ can be viewed as a polynomial over $\mathbb{F}_2[x]/(1+x^{p\tau})$. The polynomials $s_j(x)$ for $j = 0,1,\ldots,k-1$ are *information polynomials* that correspond the information columns, and polynomials $s_j(x)$ for $j = k, k+1, \ldots, k+r-1$ are *parity polynomials* that correspond the parity columns. We can compute r parity polynomials by the multiplication of k information polynomials and a $k \times r$ *encoding matrix* $\mathbf{P}_{k\times r}$ as

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

over $\mathbb{F}_2[x]/(1+x^{p\tau})$.

There is a cyclic structure in the quotient ring $R_{p\tau} = \mathbb{F}_2[x]/(1+x^{p\tau})$, as a multiplication by x in $R_{p\tau}$ can be implemented as a *cyclic-right-shift*, i.e.,

$$\begin{aligned}&x \cdot (s_{0,j} + s_{1,j}x + \ldots + s_{p\tau-1,j}x^{p\tau-1}) \bmod (1+x^{p\tau})\\ =&(s_{p\tau-1,j} + s_{0,j}x + \ldots + s_{p\tau-2,j}x^{p\tau-1}).\end{aligned}$$

The cyclic structure is crucial for reducing the repair bandwidth for one single information column failure. In choosing the encoding matrix to construct the array codes, we employ the cyclic structure to choose the parameter τ to obtain asymptotically optimal repair bandwidth for any information column.

Let $C_{p\tau}$ be the sub-ring of $R_{p\tau}$ such that each polynomial in $C_{p\tau}$ is a multiple of $1+x^{\tau}$, i.e.,

$$C_{p\tau} = \{a(x)(1+x^{\tau}) \bmod (1+x^{p\tau}) | a(x) \in R_{p\tau}\}.$$

The next lemma shows the necessary and sufficient condition to check whether a polynomial is in the ring $C_{p\tau}$ or not.

Lemma 1. *[18, Theorem 1] A polynomial $s_j(x) \in R_{p\tau}$ is in $C_{p\tau}$ if and only if the coefficients of $s_j(x)$ satisfy* (2).

By Lemma 1, each information polynomial is in $C_{p\tau}$. Given the encoding matrix $\mathbf{P}_{k\times r}$, we can compute r parity polynomials by multiplying the k information polynomials and $\mathbf{P}_{k\times r}$. The resultant r parity polynomials are also in $C_{p\tau}$ as

$$\forall c(x) \in R_{p\tau}, \forall s(x) \in C_{p\tau}, c(x)s(x) \in C_{p\tau}.$$

We summarize the encoding procedure as follows. Given $k(p-1)\tau$ information bits $s_{i,j}$ for $i = 0,1,\ldots,(p-1)\tau-1$ and $j = 0,1,\ldots,k-1$, we append τ extra bits for each $(p-1)\tau$ information bits by (2), and represent $k(p-1)\tau$ information bits and $k\tau$ extra bits by k information polynomials in $C_{p\tau}$. Then, we compute r parity polynomials by multiplying the k information polynomials and the encoding matrix $\mathbf{P}_{k\times r}$. Finally, the coefficients in the polynomial $s_j(x)$ with degrees from 0 to $(p-1)\tau-1$ are stored in column j for $j = 0,1,\ldots,k+r-1$. From the above encoding procedure, the constructed array codes are determined by the encoding matrix.

Consider an example with $k=2$, $r=2$, $\tau=4$ and $p=3$. The 16 information bits are $s_{i,j}$ for $i = 0,1,\ldots,7$ and $j = 0,1$. The encoding matrix of the example is

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

Table I shows the example. Note that we do not store the last four bits (extra bits) in column. We claim that the repair bandwidth of column 0 is optimal. We can repair the bits $s_{i,0}$ with $i = 0,2,4,6$ by $s_{i,0} = s_{i,2} + s_{i,1}$, and repair the other

TABLE I
AN EXAMPLE WITH $k=2$, $r=2$, $\tau=4$ AND $p=3$, WHERE $s_{8+i,j}=s_{i,j}+s_{4+i,j}$ FOR $i=0,1,2,3$ AND $j=0,1,2,3$.

$s_{0,0}$	$s_{0,1}$	$s_{0,2}=s_{0,0}+s_{0,1}$	$s_{0,3}=s_{11,0}+s_{10,1}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}=s_{1,0}+s_{1,1}$	$s_{1,3}=s_{0,0}+s_{11,1}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}=s_{2,0}+s_{2,1}$	$s_{2,3}=s_{1,0}+s_{0,1}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}=s_{3,0}+s_{3,1}$	$s_{3,3}=s_{2,0}+s_{1,1}$
$s_{4,0}$	$s_{4,1}$	$s_{4,2}=s_{4,0}+s_{4,1}$	$s_{4,3}=s_{3,0}+s_{2,1}$
$s_{5,0}$	$s_{5,1}$	$s_{5,2}=s_{5,0}+s_{5,1}$	$s_{5,3}=s_{4,0}+s_{3,1}$
$s_{6,0}$	$s_{6,1}$	$s_{6,2}=s_{6,0}+s_{6,1}$	$s_{6,3}=s_{5,0}+s_{4,1}$
$s_{7,0}$	$s_{7,1}$	$s_{7,2}=s_{7,0}+s_{7,1}$	$s_{7,3}=s_{6,0}+s_{5,1}$
$s_{8,0}$	$s_{8,1}$	$s_{8,2}=s_{8,0}+s_{8,1}$	$s_{8,3}=s_{7,0}+s_{6,1}$
$s_{9,0}$	$s_{9,1}$	$s_{9,2}=s_{9,0}+s_{9,1}$	$s_{9,3}=s_{8,0}+s_{7,1}$
$s_{10,0}$	$s_{10,1}$	$s_{10,2}=s_{10,0}+s_{10,1}$	$s_{10,3}=s_{9,0}+s_{8,1}$
$s_{11,0}$	$s_{11,1}$	$s_{11,2}=s_{11,0}+s_{11,1}$	$s_{11,3}=s_{10,0}+s_{9,1}$

bits $s_{i,0}$ with $i=1,3,5,7$ by $s_{i,0}=s_{i+1,3}+s_{i-1,1}$. Note that $s_{8,3}=s_{0,3}+s_{4,3}$. Therefore, we need to download the following 12 bits

$$s_{0,1},s_{2,1},s_{4,1},s_{6,1},s_{0,2},s_{2,2},s_{4,2},s_{6,2},s_{0,3},s_{2,3},s_{4,3},s_{6,3},$$

to recover the eight bits in column 0, and, according to (1), the repair bandwidth is optimal. Column 1 can be recovered by downloading the following 14 bits

$$s_{0,0},s_{1,0},s_{3,0},s_{4,0},s_{5,0},s_{7,0},s_{0,2},$$
$$s_{1,2},s_{4,2},s_{5,2},s_{0,3},s_{1,3},s_{4,3},s_{5,3}.$$

B. *The First Construction of Binary MDS Array Codes*

The first constructed binary MDS array codes have asymptotically optimal repair for any information column with $d=k+r-1$. The encoding matrix of the constructed codes is

$$\mathbf{P}_{k\times r}=\begin{bmatrix}1 & x & x^2 & \cdots & x^{r-1}\\ 1 & x^r & x^{2r} & \cdots & x^{(r-1)r}\\ 1 & x^{r^2} & x^{2r^2} & \cdots & x^{(r-1)r^2}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 1 & x^{r^{k-1}} & x^{2r^{k-1}} & \cdots & x^{(r-1)r^{k-1}}\end{bmatrix}, \quad (4)$$

where $k\geq 2$, $r\geq 2$, $\tau=r^k$ and $p\geq r$. The example in Section II-A is our code with $k=2$, $r=2$, $\tau=4$ and $p=3$.

C. *The Second Construction of Binary MDS Array Codes*

Next, we present the second construction of binary MDS array codes with $r\geq 4$ being an even integer such that the repair bandwidth of any information column is asymptotically optimal with $d=k+\frac{r}{2}-1$.

Let $k\geq 3$, $r\geq 4$ be an even number, $d=k+\frac{r}{2}-1$, $\tau=(\frac{r}{2})^{\lceil\frac{k}{2}\rceil}$ and $p\geq\frac{r}{2}$. The encoding matrix of the constructed codes is given in (5). Consider an example with $k=3$, $r=4$, $\tau=4$ and $p=3$. The 24 information bits are $s_{i,j}$ for $i=0,1,\ldots,7$ and $j=0,1,2$. The encoding matrix of the example is

$$\mathbf{P}_{4\times 4}=\begin{bmatrix}1 & x & 1 & 1\\ 1 & x^2 & x^2 & x^4\\ 1 & 1 & x & x^8\end{bmatrix}.$$

Table II shows the example. Note that we do not store the last four bits (extra bits) in column. We claim that the repair bandwidth of column 0 is optimal. We can repair the four bits $s_{i,0}$ with $i=0,2,4,6$ by $s_{i,0}=s_{i,1}+s_{i,2}+s_{i,3}$, and repair the other bits $s_{i,0}$ with $i=1,3,5,7$ by $s_{i,0}=s_{i-1,1}+s_{i+1,2}+s_{i+1,4}$. Note that $s_{8+\ell,j}=s_{\ell,j}+s_{4+\ell,j}$, for $\ell=0,1,2,3$ and $j=0,1,\ldots,7$. Therefore, we need to download the following 16 bits

$$s_{0,1},s_{2,1},s_{4,1},s_{6,1},s_{0,2},s_{2,2},s_{4,2},s_{6,2},$$
$$s_{0,3},s_{2,3},s_{4,3},s_{6,3},s_{0,4},s_{2,4},s_{4,4},s_{6,4},$$

to recover the eight bits in column 0, and, according to (1), the repair bandwidth is optimal. Column 1 can be recovered by downloading the following 18 bits

$$s_{0,0},s_{1,0},s_{3,0},s_{4,0},s_{5,0},s_{7,0},s_{0,2},s_{1,2},s_{4,2},s_{5,2},$$
$$s_{0,3},s_{1,3},s_{4,3},s_{5,3},s_{0,4},s_{1,4},s_{4,4},s_{5,4}.$$

Column 2 can be recovered by downloading the following 16 bits

$$s_{0,0},s_{2,0},s_{4,0},s_{6,0},s_{0,1},s_{2,1},s_{4,1},s_{6,1},$$
$$s_{0,5},s_{2,5},s_{4,5},s_{6,5},s_{0,6},s_{2,6},s_{4,6},s_{6,6}.$$

D. *The MDS Property*

We need to show that any k out of $k+r$ columns can reconstruct all information bits. Recall that all information polynomials and parity polynomials are in $C_{p\tau}$, we thus construct the array codes over the ring $C_{p\tau}$. By Lemma 3 in [18], the ring $C_{p\tau}$ is isomorphic to $\mathbb{F}_2[x]/M_p^\tau(x)$, where

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

The MDS property of the constructed array codes is equivalent to that the determinant of any square sub-matrix of the encoding matrix $\mathbf{P}_{k\times r}$ is invertible over $\mathbb{F}_2[x]/M_p^\tau(x)$. According to Theorem 6 in [18], the ring $\mathbb{F}_2[x]/M_p^\tau(x)$ is isomorphic to the direct sum of t rings

$$\mathbb{F}_2[x]/(f_1(x))^{\ell_1},\mathbb{F}_2[x]/(f_2(x))^{\ell_2},\ldots,\mathbb{F}_2[x]/(f_t(x))^{\ell_t},$$

where t is a positive integer, $\ell_i\geq 0$ for $i=1,2,\ldots,t$, $\gcd(f_i(x),f_j(x))=1$ for all $1\leq i\neq j\leq t$ and $\deg(f_i(x))\leq\deg(f_j(x))$ for $i<j$. Therefore, we need to show that the determinants of all $\ell\times\ell$ sub-matrices are invertible in the ring $\mathbb{F}_2[x]/(f_i(x))$ for $i=1,2,\ldots,t$. The next theorem shows the MDS property.

Theorem 2. *If* $\deg(f_1(x))$ *is larger than*

$$(r-1)\cdot r^k-\frac{r^k-r^{k-r+1}}{r-1}, \quad (6)$$

then the array codes with the encoding matrix in (4) *satisfy the MDS property. If* $\deg(f_1(x))$ *is larger than*

$$2\cdot\frac{r}{2}^{k-1}+(k-1)\cdot\frac{r}{2}^{\frac{r}{2}}-\frac{\frac{r}{2}^{k-1}+\frac{r}{2}^{k-2}-\frac{r}{2}^{k-\frac{r}{2}}-\frac{r}{2}^{k-\frac{r}{2}-1}}{(\frac{r}{2}-1)^2}, \quad (7)$$

then the array codes with the encoding matrix in (5) *satisfy the MDS property.*

Proof. We show that the determinants of all square sub-matrices are invertible in $\mathbb{F}_2[x]/(f_i(x))$ for $i=1,2,\ldots,t$. It is

$$\mathbf{P}_{k\times r}=\left[\begin{array}{ccccc|cccc} 1 & x & x^2 & \cdots & x^{\frac{r}{2}-1} & 1 & \cdots & 1 & 1 & 1\\ 1 & x^{\frac{r}{2}} & x^{2\frac{r}{2}} & \cdots & x^{(\frac{r}{2}-1)\frac{r}{2}} & x^{(\frac{r}{2}-1){\frac{r}{2}}^{k-2}} & \cdots & x^{2{\frac{r}{2}}^{k-2}} & x^{{\frac{r}{2}}^{k-2}} & x^{\frac{r}{2}^{\frac{r}{2}}}\\ 1 & x^{{\frac{r}{2}}^2} & x^{2{\frac{r}{2}}^2} & \cdots & x^{(\frac{r}{2}-1){\frac{r}{2}}^2} & x^{(\frac{r}{2}-1){\frac{r}{2}}^{k-3}} & \cdots & x^{2{\frac{r}{2}}^{k-3}} & x^{{\frac{r}{2}}^{k-3}} & x^{2\frac{r}{2}^{\frac{r}{2}}}\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\ 1 & x^{{\frac{r}{2}}^{k-3}} & x^{2{\frac{r}{2}}^{k-3}} & \cdots & x^{(\frac{r}{2}-1){\frac{r}{2}}^{k-3}} & x^{(\frac{r}{2}-1){\frac{r}{2}}^2} & \cdots & x^{2{\frac{r}{2}}^2} & x^{{\frac{r}{2}}^2} & x^{(k-3)\frac{r}{2}^{\frac{r}{2}}}\\ 1 & x^{{\frac{r}{2}}^{k-2}} & x^{2{\frac{r}{2}}^{k-2}} & \cdots & x^{(\frac{r}{2}-1){\frac{r}{2}}^{k-2}} & x^{(\frac{r}{2}-1)\frac{r}{2}} & \cdots & x^{2\frac{r}{2}} & x^{\frac{r}{2}} & x^{(k-2)\frac{r}{2}^{\frac{r}{2}}}\\ 1 & 1 & 1 & \cdots & 1 & x^{\frac{r}{2}-1} & \cdots & x^2 & x & x^{(k-1)\frac{r}{2}^{\frac{r}{2}}} \end{array}\right]. \tag{5}$$

TABLE II
AN EXAMPLE OF THE SECOND CONSTRUCTED ARRAY CODES WITH $k=3$, $r=4$, $\tau=4$ AND $p=3$, WHERE $s_{8+i,j}=s_{i,j}+s_{4+i,j}$ FOR $i=0,1,2,3$ AND $j=0,1,\ldots,6$.

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}=s_{0,0}+s_{0,1}+s_{0,2}$	$s_{0,4}=s_{11,0}+s_{10,1}+s_{0,2}$	$s_{0,5}=s_{0,0}+s_{10,1}+s_{11,2}$	$s_{0,6}=s_{0,0}+s_{8,1}+s_{4,2}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}=s_{1,0}+s_{1,1}+s_{1,2}$	$s_{1,4}=s_{0,0}+s_{11,1}+s_{1,2}$	$s_{1,5}=s_{1,0}+s_{11,1}+s_{0,2}$	$s_{1,6}=s_{1,0}+s_{9,1}+s_{5,2}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}=s_{2,0}+s_{2,1}+s_{2,2}$	$s_{2,4}=s_{1,0}+s_{0,1}+s_{2,2}$	$s_{2,5}=s_{2,0}+s_{0,1}+s_{1,2}$	$s_{2,6}=s_{2,0}+s_{10,1}+s_{6,2}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}=s_{3,0}+s_{3,1}+s_{3,2}$	$s_{3,4}=s_{2,0}+s_{1,1}+s_{3,2}$	$s_{3,5}=s_{3,0}+s_{1,1}+s_{2,2}$	$s_{3,6}=s_{3,0}+s_{11,1}+s_{7,2}$
$s_{4,0}$	$s_{4,1}$	$s_{4,2}$	$s_{4,3}=s_{4,0}+s_{4,1}+s_{4,2}$	$s_{4,4}=s_{3,0}+s_{2,1}+s_{4,2}$	$s_{4,5}=s_{4,0}+s_{2,1}+s_{3,2}$	$s_{4,6}=s_{4,0}+s_{0,1}+s_{8,2}$
$s_{5,0}$	$s_{5,1}$	$s_{5,2}$	$s_{5,3}=s_{5,0}+s_{5,1}+s_{5,2}$	$s_{5,4}=s_{4,0}+s_{3,1}+s_{5,2}$	$s_{5,5}=s_{5,0}+s_{3,1}+s_{4,2}$	$s_{5,6}=s_{5,0}+s_{1,1}+s_{9,2}$
$s_{6,0}$	$s_{6,1}$	$s_{6,2}$	$s_{6,3}=s_{6,0}+s_{6,1}+s_{6,2}$	$s_{6,4}=s_{5,0}+s_{4,1}+s_{6,2}$	$s_{6,5}=s_{6,0}+s_{4,1}+s_{5,2}$	$s_{6,6}=s_{6,0}+s_{2,1}+s_{10,2}$
$s_{7,0}$	$s_{7,1}$	$s_{7,2}$	$s_{7,3}=s_{7,0}+s_{7,1}+s_{7,2}$	$s_{7,4}=s_{6,0}+s_{5,1}+s_{7,2}$	$s_{7,5}=s_{7,0}+s_{5,1}+s_{6,2}$	$s_{7,6}=s_{7,0}+s_{3,1}+s_{11,2}$
$s_{8,0}$	$s_{8,1}$	$s_{8,2}$	$s_{8,3}=s_{8,0}+s_{8,1}+s_{8,2}$	$s_{8,4}=s_{7,0}+s_{6,1}+s_{5,2}$	$s_{8,5}=s_{8,0}+s_{6,1}+s_{7,2}$	$s_{8,6}=s_{8,0}+s_{4,1}+s_{0,2}$
$s_{9,0}$	$s_{9,1}$	$s_{9,2}$	$s_{9,3}=s_{9,0}+s_{9,1}+s_{9,2}$	$s_{9,4}=s_{8,0}+s_{7,1}+s_{6,2}$	$s_{9,5}=s_{9,0}+s_{7,1}+s_{8,2}$	$s_{9,6}=s_{9,0}+s_{5,1}+s_{1,2}$
$s_{10,0}$	$s_{10,1}$	$s_{10,2}$	$s_{10,3}=s_{10,0}+s_{10,1}+s_{10,2}$	$s_{10,4}=s_{9,0}+s_{8,1}+s_{7,2}$	$s_{10,5}=s_{10,0}+s_{8,1}+s_{9,2}$	$s_{10,6}=s_{10,0}+s_{6,1}+s_{2,2}$
$s_{11,0}$	$s_{11,1}$	$s_{11,2}$	$s_{11,3}=s_{11,0}+s_{11,1}+s_{11,2}$	$s_{11,4}=s_{10,0}+s_{9,1}+s_{8,2}$	$s_{11,5}=s_{11,0}+s_{9,1}+s_{10,2}$	$s_{11,6}=s_{11,0}+s_{7,1}+s_{3,2}$

easy to check that the determinant of any square sub-matrix of the encoding matrices in (4) and (5) is a non-zero polynomial in $\mathbb{F}_2[x]$. If the maximum degree of the determinant is less than $\deg(f_i(x))$, then the determinant is invertible in $\mathbb{F}_2[x]/(f_i(x))$. Recall that $\deg(f_1(x)) \le \deg(f_2(x)) \le \ldots \le \deg(f_t(x))$. It is sufficient to show that the maximum degree of the determinants of all the square sub-matrices is less than $\deg(f_1(x))$. The maximum degree among all the determinants is achieved when choosing the $r\times r$ sub-matrix of the matrix in (4) as the last r rows. After calculating the sum of the degrees, the maximum degree becomes (6). With the same argument for the matrix in (5), we can calculate that the maximum degree of the determinants of all the sub-matrices is (7). Therefore, the two codes satisfy the MDS property if $\deg(f_1(x))$ is larger than (6) and (7), respectively. $\square$

By Theorem 2, we choose parameters p and τ such that $\deg(f_1(x))$ is larger than (6) and (7), respectively, in order to ensure that the two codes are MDS codes. Although the values in (6) and (7) are exponentially increasing with k and r, we can choose a special prime for the parameter p which is small when r is given. When p is a prime number with 2 be a primitive element in $\mathbb{F}_p$ and τ is a power of p, then $M_p^\tau(x)$ is irreducible [18]. We then have $f_1(x)=1+x^\tau+\ldots+x^{(p-1)\tau}$ and $\ell_1=\tau$. Therefore, the first constructed codes and the second constructed codes are MDS codes if $(p-1)\tau$ is larger than the values in (6) and (7), respectively. For example, when $r=3$ and $p=3$, the first constructed codes are MDS codes for $k\ge 2$.

III. Repair Algorithm for Any Information Column

In this section, we present repair algorithm for any information column of the two constructions such that the repair bandwidths of two codes are asymptotically optimal.

A. Repair Algorithm for the First Construction

Suppose that the information column f fails, where $0 \le f \le k-1$, we want to present a repair algorithm to repair column f by downloading bits from the other healthy $k+r-1$ columns.

As we can compute the extra bits for each column by (2) when necessary, we assume that all the extras are known in the following. According to column j of the encoding matrix in (4), the parity bits $s_{i,k+j}$ for $i=0,1,\ldots,p\tau-1$ are computed as

$$s_{i,k+j}=s_{i-j,0}+s_{i-jr,1}+s_{i-jr^2,2}+\ldots+s_{i-jr^{k-1},k-1}, \tag{8}$$

where $j=0,1,\ldots,r-1$. Note that all the indices are taken modulo $p\tau$. By (8), we can repair the bit $s_{i,f}$ stored in the failed column by

$$s_{i,f}=s_{i+jr^f,k+j}+\sum_{\ell=0,\ell\neq f}^{k-1} s_{i+jr^f-jr^\ell,\ell}, \tag{9}$$

where $j\in\{0,1,\ldots,r-1\}$. That is, we can repair the bit $s_{i,f}$ by downloading one parity bit $s_{i+jr^f,k+j}$ from column $k+j$ and $k-1$ information bits from the surviving $k-1$ information columns, where $j\in\{0,1,\ldots,r-1\}$. For each failed bit in column f, we need to obtain k bits (one parity bit plus $k-1$ information bits) to recover it. There may exist some common information bits in repairing two or more bits in column f. For each failed bit, we need to carefully choose the parity column j to repair the bits by (9) such that the common bits are as many as possible, and thus we can reduce the most repair bandwidth of column f. Based on the above idea, we propose the repair algorithm in Algorithm 1.

Next theorem shows that the repair bandwidth of information column f is asymptotically optimal.

Algorithm 1 Repair algorithm for one information column failure of the first construction

```
1: Suppose that column $f$ fails, where $f \in \{0, 1, \ldots, k-1\}$.
2: for $i \bmod r^{f+1} \in \{0, 1, \ldots, r^f - 1\}$ do
3:     Repair the bit $s_{i,f}$ by (9) with $j = 0$.
4: for $t = 1, 2, \ldots, r-1$ do
5:     for $i \bmod r^{f+1} \in \{t \cdot r^f, 1 + t \cdot r^f, \ldots, (t+1) \cdot r^f - 1\}$ do
6:         Repair the bit $s_{i,f}$ by (9) with $j = r - t$.
7: return
```

Theorem 3. *Suppose that information column f fails, where $0 \le f \le k-1$. One can recover column f by Algorithm 1 and the repair bandwidth is*

$$\frac{(p-1)\tau(k+r-1)}{r} + \frac{(p-1)\tau(r^f-1)}{(r-1)r^f}. \tag{10}$$

Proof. The proof is given in Appendix A. □

By Theorem 3, the repair bandwidth of column 0 is $\frac{(p-1)(k+r-1)\tau}{r}$, which is optimal. When $f \ge 1$, we have that the repair bandwidth is

$$\frac{(p-1)(k+r-1)\tau}{r} + \sum_{\ell=0}^{f-1} r^\ell \frac{(p-1)\tau}{r^f} < \frac{(p-1)(k+r)\tau}{r},$$

which is strictly less than $\frac{k+r}{k+r-1}$ times of the optimal value. Therefore, the repair bandwidth of any information column is asymptotically optimal when $k+r$ is large enough.

B. Repair Algorithm for the Second Construction

Suppose that column f fails, where $0 \le f \le k-1$. In the following, we present a repair algorithm to repair column f by downloading bits from $k + \frac{r}{2} - 1$ columns.

Similar to the repair algorithm of the first construction, we assume that all the extra bits are known in the following. According to column j of the encoding matrix in (5), the parity bits $s_{i,k+j}$ for $i = 0, 1, \ldots, p\tau - 1$ are computed as

$$s_{i,k+j} = s_{i-j,0} + s_{i-j\frac{r}{2},1} + s_{i-j{\frac{r}{2}}^2,2} + \ldots + s_{i-j{\frac{r}{2}}^{k-2},k-2} + s_{i,k-1}, \tag{11}$$

when $j = 0, 1, \ldots, \frac{r}{2} - 1$, and

$$\begin{aligned} s_{i,k+j} = \; & s_{i,0} + s_{i-(r-1-j)(\frac{r}{2})^{k-2},1} + s_{i-(r-1-j)(\frac{r}{2})^{k-3},2} \\ & + \ldots + s_{i-(r-1-j)(\frac{r}{2}),k-2} + s_{i-(r-1-j),k-1}, \end{aligned} \tag{12}$$

when $j = \frac{r}{2}, \frac{r}{2}+1, \ldots, r-2$, and

$$\begin{aligned} s_{i,k+r-1} = \; & s_{i,0} + s_{i-(k-1)(\frac{r}{2})^{\frac{r}{2}},1} + s_{i-(k-2)(\frac{r}{2})^{\frac{r}{2}},2} \\ & + \ldots + s_{i-2(\frac{r}{2})^{\frac{r}{2}},k-2} + s_{i-(\frac{r}{2})^{\frac{r}{2}},k-1}. \end{aligned} \tag{13}$$

We can repair the bit $s_{i,f}$ stored in column f by

$$s_{i,f} = \begin{cases} s_{i+j{\frac{r}{2}}^f,k+j} + s_{i+j{\frac{r}{2}}^f,k-1} + \\ \sum_{\ell=0,\ell\neq f}^{k-2} s_{i+j{\frac{r}{2}}^f - j{\frac{r}{2}}^\ell,\ell}, 0 \le f \le k-2; \\ s_{i,k+j} + \sum_{\ell=0}^{k-2} s_{i-j(\frac{r}{2})^\ell,\ell}, f = k-1, \end{cases} \tag{14}$$

when $j \in \{0, 1, \ldots, \frac{r}{2} - 1\}$,

$$s_{i,f} = \begin{cases} s_{i+(r-1-j){\frac{r}{2}}^{k-1-f},k+j} + s_{i+(r-1-j){\frac{r}{2}}^{k-1-f},0} + \\ \sum_{\ell=1,\ell\neq f}^{k-1} s_{i+(r-1-j){\frac{r}{2}}^{k-1-f}-(r-1-j){\frac{r}{2}}^{k-1-\ell},\ell}, \\ 1 \le f \le k-1; \\ s_{i,k+j} + \sum_{\ell=1}^{k-1} s_{i-(r-1-j)(\frac{r}{2})^{k-1-\ell},\ell}, f = 0, \end{cases} \tag{15}$$

when $j \in \{\frac{r}{2}, \frac{r}{2}+1, \ldots, r-2\}$, and

$$s_{i,f} = \begin{cases} s_{i+(k-f){\frac{r}{2}}^{\frac{r}{2}},k+r-1} + s_{i+(k-f){\frac{r}{2}}^{\frac{r}{2}},0} + \\ \sum_{\ell=1,\ell\neq f}^{k-1} s_{i+(k-f){\frac{r}{2}}^{\frac{r}{2}}-(k-\ell){\frac{r}{2}}^{\frac{r}{2}},\ell}, 1 \le f \le k-1; \\ s_{i,k+r-1} + \sum_{\ell=1}^{k-1} s_{i-(k-\ell)(\frac{r}{2})^{\frac{r}{2}},\ell}, f = 0. \end{cases} \tag{16}$$

We can carefully choose the parity column j to repair each bit $s_{i,f}$ by (14), (15), or (16) to reduce the repair bandwidth. The repair algorithm is given in Algorithm 2.

Algorithm 2 Repair algorithm for one information column failure of the second construction

```
1: Suppose that column $f$ fails, where $f \in \{0, 1, \ldots, k-1\}$.
2: for $f \in \{0, 1, \ldots, \lceil\frac{k}{2}\rceil - 1\}$ do
3:     for $i \bmod (\frac{r}{2})^{f+1} \in \{0, 1, \ldots, (\frac{r}{2})^f - 1\}$ do
4:         Repair the bit $s_{i,f}$ by (14) with $j = 0$.
5:     for $t = 1, 2, \ldots, \frac{r}{2} - 1$ do
6:         for $i \bmod (\frac{r}{2})^{f+1} \in \{t \cdot (\frac{r}{2})^f, 1 + t \cdot (\frac{r}{2})^f, \ldots, (t+1) \cdot (\frac{r}{2})^f - 1\}$ do
7:             Repair the bit $s_{i,f}$ by (14) with $j = \frac{r}{2} - t$.
8: for $f \in \{\lceil\frac{k}{2}\rceil, \lceil\frac{k}{2}\rceil + 1, \ldots, k-1\}$ do
9:     for $i \bmod (\frac{r}{2})^{k-f} \in \{0, 1, \ldots, (\frac{r}{2})^{k-f-1} - 1\}$ do
10:        Repair the bit $s_{i,f}$ by (16).
11:    for $t = 1, 2, \ldots, \frac{r}{2} - 1$ do
12:        for $i \bmod (\frac{r}{2})^{k-f} \in \{t \cdot (\frac{r}{2})^{k-f-1}, 1 + t \cdot (\frac{r}{2})^{k-f-1}, \ldots, (t+1) \cdot (\frac{r}{2})^{k-f-1} - 1\}$ do
13:            Repair the bit $s_{i,f}$ by (15) with $j = \frac{r}{2} + t - 1$.
14: return
```

We present the repair bandwidth of information column f in the next theorem.

Theorem 4. *Suppose that information column f fails, where $0 \le f \le k-1$. We can recover column f by Algorithm 2 and the repair bandwidth is*

$$\frac{(p-1)\tau(k+\frac{r}{2}-1)}{\frac{r}{2}} + \frac{(p-1)\tau((\frac{r}{2})^{\min(f,k-f-1)} - 1)}{(\frac{r}{2}-1)(\frac{r}{2})^{\min(f,k-f-1)}}. \tag{17}$$

Proof. The proof is given in Appendix B. □

By Theorem 4, the repair bandwidth of column 0 is $\frac{(p-1)(k+\frac{r}{2}-1)\tau}{\frac{r}{2}}$, which is optimal. When $f \ge 1$, we have that the repair bandwidth is

$$\frac{(p-1)(k+\frac{r}{2}-1)\tau}{\frac{r}{2}} + \sum_{\ell=0}^{f-1} (\frac{r}{2})^\ell \frac{(p-1)\tau}{(\frac{r}{2})^f} < \frac{(p-1)(k+\frac{r}{2})\tau}{\frac{r}{2}},$$

which is strictly less than $\frac{k+\frac{r}{2}}{k+\frac{r}{2}-1}$ times of the optimal value. Therefore, the repair bandwidth of any information column is asymptotically optimal when $k + \frac{r}{2}$ is large enough.

IV. Transformed Codes with Asymptotically Optimal Repair for Any Column

In this section, we present two constructions of the transformed array codes that have optimal repair bandwidth for any parity column and asymptotically optimal repair bandwidth for any information column. The first constructed code is obtained by applying the transformation in [26], [28] for the array codes in Section II-B, and the second constructed code is obtained by applying the transformation in [28] for the codes in Section II-C.

We first review the transformation in [28] that can be employed for any (n,k) MDS code to obtain the transformed MDS code with optimal repair for each of the chosen $(d-k+1)\eta$ columns, where $\eta=\lfloor\frac{r-1}{d-k}\rfloor$.

A. Review of the Transformation in [28] for Codes over $R_{p\tau}$

Let $t=d-k+1$. In the transformed codes over $R_{p\tau}$, each column contains $t(p-1)\tau$ bits. For $j=0,1,\ldots,k-1$, let $t(p-1)\tau$ bits stored in column j be $s^{\ell}_{0,j},s^{\ell}_{1,j},\ldots,s^{\ell}_{(p-1)\tau-1,j}$ with $\ell=0,1,\ldots,t-1$. For $j=k,k+1,\ldots,k+r-1$, the $t(p-1)\tau$ bits in column j are $c^{\ell}_{0,j},c^{\ell}_{1,j},\ldots,c^{\ell}_{(p-1)\tau-1,j}$ with $\ell=0,1,\ldots,t-1$. We first compute τ extra bits $s^{\ell}_{(p-1)\tau,j},s^{\ell}_{(p-1)\tau+1,j},\ldots,s^{\ell}_{p\tau-1,j}$ by

$$s^{\ell}_{(p-1)\tau+\mu,j}=\sum_{i=0}^{p-2}s^{\ell}_{i\tau+\mu,j},\ \text{ for } \mu=0,1,\ldots,\tau-1,$$

and then represent $t(p-1)\tau$ bits stored in column j and the associated $t\tau$ extra bits by information polynomial

$$s^{\ell}_j(x)=s^{\ell}_{0,j}+s^{\ell}_{1,j}x+\ldots+s^{\ell}_{p\tau-1,j}x^{p\tau-1},$$

where $j=0,1,\ldots,k-1$ and $\ell=0,1,\ldots,t-1$. Similarly, we compute τ extra bits $c^{\ell}_{(p-1)\tau,j},c^{\ell}_{(p-1)\tau+1,j},\ldots,c^{\ell}_{p\tau-1,j}$ for each $(p-1)\tau$ bits in column j with $j=k,k+1,\ldots,k+r-1$ by

$$c^{\ell}_{(p-1)\tau+\mu,j}=\sum_{i=0}^{p-2}c^{\ell}_{i\tau+\mu,j},\ \text{ for } \mu=0,1,\ldots,\tau-1,$$

and represent the associated $p\tau$ bits by the parity polynomial

$$c^{\ell}_j(x)=c^{\ell}_{0,j}+c^{\ell}_{1,j}x+\ldots+c^{\ell}_{p\tau-1,j}x^{p\tau-1}.$$

Given k information polynomials $s^{\ell}_0(x),s^{\ell}_1(x),\ldots,s^{\ell}_{k-1}(x)$, we can obtain a codeword $s^{\ell}_0(x),s^{\ell}_1(x),\ldots,s^{\ell}_{n-1}(x)$ of an (n,k) MDS code over $R_{p\tau}$ with a specific construction such as the construction in Section II-B or in Section II-C, where $\ell=0,1,\ldots,t-1$. We term the last r polynomials $s^{\ell}_k(x),s^{\ell}_{k+1}(x),\ldots,s^{\ell}_{n-1}(x)$ as *intermediate polynomials*.

Recall that the information polynomial $s^{\ell}_j(x)$ is in $C_{p\tau}$ and the r intermediate polynomials are computed by the multiplication of k information polynomials and the $k\times r$ encoding matrix over $R_{p\tau}$ for each ℓ. We can show that each intermediate polynomial is also in $C_{p\tau}$.

In the transformed codes [28], each column contains t polynomials and the repair bandwidth of each of the last $t\eta$ columns is optimal, where $\eta=\lfloor\frac{r-1}{d-k}\rfloor$. Note that when we say a column contains or stores a polynomial in $R_{p\tau}$ in this section, it means that the first $(p-1)\tau$ coefficients of the polynomial are stored in the column.

For $i=0,1,\ldots,t-1$ and $j=0,1,\ldots,\eta-1$, column $n-t\eta+jt+i$ contains the following t polynomials

$$\begin{aligned}
c^{0}_{n-t\eta+jt+i}(x)=&\ s^{0}_{n-t\eta+jt+i}(x)+s^{i}_{n-t\eta+jt}(x),\\
c^{1}_{n-t\eta+jt+i}(x)=&\ s^{1}_{n-t\eta+jt+i}(x)+s^{i}_{n-t\eta+jt+1}(x),\ldots,\\
c^{i-1}_{n-t\eta+jt+i}(x)=&\ s^{i-1}_{n-t\eta+jt+i}(x)+s^{i}_{n-t\eta+jt+i-1}(x),\\
c^{i}_{n-t\eta+jt+i}(x)=&\ s^{i}_{n-t\eta+jt+i}(x),\\
c^{i+1}_{n-t\eta+jt+i}(x)=&\ s^{i+1}_{n-t\eta+jt+i}(x)+x^{e_j}s^{i}_{n-t\eta+jt+i+1}(x),\\
c^{i+2}_{n-t\eta+jt+i}(x)=&\ s^{i+2}_{n-t\eta+jt+i}(x)+x^{e_j}s^{i}_{n-t\eta+jt+i+2}(x),\ldots,\\
c^{t-1}_{n-t\eta+jt+i}(x)=&\ s^{t-1}_{n-t\eta+jt+i}(x)+x^{e_j}s^{i}_{n-t\eta+jt+t-1}(x),
\end{aligned}\tag{18}$$

where $e_j\neq 0$. For $h=0,1,\ldots,n-t\eta-1$, column h stores t polynomials

$$s^{0}_h(x),s^{1}_h(x),\ldots,s^{t-1}_h(x).$$

The above obtained codes are called *transformed codes*. Note that our transformed codes and the transformed codes in [28] are essentially the same codes, the difference is that the transformed codes in [28] are operated over the finite field $\mathbb{F}_q$ and our transformed codes are operated over the ring $R_{p\tau}$. If $\eta=1$ and we replace the (n,k) MDS code over $R_{p\tau}$ by (n,k) EVENODD code, then the obtained transformed code is reduced to the transformed EVENODD code in [26].

With the same proof of Theorem 4 in [28], we can show that the transformed codes satisfy the MDS property if the underlying codes over $R_{p\tau}$ satisfy the MDS property and p is large enough such that $2^{\deg(f_1(x))}$ is larger than

$$\eta(\frac{t}{2}-1)\frac{t}{2}\left(\binom{n}{k}-\sum_{\ell=0}^{\eta}\binom{n-\eta t}{k-\ell t}\cdot\binom{\eta}{\ell}\right).$$

We can also show that the repair bandwidth of each of the first ηt columns is optimal, with similar proof of Theorem 3 in [28].

B. The First Transformed Codes

We can obtain the first transformed codes by applying the transformation in Section IV-A for the codes in Section II-B. In the first transformed codes, each parity column has optimal repair bandwidth with $d=k+r-1$. Therefore, we have $\eta=1$ and $t=r$.

According to the construction of the transformed codes in Section IV-A, we need to first create $t=r$ instances of the codes in Section II-B. For each ℓ, the r intermediate polynomials $s^{\ell}_k(x),s^{\ell}_{k+1}(x),\ldots,s^{\ell}_{k+r-1}(x)$ are computed by

$$\begin{aligned}&\begin{bmatrix}s^{\ell}_k(x) & s^{\ell}_{k+1}(x) & \cdots & s^{\ell}_{k+r-1}(x)\end{bmatrix}\\
=&\begin{bmatrix}s^{\ell}_0(x) & s^{\ell}_1(x) & \cdots & s^{\ell}_{k-1}(x)\end{bmatrix}\cdot\mathbf{P}_{k\times r},\end{aligned}$$

over $R_{p\tau}$, where $\mathbf{P}_{k\times r}$ is the matrix in (4). According to (18), for $j=k,k+1,\ldots,k+r-1$, the r parity polynomials $c^{0}_j(x),c^{1}_j(x),\ldots,c^{r-1}_j(x)$ stored in column j are

$$\begin{aligned}
&c^{i}_j(x)=s^{i}_j(x)+s^{j-k}_{k+i}(x),\ \text{ for } i=0,1,\ldots,j-k-1,\\
&c^{j-k}_j(x)=s^{j-k}_j(x),\\
&c^{i}_j(x)=x^{\tau}s^{i}_j(x)+s^{j-k}_{i+k}(x),\ \text{ for } i=j-k+1,\ldots,r-1,
\end{aligned}\tag{19}$$

with $x^{e_0} = x^{\tau}$. For $j = 0, 1, \ldots, k-1$, column j stores r information polynomials $s_j^0(x), s_j^1(x), \ldots, s_j^{r-1}(x)$. Similar to the discussion in Section II-B, we have each parity polynomial $c_j^{\ell}(x)$ being in $C_{p\tau}$.

Table III shows the r parity polynomials stored in each parity column. The resultant array codes are called *the first transformed codes*. Similar to the proof of Theorem 1 in [26] and Theorem 4 in [28], we can show that the first transformed codes have the MDS property, if the array codes in Section II-B have the MDS property. We show that the first transformed codes have optimal repair bandwidth for any parity column in the next theorem.

Theorem 5. *The repair bandwidth of column j for $j = k, k+1, \ldots, k+r-1$ of the first transformed codes is optimal.*

Proof. The proof is similar to that in Theorem 3 in [28]. We can repair column j by downloading k information polynomials $s_0^{j-k}(x), s_1^{j-k}(x), \ldots, s_{k-1}^{j-k}(x)$ from k information columns and $r-1$ parity polynomials

$$x^{\tau} s_k^{j-k}(x) + s_j^0(x), \ldots, x^{\tau} s_{j-1}^{j-k}(x) + s_j^{j-k-1}(x),$$
$$s_{j+1}^{j-k}(x) + s_j^{j-k+1}(x), \ldots, s_{k+r-1}^{j-k}(x) + s_j^{r-1}(x),$$

from the other $r-1$ surviving parity columns, and the repair bandwidth is optimal.

We can first compute $s_k^{j-k}(x), s_{k+1}^{j-k}(x), \ldots, s_{k+r-1}^{j-k}(x)$ from the downloaded k information polynomials $s_0^{j-k}(x)$, $s_1^{j-k}(x), \ldots, s_{k-1}^{j-k}(x)$. As one polynomial $s_j^{j-k}(x)$ is known, we can recover the other $r-1$ parity polynomials by

$$\begin{aligned}
s_j^0(x) + s_k^{j-k}(x) =& (1+x^{\tau}) s_k^{j-k}(x) + (x^{\tau} s_k^{j-k}(x) + s_j^0(x)),\\
&\vdots\\
s_j^{j-k-1}(x) + s_{j-1}^{j-k}(x) =& (1+x^{\tau}) s_{j-1}^{j-k}(x) +\\
& (x^{\tau} s_{j-1}^{j-k}(x) + s_j^{j-k-1}(x)),\\
x^{\tau} s_j^{j-k+1}(x) + s_{j+1}^{j-k}(x) =& (1+x^{\tau}) s_j^{j-k+1}(x) +\\
& (s_{j+1}^{j-k}(x) + s_j^{j-k+1}(x)),\\
&\vdots\\
x^{\tau} s_j^{r-1}(x) + s_{k+r-1}^{j-k}(x) =& (1+x^{\tau}) s_j^{r-1}(x) + (s_{k+r-1}^{j-k}(x) + s_j^{r-1}(x)).
\end{aligned}$$

□

Note that the sizes of information bits in the transformed codes are different from those in the array codes given in Section II-B. In order for fair comparison, we define the normalized repair bandwidth as the ratio of repair bandwidth to all information bits. Next, we show that the repair bandwidth of any information column is asymptotically optimal.

Theorem 6. *The normalized repair bandwidth of column f of the first transformed codes is the same as that of the array codes in Section II-B, where $f = 0, 1, \ldots, k-1$.*

Proof. The proof is given in Appendix C. □

Take an example of $k = 2$ and $r = 2$ to illustrate the first transformed codes. In the example, we have 32 information bits that are $s_{i,j}^{\ell}$ for $\ell = 0, 1$, $i = 0, 1, \ldots, 7$ and $j = 0, 1$. For $\ell = 0, 1$ and $j = 0, 1$, we represent the eight information bits $s_{0,j}^{\ell}, s_{1,j}^{\ell}, \ldots, s_{7,j}^{\ell}$ and four extra bits $s_{8,j}^{\ell}, s_{9,j}^{\ell}, s_{10,j}^{\ell}, s_{11,j}^{\ell}$ by polynomial

$$s_j^{\ell}(x) = s_{0,j}^{\ell} + s_{1,j}^{\ell} x + \ldots + s_{11,j}^{\ell} x^{11}.$$

We first compute the polynomials $s_2^{\ell}(x)$ and $s_3^{\ell}(x)$ by

$$\begin{bmatrix} s_2^{\ell}(x) & s_3^{\ell}(x) \end{bmatrix} = \begin{bmatrix} s_0^{\ell}(x) & s_1^{\ell}(x) \end{bmatrix} \begin{bmatrix} 1 & x \\ 1 & x^2 \end{bmatrix},$$

over $R_{3\cdot 4}$ for $\ell = 0, 1$, and then store the eight coefficients of degrees from zero to seven of the polynomials $s_2^0(x), x^4 s_2^1(x) + s_3^0(x)$ and $s_3^0(x) + s_2^1(x), s_3^1(x)$ in column 2 and column 3, respectively. Table IV shows the example of the first transformed codes.

We show that the efficient repairing procedure of any one information column of the example in Table I is preserved in the example in Table IV. We can repair column 0 by downloading 24 bits

$$\begin{aligned}
& s_{0,1}^0, s_{2,1}^0, s_{4,1}^0, s_{6,1}^0, s_{0,1}^1, s_{2,1}^1, s_{4,1}^1, s_{6,1}^1,\\
& s_{0,2}^0, s_{2,2}^0, s_{4,2}^0, s_{6,2}^0, s_{0,3}^1, s_{2,3}^1, s_{4,3}^1, s_{6,3}^1,\\
& s_{0,3}^0 + s_{8,2}^1, s_{2,3}^0 + s_{10,2}^1, s_{4,3}^0 + s_{0,2}^1, s_{6,3}^0 + s_{2,2}^1,\\
& s_{0,3}^0 + s_{0,2}^1, s_{2,3}^0 + s_{2,2}^1, s_{4,3}^0 + s_{4,2}^1, s_{6,3}^0 + s_{6,2}^1.
\end{aligned}$$

We can compute

$$\begin{aligned}
s_{4,2}^1 =& (s_{0,3}^0 + s_{8,2}^1) + (s_{0,3}^0 + s_{0,2}^1),\\
s_{6,2}^1 =& (s_{2,3}^0 + s_{10,2}^1) + (s_{2,3}^0 + s_{2,2}^1),\\
s_{8,2}^1 =& (s_{4,3}^0 + s_{0,2}^1) + (s_{4,3}^0 + s_{4,2}^1),\\
s_{10,2}^1 =& (s_{6,3}^0 + s_{2,2}^1) + (s_{6,3}^0 + s_{6,2}^1),
\end{aligned}$$

and further compute

$$\begin{aligned}
s_{4,3}^0 =& s_{4,2}^1 + (s_{4,3}^0 + s_{4,2}^1),\\
s_{6,3}^0 =& s_{6,2}^1 + (s_{6,3}^0 + s_{6,2}^1),\\
s_{0,3}^0 =& s_{8,2}^1 + (s_{0,3}^0 + s_{8,2}^1),\\
s_{2,3}^0 =& s_{10,2}^1 + (s_{2,3}^0 + s_{10,2}^1).
\end{aligned}$$

Then, we can recover the bits $s_{0,0}^0, s_{0,0}^1, s_{2,0}^0, s_{2,0}^1, s_{4,0}^0$, $s_{4,0}^1, s_{6,0}^0, s_{6,0}^1$ in column 0 by

$$\begin{aligned}
s_{0,0}^0 =& s_{0,1}^0 + s_{0,2}^0,\\
s_{0,0}^1 =& s_{0,1}^1 + s_{4,2}^1 + s_{8,2}^1,\\
s_{2,0}^0 =& s_{2,1}^0 + s_{2,2}^0,\\
s_{2,0}^1 =& s_{2,1}^1 + s_{6,2}^1 + s_{10,2}^1,\\
s_{4,0}^0 =& s_{4,1}^0 + s_{4,2}^0,\\
s_{4,0}^1 =& s_{4,1}^1 + s_{4,2}^1,\\
s_{6,0}^0 =& s_{6,1}^0 + s_{6,2}^0,\\
s_{6,0}^1 =& s_{6,1}^1 + s_{6,2}^1,
\end{aligned}$$

TABLE III
THE r PARITY POLYNOMIALS STORED IN EACH PARITY COLUMN.

Column k	Column $k+1$	Column $k+2$	$\cdots$	Column $k+r-1$
$s_k^0(x)$	$s_{k+1}^0(x)+s_k^1(x)$	$s_{k+2}^0(x)+s_k^2(x)$	$\cdots$	$s_{k+r-1}^0(x)+s_k^{r-1}(x)$
$x^\tau s_k^1(x)+s_{k+1}^0(x)$	$s_{k+1}^1(x)$	$s_{k+2}^1(x)+s_{k+1}^2(x)$	$\cdots$	$s_{k+r-1}^1(x)+s_{k+1}^{r-1}(x)$
$x^\tau s_k^2(x)+s_{k+2}^0(x)$	$x^\tau s_{k+1}^2(x)+s_{k+2}^1(x)$	$s_{k+2}^2(x)$	$\cdots$	$s_{k+r-1}^2(x)+s_{k+2}^{r-1}(x)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$x^\tau s_k^{r-1}(x)+s_{k+r-1}^0(x)$	$x^\tau s_{k+1}^{r-1}(x)+s_{k+r-1}^1(x)$	$x^\tau s_{k+2}^{r-1}(x)+s_{k+r-1}^2(x)$	$\cdots$	$s_{k+r-1}^{r-1}(x)$

TABLE IV
THE EXAMPLE OF THE TRANSFORMED CODES WITH $k=2$ AND $r=2$.

Column 0	Column 1	Column 2	Column 3
$s_{0,0}^0$ $s_{0,0}^1$	$s_{0,1}^0$ $s_{0,1}^1$	$s_{0,2}^0=s_{0,0}^0+s_{0,1}^0$ $s_{0,3}^0+s_{8,2}^1=s_{11,0}^0+s_{10,1}^0+s_{8,0}^1+s_{8,1}^1$	$s_{0,3}^0+s_{0,2}^1=s_{11,0}^0+s_{10,1}^0+s_{0,0}^1+s_{0,1}^1$ $s_{0,3}^1=s_{11,0}^1+s_{10,1}^1$
$s_{1,0}^0$ $s_{1,0}^1$	$s_{1,1}^0$ $s_{1,1}^1$	$s_{1,2}^0=s_{1,0}^0+s_{1,1}^0$ $s_{1,3}^0+s_{9,2}^1=s_{0,0}^0+s_{11,1}^0+s_{9,0}^1+s_{9,1}^1$	$s_{1,3}^0+s_{1,2}^1=s_{0,0}^0+s_{11,1}^0+s_{1,0}^1+s_{1,1}^1$ $s_{1,3}^1=s_{0,0}^1+s_{11,1}^1$
$s_{2,0}^0$ $s_{2,0}^1$	$s_{2,1}^0$ $s_{2,1}^1$	$s_{2,2}^0=s_{2,0}^0+s_{2,1}^0$ $s_{2,3}^0+s_{10,2}^1=s_{1,0}^0+s_{0,1}^0+s_{10,0}^1+s_{10,1}^1$	$s_{2,3}^0+s_{2,2}^1=s_{1,0}^0+s_{0,1}^0+s_{2,0}^1+s_{2,1}^1$ $s_{2,3}^1=s_{1,0}^1+s_{0,1}^1$
$s_{3,0}^0$ $s_{3,0}^1$	$s_{3,1}^0$ $s_{3,1}^1$	$s_{3,2}^0=s_{3,0}^0+s_{3,1}^0$ $s_{3,3}^0+s_{11,2}^1=s_{2,0}^0+s_{1,1}^0+s_{11,0}^1+s_{11,1}^1$	$s_{3,3}^0+s_{3,2}^1=s_{2,0}^0+s_{1,1}^0+s_{3,0}^1+s_{3,1}^1$ $s_{3,3}^1=s_{2,0}^1+s_{1,1}^1$
$s_{4,0}^0$ $s_{4,0}^1$	$s_{4,1}^0$ $s_{4,1}^1$	$s_{4,2}^0=s_{4,0}^0+s_{4,1}^0$ $s_{4,3}^0+s_{0,2}^1=s_{3,0}^0+s_{2,1}^0+s_{0,0}^1+s_{0,1}^1$	$s_{4,3}^0+s_{4,2}^1=s_{3,0}^0+s_{2,1}^0+s_{4,0}^1+s_{4,1}^1$ $s_{4,3}^1=s_{3,0}^1+s_{2,1}^1$
$s_{5,0}^0$ $s_{5,0}^1$	$s_{5,1}^0$ $s_{5,1}^1$	$s_{5,2}^0=s_{5,0}^0+s_{5,1}^0$ $s_{5,3}^0+s_{1,2}^1=s_{4,0}^0+s_{3,1}^0+s_{1,0}^1+s_{1,1}^1$	$s_{5,3}^0+s_{5,2}^1=s_{4,0}^0+s_{3,1}^0+s_{5,0}^1+s_{5,1}^1$ $s_{5,3}^1=s_{4,0}^1+s_{3,1}^1$
$s_{6,0}^0$ $s_{6,0}^1$	$s_{6,1}^0$ $s_{6,1}^1$	$s_{6,2}^0=s_{6,0}^0+s_{6,1}^0$ $s_{6,3}^0+s_{2,2}^1=s_{5,0}^0+s_{4,1}^0+s_{2,0}^1+s_{2,1}^1$	$s_{6,3}^0+s_{6,2}^1=s_{5,0}^0+s_{4,1}^0+s_{6,0}^1+s_{6,1}^1$ $s_{6,3}^1=s_{5,0}^1+s_{4,1}^1$
$s_{7,0}^0$ $s_{7,0}^1$	$s_{7,1}^0$ $s_{7,1}^1$	$s_{7,2}^0=s_{7,0}^0+s_{7,1}^0$ $s_{7,3}^0+s_{3,2}^1=s_{6,0}^0+s_{5,1}^0+s_{3,0}^1+s_{3,1}^1$	$s_{7,3}^0+s_{7,2}^1=s_{6,0}^0+s_{5,1}^0+s_{7,0}^1+s_{7,1}^1$ $s_{7,3}^1=s_{6,0}^1+s_{5,1}^1$

and repair $s_{1,0}^0, s_{1,0}^1, s_{3,0}^0, s_{3,0}^1, s_{5,0}^0, s_{5,0}^1, s_{7,0}^0, s_{7,0}^1$ by

$$\begin{aligned}
s_{1,0}^0 =& s_{0,1}^0 + s_{2,3}^0,\\
s_{1,0}^1 =& s_{0,1}^1 + s_{2,3}^1,\\
s_{3,0}^0 =& s_{2,1}^0 + s_{4,3}^0,\\
s_{3,0}^1 =& s_{2,1}^1 + s_{4,3}^1,\\
s_{5,0}^0 =& s_{4,1}^0 + s_{6,3}^0,\\
s_{5,0}^1 =& s_{4,1}^1 + s_{6,3}^1,\\
s_{7,0}^0 =& s_{6,1}^0 + s_{0,3}^0 + s_{4,3}^0,\\
s_{7,0}^1 =& s_{6,1}^1 + s_{0,3}^1 + s_{4,3}^1.
\end{aligned}$$

C. The Second Transformed Codes

The second transformed codes are obtained by applying the transformation in Section IV-A for the array codes in Section II-C. In the second transformed array codes, we have k information columns and r parity columns, where $r \geq 4$ is an even number. Each parity column has optimal repair bandwidth with $d = k+\frac{r}{2}-1$. We thus have $t = d-k+1 = \frac{r}{2}$ and $\eta = \left\lfloor \frac{r-1}{d-k} \right\rfloor = 2$.

We should first create $\frac{r}{2}$ instances of the codes in Section II-C and then store $\frac{r}{2}$ polynomials for each column according to the transformation in Section IV-A. For $\ell = 0, 1, \ldots, \frac{r}{2}-1$, we compute the intermediate polynomials $s_k^\ell(x), s_{k+1}^\ell(x), \ldots, s_{k+r-1}^\ell(x)$ by

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

over $R_{p\tau}$, where $\mathbf{P}_{k\times r}$ is the matrix in (5). According to (18), for $i = 0, 1, \ldots, \frac{r}{2}-1$ and $j = 0, 1$, the $\frac{r}{2}$ parity polynomials $c_j^0(x), c_j^1(x), \ldots, c_j^{\frac{r}{2}-1}(x)$ stored in column $k+j\frac{r}{2}+i$ are

$$\begin{aligned}
c_{k+j\frac{r}{2}+i}^\ell(x) =& \quad s_{k+j\frac{r}{2}+i}^\ell + s_{k+j\frac{r}{2}+\ell}^i, \text{ for } \ell = 0, 1, \ldots, i-1,\\
c_{k+j\frac{r}{2}+i}^i(x) =& \quad s_{k+j\frac{r}{2}+i}^i(x),\\
c_{k+j\frac{r}{2}+i}^\ell(x) =& \quad x^\tau s_{k+j\frac{r}{2}+i}^\ell(x) + s_{k+j\frac{r}{2}+\ell}^i(x)\\
& \quad \text{for } \ell = i+1, i+2, \ldots, \tfrac{r}{2}-1,
\end{aligned} \tag{20}$$

with $x^{e_j} = x^\tau$. Column j stores $\frac{r}{2}$ information polynomials $s_j^0(x), s_j^1(x), \ldots, s_j^{\frac{r}{2}-1}(x)$, where $j = 0, 1, \ldots, k-1$.

The above array codes are called *the second transformed codes*. Similar to the proof of Theorem 7 in [18], we can show that the second transformed codes have the MDS property, if $2^{\deg(f_1(x))}$ is larger than

$$(\frac{r}{2}-1)\frac{r}{2}\left(\binom{n}{k} - \sum_{\ell=0}^{2}\binom{n-r}{k-\frac{r}{2}\ell}\cdot\binom{2}{\ell}\right). \tag{21}$$

Although the value in (21) is exponentially increasing with k and r, we can choose a small prime p for a given r. When $\tau = (\frac{r}{2})^{\lceil \frac{k}{2} \rceil}$ is a power of 2, we have

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

If p is a prime number with 2 be a primitive element in $\mathbb{F}_p$, then the polynomial $1 + x + \cdots + x^{p-1}$ is irreducible. Therefore, when $\frac{r}{2}$ is a power of 2, the second transformed codes are MDS codes, if 2^p is larger than the value in (21) and p is a prime number with 2 be a primitive element in the field $\mathbb{F}_p$. For example, when $r = 4$, the value in (21) is strictly less than 2^n, then the code is MDS code when $p \geq n$ and p is a prime number with 2 be a primitive element in the field $\mathbb{F}_p$.

We present that the second transformed codes have optimal repair bandwidth for any parity column in the next theorem.

Theorem 7. *For $i = 0, 1, \ldots, \frac{r}{2} - 1$ and $j = 0, 1$, the repair bandwidth of column $k + j\frac{r}{2} + i$ of the second transformed codes is optimal.*

Proof. See the proof of Theorem 3 in [28]. □

Note that the d helper columns of the transformed codes in [28] are specific, so the d helper columns of our second transformed codes are also specific. Next theorem shows that the repair bandwidth of any information column is asymptotically optimal.

Theorem 8. *The normalized repair bandwidth of column f of the second transformed codes is the same as that of the array codes in Section II-C, where $f = 0, 1, \ldots, k - 1$.*

Proof. Although the parity polynomials stored in parity columns are linear combinations of some intermediate polynomials and the needed intermediate bits are mixed together, with the same proof of Theorem 6, we can first compute all the needed intermediate bits from the downloaded parity bits and then recover the failed information bits in column f with the same normalized repair bandwidth as the codes in Section II-C. □

Consider the second transformed code with $k = 4$, $r = 4$, $\tau = 4$ and $p = 5$. In the example, we have 128 information bits that are $s_{i,j}^{\ell}$ for $\ell = 0, 1$, $i = 0, 1, \ldots, 15$ and $j = 0, 1, 2, 3$. For $\ell = 0, 1$ and $j = 0, 1, 2, 3$, we represent the 16 information bits $s_{0,j}^{\ell}, s_{1,j}^{\ell}, \ldots, s_{15,j}^{\ell}$ and four extra bits $s_{16,j}^{\ell}, s_{17,j}^{\ell}, s_{18,j}^{\ell}, s_{19,j}^{\ell}$ by polynomial

$$s_j^{\ell}(x) = s_{0,j}^{\ell} + s_{1,j}^{\ell}x + \ldots + s_{19,j}^{\ell}x^{19}.$$

We first compute $s_4^{\ell}(x), s_5^{\ell}(x), s_6^{\ell}(x), s_7^{\ell}(x)$ by

$$\begin{aligned} &\begin{bmatrix} s_4^{\ell}(x) & s_5^{\ell}(x) & s_6^{\ell}(x) & s_7^{\ell}(x) \end{bmatrix} \\ =& \begin{bmatrix} s_0^{\ell}(x) & s_1^{\ell}(x) & s_2^{\ell}(x) & s_3^{\ell}(x) \end{bmatrix} \begin{bmatrix} 1 & x & 1 & 1 \\ 1 & x^2 & x^4 & x^{12} \\ 1 & x^4 & x^2 & x^8 \\ 1 & 1 & x & x^4 \end{bmatrix} \end{aligned}$$

over $R_{5\cdot 4}$ for $\ell = 0, 1$. Table V shows the example of the second transformed code. Note that we only store the 16 coefficients of degrees from zero to 15 of the polynomials in Table V.

We can repair column 4 by downloading the first 16 coefficients from each of the five polynomials

$$s_0^0(x), s_1^0(x), s_2^0(x), s_3^0(x), s_5^0(x) + s_4^1(x),$$

as we can first compute $s_4^0(x), s_5^0(x)$ from $s_0^0(x), s_1^0(x), s_2^0(x), s_3^0(x)$ and then compute $x^4 s_4^1(x) + s_5^0(x)$ by $x^4(s_5^0(x) + s_4^1(x)) + (1 + x^4)s_5^0(x)$. Similarly, we can repair column 5, column 6 and column 7 by downloading the first 16 coefficients from each of polynomials

$$s_0^1(x), s_1^1(x), s_2^1(x), s_3^1(x), x^4 s_4^1(x) + s_5^0(x),$$

$$s_0^0(x), s_1^0(x), s_2^0(x), s_3^0(x), s_7^0(x) + s_6^1(x),$$

and

$$s_0^1(x), s_1^1(x), s_2^1(x), s_3^1(x), x^4 s_6^1(x) + s_7^0(x),$$

respectively.

V. Comparisons

In this section, we compare our two transformed codes and the existing binary MDS array codes with exactly or asymptotically optimal repair bandwidth for any column in [18], [26], in terms of sub-packetization, encoding complexity, supported parameters and repair bandwidth.

We define the encoding complexity as the ratio of total number of XORs required in the encoding procedure to the number of parity bits. In the first transformed codes, each column contains r polynomials. By construction, we should compute $rk\tau$ extra bits that takes $rk\tau(p-2)$ XORs. Then generating r^2 intermediate polynomials takes $r^2 p\tau(k-1)$ XORs. Finally, we compute r^2 parity polynomials stored in r parity columns by (19) that takes $r(r-1)p\tau$ XORs. Therefore, the encoding complexity of the first transformed codes is

$$\frac{rk\tau(p-2) + r^2 p\tau(k-1) + r(r-1)p\tau}{r^2(p-1)\tau} \approx \frac{(r+1)k}{r} - \frac{1}{r}.$$

Similarly, we can calculate that the encoding complexity of the second transformed codes is

$$\frac{rk\tau(p-2) + r^2 p\tau(k-1) + r(r-1)p\tau}{r^2(p-1)\tau} \approx \frac{(r+1)k}{r} - \frac{2}{r}.$$

Table VI shows the comparison of existing binary MDS array codes with exact or asymptotical optimal repair bandwidth with two transformed codes. The results in Table VI show that the two transformed codes have less encoding complexity than the codes in [26] and the second codes in [18].

Note that p should be large enough for the second codes in [18] and the proposed two transformed codes, while $p \geq n$ is sufficient for codes in [26]. However, when r is small, we can choose a small value of p for the proposed two transformed codes. For example, when $r = 3$, $p = 3$ and τ is a power of p, the proposed first transformed code is MDS code for $k \geq 2$. In the following, we consider the sub-packetization for general parameters and we assume that the value of p in codes in [26] is smaller than that for both the second codes in [18] and the proposed two transformed codes. We also assume that the value of p in the second codes in [18] and the proposed two transformed codes is the same. Compared with the second codes in [18], the proposed second transformed codes not only have much less encoding complexity but also smaller sub-packetization. Compared with the codes in [26], the proposed two transformed codes have larger sub-packetization. Note that the proposed first transformed codes can support all parameters $n \geq k + 2$ with $d = n - 1$ and the supported parameters of the proposed second transformed codes should satisfy that r is even number and $d = k + \frac{r}{2} - 1$. The codes in [26] can support all the parameters $k < d < n$. Moreover, the repair bandwidths of the proposed two transformed codes are asymptotically optimal, while the codes in [26] have optimal

TABLE V
THE EXAMPLE OF THE SECOND TRANSFORMED CODE WITH $k = 4$ AND $r = 4$.

Column 0	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7
$s_0^0(x)$	$s_1^0(x)$	$s_2^0(x)$	$s_3^0(x)$	$s_4^0(x)$	$s_5^0(x)+s_4^1(x)$	$s_6^0(x)$	$s_7^0(x)+s_6^1(x)$
$s_0^1(x)$	$s_1^1(x)$	$s_2^1(x)$	$s_3^1(x)$	$x^4s_4^1(x)+s_5^0(x)$	$s_5^1(x)$	$x^4s_6^1(x)+s_7^0(x)$	$s_7^1(x)$

TABLE VI
COMPARISON OF BINARY MDS ARRAY CODES WITH EXACTLY OR ASYMPTOTICALLY OPTIMAL REPAIR BANDWIDTH FOR ANY COLUMN, WHERE $\eta = d-k+1$ FOR THE CODES IN [26].

Codes	Sub-packetization	Encoding Complexity	Parameters (n,k,d)	Repair bandwidth
Second code in [18]	$(p-1)\cdot(\frac{r}{2})^{d-1}$	$\frac{(r+1)k}{r}-1+\frac{rp\tau}{2}$	r is even, $d=k+\frac{r}{2}-1$	asym. opt. for all columns
Code in [26]	$(p-1)(\eta)^{\lceil\frac{k}{\eta}\rceil+\lceil\frac{r}{\eta}\rceil}$	$k-1+\frac{3n}{2r}+\frac{k-2}{p-1}$	$k<d<n$	opt. for all columns
Proposed code I	$(p-1)(r)^{k+1}$	$\frac{(r+1)k}{r}-\frac{1}{r}$	$n\ge k+2, d=n-1$	asym. opt. for all columns
Proposed code II	$(p-1)(\frac{r}{2})^{\lceil\frac{k}{2}\rceil+1}$	$\frac{(r+1)k}{r}-\frac{2}{r}$	r is even, $d=k+\frac{r}{2}-1$	asym. opt. for all columns

repair bandwidth. However, the encoding complexity of the proposed two transformed codes is less than that of the codes in [26].

VI. CONCLUSION

In this paper, we propose two constructions of binary MDS array codes that have asymptotically optimal repair bandwidth for any information column. By applying the transformation in [28] for the proposed binary MDS array codes, we obtain two transformed codes that have asymptotically optimal repair bandwidth for any information column and optimal repair bandwidth for any parity column. The implementation of the proposed two transformed codes in practical distributed storage systems is one of our future work. Furthermore, how to design new transformation for codes in [17], [19] to enable optimal repair bandwidth for any single parity column and asymptotically optimal repair bandwidth for any single information column is also a future work.

APPENDIX A
PROOF OF THEOREM 3

Proof. We first show that Algorithm 1 can recover all information bits in column f. For each $i \in \{0,1,\ldots,(p-1)\tau-1\}$, we have

$$i \bmod r^{f+1} \in \{t\cdot r^f, 1+t\cdot r^f, \ldots, (t+1)\cdot r^f-1\}$$

for a given t with $0 \le t \le r-1$. By Algorithm 1, we can recover each bit $s_{i,f}$ with either $j=0$ or $j=r-t$, and thus we can recover all the bits in column f by Algorithm 1.

Then, we calculate the repair bandwidth of column f. According to Algorithm 1, we recover the bits $s_{i,f}$ by (9), i.e., by downloading one parity bit $s_{i+(r-t)r^f,k+t}$ and $k-1$ information bits $s_{i+(r-t)r^f-(r-t)r^\ell,\ell}$ with $\ell = 0,1,\ldots,f-1,f+1,\ldots,k-1$ and i satisfying

$$i \bmod r^{f+1} \in \{t\cdot r^f, 1+t\cdot r^f, \ldots, (t+1)\cdot r^f-1\} \quad (22)$$

for $t = 1,2,\ldots,r-1$. If

$$i \bmod r^{f+1} \in \{0,1,\ldots,r^f-1\}, \quad (23)$$

we recover the bits $s_{i,f}$ by downloading one parity bit $s_{i,k}$ and $k-1$ information bits $s_{i,0}, s_{i,1}, \ldots, s_{i,f-1}, s_{i,f+1}, \ldots, s_{i,k-1}$.

Recall that $(p-1)\tau$ is a multiple of r^k and also a multiple of r^{f+1} for $f = 0,1,\ldots,k-1$, and thus $i \bmod r^{f+1}$ is uniformed distributed over $\{0,1,\ldots,r^{f+1}-1\}$.

We can first download $k(p-1)\tau/r$ bits $s_{i,\ell}$ with $\ell = 0,1,\ldots,f-1,f+1,\ldots,k-1$ and i satisfying (23) by steps 2 and 3 in Algorithm 1. Then, we only need to download the bits in steps 5 and 6 which have not downloaded in steps 2 and 3. We first consider the bits $s_{i+(r-t)r^f-(r-t),0}$ downloaded from column 0. If $i \bmod r^{f+1} = t\cdot r^f$, then there exists an integer m such that $i = m\cdot r^{f+1}+t\cdot r^f$ and

$$\begin{aligned}&(i+(r-t)r^f-(r-t)) \bmod r^{f+1}\\ =&((m\cdot r^{f+1}+t\cdot r^f+(r-t)r^f-(r-t))) \bmod r^{f+1}\\ =&r^{f+1}-(r-t) \bmod r^{f+1}.\end{aligned}$$

By repeating the above procedure for $i \bmod r^{f+1} = t\cdot r^f, 1+t\cdot r^f, \ldots, (t+1)\cdot r^f-1$, we can obtain

$$\begin{aligned}&(i+(r-t)r^f-(r-t)) \bmod r^{f+1} \in\\ &\{r^{f+1}-(r-t),\ldots,r^{f+1}-1,0,1,\ldots,r^f-(r-t)-1\}.\end{aligned}$$

Recall that $(p-1)\tau/r$ bits $s_{i,0}$ with i satisfying (23) have already been downloaded. As

$$\begin{aligned}&\{r^{f+1}-(r-t),\ldots,r^{f+1}-1,0,1,\ldots,r^f-(r-t)-1\}\backslash\\ &\{0,1,\ldots,r^f-1\} = \{r^{f+1}-(r-t),\ldots,r^{f+1}-1\},\end{aligned}$$

we only need to download $\frac{(p-1)\tau}{r^f}$ bits $s_{i,0}$ from column 0 with

$$i \bmod r^{f+1} \in \{r^{f+1}-(r-t),\ldots,r^{f+1}-1\}.$$

By repeating the above discussion for column j, we need to download $r^j\frac{(p-1)\tau}{r^f}$ bits from column j in steps 5 and 6, where $j = 0,1,\ldots,f-1$. Consider column j with $j = f+1, f+2, \ldots, k-1$. When $i \bmod r^{f+1}$ runs from $t\cdot r^f$ to $(t+1)\cdot r^f-1$, we have

$$(i+(r-t)r^f-(r-t)r^\ell) \bmod r^{f+1} = 0,1,\ldots,r^f-1,$$

and we do not need to download bit from column j any more in steps 5 and 6.

Therefore, the bits downloaded in repairing column f is

$$\frac{(p-1)(k+r-1)\tau}{r}+\sum_{\ell=0}^{f-1}r^\ell\frac{(p-1)\tau}{r^f}$$

as given in (10). □

Appendix B
Proof of Theorem 4

Proof. First we consider that $f \in \{0,1,\ldots,\lceil\frac{k}{2}\rceil-1\}$. Similar to the proof of Theorem 3, we can show that Algorithm 2 can recover all information bits in column f.

Next, we calculate the repair bandwidth of column f. When $f \in \{0,1,\ldots,\lceil\frac{k}{2}\rceil-1\}$, we recover the bits $s_{i,f}$ by (14) by Algorithm 2. If

$$i \bmod (\frac{r}{2})^{f+1} \in \{t\cdot(\frac{r}{2})^f, 1+t\cdot(\frac{r}{2})^f,\ldots,(t+1)\cdot(\frac{r}{2})^f-1\} \tag{24}$$

for $t=1,2,\ldots,r-1$, then we need to download one parity bit $s_{i+(\frac{r}{2}-t)(\frac{r}{2})^f,k+\frac{r}{2}-t}$, and $k-1$ information bits $s_{i+(\frac{r}{2}-t)(\frac{r}{2})^f,k-1}$ and $s_{i+(\frac{r}{2}-t)(\frac{r}{2})^f-(\frac{r}{2}-t)(\frac{r}{2})^\ell,\ell}$ with $\ell=0,1,\ldots,f-1,f+1,\ldots,k-2$. If

$$i \bmod (\frac{r}{2})^{f+1} \in \{0,1,\ldots,(\frac{r}{2})^f-1\}, \tag{25}$$

we need to download one parity bit $s_{i,k}$ and $k-1$ information bits $s_{i,0},s_{i,1},\ldots,s_{i,f-1},s_{i,f+1},\ldots,s_{i,k-1}$. Recall that $(p-1)\tau$ is a multiple of $(\frac{r}{2})^{\frac{r}{2}}$ and also a multiple of $(\frac{r}{2})^{f+1}$ for $f=0,1,\ldots,\frac{r}{2}-1$, and thus $i \bmod (\frac{r}{2})^{f+1}$ is uniformed distributed over $\{0,1,\ldots,(\frac{r}{2})^{f+1}-1\}$.

We can first download $k(p-1)\tau/\frac{r}{2}$ bits $s_{i,\ell}$ with $\ell=0,1,\ldots,f-1,f+1,\ldots,k-1$ and i satisfying (25) by steps 2 to 4 in Algorithm 2. Then, we only need to download the bits in steps 5 to 7 which have not downloaded in steps 2 to 4. We first consider the bits $s_{i+(\frac{r}{2}-t)(\frac{r}{2})^f-(\frac{r}{2}-t),0}$ downloaded from column 0. If $i \bmod (\frac{r}{2})^{f+1}=t\cdot(\frac{r}{2})^f$, then there exists an integer m such that $i=m\cdot(\frac{r}{2})^{f+1}+t\cdot(\frac{r}{2})^f$ and

$$\begin{aligned}&(i+(\frac{r}{2}-t)(\frac{r}{2})^f-(\frac{r}{2}-t)) \bmod (\frac{r}{2})^{f+1}\\ =&((m\cdot\frac{r}{2}^{f+1}+t\cdot\frac{r}{2}^f+(\frac{r}{2}-t)\frac{r}{2}^f-(\frac{r}{2}-t))) \bmod \frac{r}{2}^{f+1}\\ =&(\frac{r}{2})^{f+1}-(\frac{r}{2}-t) \bmod (\frac{r}{2})^{f+1}.\end{aligned}$$

By repeating the above procedure for $i \bmod (\frac{r}{2})^{f+1}=t\cdot(\frac{r}{2})^f,1+t\cdot(\frac{r}{2})^f,\ldots,(t+1)\cdot(\frac{r}{2})^f-1$, we can obtain

$$\begin{aligned}&(i+(\frac{r}{2}-t)(\frac{r}{2})^f-(\frac{r}{2}-t)) \bmod (\frac{r}{2})^{f+1}\in\\ &\{\frac{r}{2}^{f+1}-(\frac{r}{2}-t),\ldots,\frac{r}{2}^{f+1}-1,0,1,\ldots,\frac{r}{2}^f-(\frac{r}{2}-t)-1\}.\end{aligned}$$

Recall that $(p-1)\tau/\frac{r}{2}$ bits $s_{i,0}$ with i satisfying (25) have already been downloaded. As

$$\begin{aligned}&\{\frac{r}{2}^{f+1}-(\frac{r}{2}-t),\ldots,\frac{r}{2}^{f+1}-1,0,1,\ldots,\frac{r}{2}^f-(\frac{r}{2}-t)-1\}\backslash\\ &\{0,1,\ldots,\frac{r}{2}^f-1\}=\{\frac{r}{2}^{f+1}-(\frac{r}{2}-t),\ldots,\frac{r}{2}^{f+1}-1\},\end{aligned}$$

we only need to download $\frac{(p-1)\tau}{(\frac{r}{2})^f}$ bits $s_{i,0}$ from column 0 with

$$i \bmod (\frac{r}{2})^{f+1}\in\{(\frac{r}{2})^{f+1}-(\frac{r}{2}-t),\ldots,(\frac{r}{2})^{f+1}-1\}.$$

By repeating the above discussion for column j, we need to download $(\frac{r}{2})^j\frac{(p-1)\tau}{(\frac{r}{2})^f}$ bits from column j in steps 5 to 7, where $j=0,1,\ldots,f-1$. Consider column j with $j=f+1,f+2,\ldots,k-1$. When $i \bmod (\frac{r}{2})^{f+1}$ runs from $t\cdot(\frac{r}{2})^f$ to $(t+1)\cdot(\frac{r}{2})^f-1$, we have

$$(i+(\frac{r}{2}-t)\frac{r}{2}^f-(\frac{r}{2}-t)\frac{r}{2}^\ell) \bmod \frac{r}{2}^{f+1}=0,1,\ldots,\frac{r}{2}^f-1,$$

and we do not need to download bit from column j any more in steps 5 to 7.

Therefore, the bits downloaded in repairing column f is

$$\frac{(p-1)(k+\frac{r}{2}-1)\tau}{\frac{r}{2}}+\sum_{\ell=0}^{f-1}(\frac{r}{2})^\ell\frac{(p-1)\tau}{(\frac{r}{2})^f}$$

as given in (17).

For $f\in\{\lceil\frac{k}{2}\rceil,\lceil\frac{k}{2}\rceil+1,\ldots,k-1\}$, the repair bandwidth of column f can be obtained similarly. □

Appendix C
Proof of Theorem 6

Proof. Recall that, in Algorithm 1, we need to download the parity bits $s_{i,k}$ from column k for $i \bmod r^{f+1}\in\{0,1,\ldots,r^f-1\}$ and $s_{i+(r-t)r^f,k+r-t}$ from column $k+r-t$ for $t=1,2,\ldots,r-1$ and $i \bmod r^{f+1}\in\{tr^f,1+tr^f,\ldots,(t+1)r^f-1\}$. When $i \bmod r^{f+1}$ runs from tr^f to $(t-1)r^f-1$, we have

$$(i+(r-t)r^f) \bmod r^{f+1}=\{0,1,\ldots,r^f-1\},$$

where we need to download $s_{i,k+j}$ from column $k+j$ for $i \bmod r^{f+1}\in\{0,1,\ldots,r^f-1\}$ and $j=0,1,\ldots,r-1$, in repairing column f. Let the set of the indices of information bits downloaded from column j with $j=0,1,\ldots,f-1,f+1,\ldots,k-1$ in repairing column f of the array codes in Section II-B is denoted by $S_{f,j}$. In the following, we show that we can repair column f of the transformed codes by downloading all information bits $s^\ell_{i,j}$ from column j for $\ell=0,1,\ldots,r-1$, $j=0,1,\ldots,f-1,f+1,\ldots,k-1$. $i\in S_{f,j}$, all parity bits $c^\ell_{i,j}$ from column j for $\ell=0,1,\ldots,r-1$, $j=k,k+1,\ldots,k+r-1$, and $i \bmod r^{f+1}\in\{0,1,\ldots,r^f-1\}$.

Consider the downloaded parity bits $c^\ell_{i,j}$ for $\ell=0,1,\ldots,r-1$, $j=k,k+1,\ldots,k+r-1$ and $i=0,\tau,2\tau,\ldots,(p-1)\tau$. According to (19), we have $c^\ell_{i,j}=s^\ell_{i,j}$ when $\ell=j-k$, $c^\ell_{i,j}=s^\ell_{i,j}+s^{j-k}_{i,k+\ell}$ when $\ell\le j-k$ and $c^\ell_{i,j}=s^\ell_{i-\tau,j}+s^{j-k}_{i,k+\ell}$ when $\ell\ge j-k$. For $\ell\ne j-k$, we have downloaded the following bits

$$\begin{array}{ll} s^\ell_{0,j}+s^{j-k}_{0,k+\ell}, & s^\ell_{(p-1)\tau,j}+s^{j-k}_{0,k+\ell},\\ s^\ell_{\tau,j}+s^{j-k}_{\tau,k+\ell}, & s^\ell_{0,j}+s^{j-k}_{\tau,k+\ell},\\ \vdots & \vdots\\ s^\ell_{(p-2)\tau,j}+s^{j-k}_{(p-2)\tau,k+\ell}, & s^\ell_{(p-3)\tau,j}+s^{j-k}_{(p-2)\tau,k+\ell}, \end{array}$$

and then we can compute

$$\begin{aligned} s^\ell_{0,j}+s^\ell_{(p-1)\tau,j}=&(s^\ell_{0,j}+s^{j-k}_{0,k+\ell})+(s^\ell_{(p-1)\tau,j}+s^{j-k}_{0,k+\ell}),\\ s^\ell_{\tau,j}+s^\ell_{0,j}=&(s^\ell_{\tau,j}+s^{j-k}_{\tau,k+\ell})+(s^\ell_{0,j}+s^{j-k}_{\tau,k+\ell}),\\ &\vdots\\ s^\ell_{(p-2)\tau,j}+s^\ell_{(p-3)\tau,j}=&(s^\ell_{(p-2)\tau,j}+s^{j-k}_{(p-2)\tau,k+\ell})+\\ &(s^\ell_{(p-3)\tau,j}+s^{j-k}_{(p-2)\tau,k+\ell}). \end{aligned}$$

Recall that the extra bit $s^\ell_{(p-1)\tau,j}$ is computed as $s^\ell_{(p-1)\tau,j} = s^\ell_{0,j}+s^\ell_{\tau,j}+\ldots+s^\ell_{(p-2)\tau,j}$. We can obtain $s^\ell_{(p-2)\tau,j}+s^\ell_{(p-3)\tau,j}$ and $s^\ell_{(p-2)\tau,j}$ by

$$\begin{aligned}&s^\ell_{(p-2)\tau,j}+s^\ell_{(p-3)\tau,j} = (s^\ell_{0,j}+s^\ell_{(p-1)\tau,j}) + (s^\ell_{\tau,j}+s^\ell_{0,j}) + \ldots\\&+(s^\ell_{(p-2)\tau,j}+s^\ell_{(p-3)\tau,j}),\\&s^\ell_{(p-2)\tau,j} = (s^\ell_{0,j}+s^\ell_{(p-1)\tau,j}) + (s^\ell_{2\tau,j}+s^\ell_{\tau,j})+\\&(s^\ell_{4\tau,j}+s^\ell_{3\tau,j}) + \ldots + (s^\ell_{(p-3)\tau,j}+s^\ell_{(p-4)\tau,j}).\end{aligned}$$

We can compute the other bits $s^\ell_{i\tau,j}$ for $i = p-3, p-4, \ldots, 0$ by

$$s^\ell_{i\tau,j} = s^\ell_{(i+1)\tau,j} + (s^\ell_{(i+1)\tau,j}+s^\ell_{i\tau,j}).$$

Once the bits $s^\ell_{i\tau,j}$ for $i = 0, 1, \ldots, p-2$ are known, we can compute the bits $s^{j-k}_{i\tau,k+\ell}$ by

$$s^{j-k}_{i\tau,k+\ell} = s^\ell_{i\tau,j} + (s^\ell_{i\tau,j}+s^{j-k}_{i\tau,k+\ell}).$$

As τ is a multiple of r^{f+1}, we can compute the bits $s^\ell_{i\tau,j}$ and $s^{j-k}_{i\tau,k+\ell}$ for all $i \bmod r^{f+1} \in \{0,1,\ldots,r^f-1\}$ by repeating the above procedure.

Recall that we can recover the polynomial $s^\ell_f(x)$ in column f from the information bits $s^\ell_{i,j}$ for $j = 0, 1, \ldots, k-1$, $i \in S_{f,j}$, the bits $s^\ell_{i,j}$ for $j = k, k+1, \ldots, k+r-1$, and $i \bmod r^{f+1} \in \{0,1,\ldots,r^f-1\}$. As we have computed all bits $s^\ell_{i,j}$ for $\ell = 0, 1, \ldots, r-1$, $j = k, k+1, \ldots, k+r-1$, and $i \bmod r^{f+1} \in \{0,1,\ldots,r^f-1\}$ by the above procedure, together with the downloaded information bits $s^\ell_{i,j}$ from column j for $\ell = 0, 1, \ldots, r-1$, $j = 0, 1, \ldots, f-1, f+1, \ldots, k-1$, and $i \in S_{f,j}$, we can recover the r polynomials in column f by Algorithm 1. By Theorem 3, the repair bandwidth of column f of the first constructed array codes is

$$\frac{(p-1)\tau(k+r-1)}{r} + \frac{(p-1)\tau(r^f-1)}{(r-1)r^f}.$$

Hence, the repair bandwidth of column f of the first transformed codes is

$$r(\frac{(p-1)\tau(k+r-1)}{r} + \frac{(p-1)\tau(r^f-1)}{(r-1)r^f}).$$

Recall that the number of all information bits of the first constructed array codes and the first transformed codes is $k(p-1)\tau$ and $rk(p-1)\tau$, respectively. The normalized repair bandwidth of column f of the first constructed array codes is the same as that of the first transformed codes. □

References

[1] H. Hou, Y. S. Han, and P. P. C. Lee, "Binary MDS Array Codes with Asymptotically Optimal Repair for All Columns," in *The 28th International Conference on Computer Communications and Networks (ICCCN)*, 2019.

[2] L. Xu and J. Bruck, "X-code: MDS Array Codes with Optimal Encoding," *IEEE Trans. on Information Theory*, vol. 45, no. 1, pp. 272–276, 1999.

[3] 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*, 2004, pp. 1–14.

[4] M. Blaum, J. Brady, J. Bruck, and J. 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] 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.

[6] M. Blaum, "A Family of MDS Array Codes with Minimal Number of Encoding Operations," in *IEEE Int. Symp. on Inf. Theory*, 2006, pp. 2784–2788.

[7] M. Blaum, J. Bruck, and A. Vardy, "MDS Array Codes with Independent Parity Symbols," *IEEE Trans. on Information Theory*, vol. 42, no. 2, pp. 529–542, 1996.

[8] D. Ford, F. Labelle, F. I. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in Globally Distributed Storage Systems," in *Proc. of USENIX OSDI*, 2010.

[9] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," *IEEE Trans. on Information Theory*, vol. 56, no. 9, pp. 4539–4551, September 2010.

[10] I. Tamo, Z. Wang, and J. Bruck, "Zigzag Codes: MDS Array Codes with Optimal Rebuilding," *IEEE Trans. on Information Theory*, vol. 59, no. 3, pp. 1597–1616, 2013.

[11] M. Ye and A. Barg, "Explicit Constructions of High-Rate MDS Array Codes With Optimal Repair Bandwidth," *IEEE Trans. on Information Theory*, vol. 63, no. 4, pp. 2001–2014, 2016.

[12] J. Li, X. Tang, and C. Tian, "A Generic Transformation to Enable Optimal Repair in MDS Codes for Distributed Storage Systems," *IEEE Trans. on Information Theory*, vol. 64, no. 9, pp. 6257–6267, 2018.

[13] T. Zhou and C. Tian, "Fast Erasure Coding for Data Storage: A Comprehensive Study of the Acceleration Techniques," in *17th USENIX Conference on File and Storage Technologies (FAST 19)*, 2019, p. 317329.

[14] A. Chowdhury and A. Vardy, "New Constructions of MDS Codes with Asymptotically Optimal Repair," in *Proc. IEEE Int. Symp. Inf. Theory*, 2018, pp. 1944–1948.

[15] Y. Wang, X. Yin, and X. Wang, "MDR Codes: A New Class of RAID-6 Codes with Optimal Rebuilding and Encoding," *IEEE J. Selected Areas in Communications*, vol. 32, no. 5, pp. 1008–1018, 2013.

[16] E. E. Gad, R. Mateescu, F. Blagojevic, C. Guyot, and Z. Bandic, "Repair-Optimal MDS Array Codes over GF(2)," in *Proc. IEEE Int. Symp. Inf. Theory*, 2013, pp. 887–891.

[17] H. Hou, P. P. C. Lee, Y. S. Han, and Y. Hu, "Triple-Fault-Tolerant Binary MDS Array Codes with Asymptotically Optimal Repair," in *Proc. IEEE Int. Symp. Inf. Theory*, Aachen, June 2017.

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

[19] H. Hou and Y. S. Han, "A Class of Binary MDS Array Codes with Asymptotically Weak-Optimal Repair," *SCIENCE CHINA Information Sciences*, vol. 61, no. 10, pp. 1–12, 2018.

[20] Y. Wang, X. Yin, and X. Wang, "Two New Classes of Two-Parity MDS Array Codes With Optimal Repair," *IEEE Communications Letters*, vol. 20, no. 7, pp. 1293–1296, 2016.

[21] L. Pamies-Juarez, F. Blagojevic, R. Mateescu, C. Gyuot, E. E. Gad, and Z. Bandic, "Opening the Chrysalis: on the Real Repair Performance of MSR Codes," in *14th USENIX Conference on File and Storage Technologies (FAST 16)*, 2016, pp. 81–94.

[22] H. Hou, K. W. Shum, M. Chen, and H. Li, "BASIC Regenerating Code: Binary Addition and Shift for Exact Repair," in *Proc. IEEE Int. Symp. Inf. Theory*, Istanbul, July 2013, pp. 1621–1625.

[23] K. W. Shum, H. Hou, M. Chen, H. Xu, and H. Li, "Regenerating Codes over a Binary Cyclic Code," in *Proc. IEEE Int. Symp. Inf. Theory*, Honolulu, July 2014, pp. 1046–1050.

[24] H. Hou, K. W. Shum, M. Chen, and H. Li, "BASIC Codes: Low-Complexity Regenerating Codes for Distributed Storage Systems," *IEEE Trans. on Information Theory*, vol. 62, no. 6, pp. 3053–3069, 2016.

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

[26] H. Hou and P. P. C. Lee, "Binary MDS Array Codes with Optimal Repair," *IEEE Trans. on Information Theory*, vol. 66, no. 3, pp. 1405–1422, 2020.

[27] J. Li and X. Tang, "A Note on the Transformation to Enable Optimal Repair in MDS Codes for Distributed Storage Systems," *arXiv preprint: arXiv:1901.06067*, 2019.

[28] H. Hou, P. P. C. Lee, and Y. S. Han, "Multi-Layer Transformed MDS Codes with Optimal Repair Access and Low Sub-Packetization," *arXiv preprint arXiv:1907.08938*, 2019.

[29] J. Li, X. Tang, and U. Parampalli, "A Framework of Constructions of Minimal Storage Regenerating Codes With the Optimal Access/Update Property," *IEEE Trans. on Information Theory*, vol. 61, no. 4, pp. 1920–1932, 2015.

[30] M. Ye and A. Barg, "Explicit Constructions of Optimal-Access MDS Codes with Nearly Optimal Sub-Packetization," *IEEE Trans. on Information Theory*, pp. 6307–6317, 2017.

[31] S. Goparaju, A. Fazeli, and A. Vardy, "Minimum Storage Regenerating Codes for all Parameters," *IEEE Trans. on Information Theory*, vol. 63, no. 10, pp. 6318–6328, 2017.

[32] K. Rashmi, N. B. Shah, and K. Ramchandran, "A Piggybacking Design Framework for Read-and Download-Efficient Distributed Storage Codes," *IEEE Trans. on Information Theory*, vol. 63, no. 9, pp. 5802–5820, 2017.

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, Peking University. He is now an Associate Professor with the School of Electrical Engineering & Intelligentization, Dongguan University of Technology. His research interests include erasure coding and coding for distributed storage systems.

Yunghsiang S. Han 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. Now he is with School of Electrical Engineering & Intelligentization at Dongguan University of Technology, 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 an Associate 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