A New Construction of EVENODD Codes with Lower Computational Complexity

Hanxu Hou[†*] and Patrick P. C. Lee[§]
[†] School of Electrical Engineering & Intelligentization, Dongguan University of Technology
[§] Department of Computer Science and Engineering, The Chinese University of Hong Kong

***Abstract*— EVENODD codes are binary array codes for correcting double disk failures in RAID-6 with asymptotically optimal encoding and decoding complexities. However, the update complexity of EVENODD is sub-optimal. We propose a new construction of binary MDS array codes, namely EVENODD+, such that the encoding, decoding, and update complexities of EVENODD+ are less than those of EVENODD in general. Moreover, EVENODD+ achieves asymptotically optimal update complexity.**

***Index Terms*—EVENODD codes, update complexity.**

I. Introduction

ARRAY codes have been widely employed in storage systems, such as Redundant Arrays of Inexpensive Disks (RAID) [1], for data reliability. In particular, RAID-6 systems dedicate two disks for storing parity-check bits and are tolerable against any two disk failures.

Consider a binary array code of size $r \times n$, in which each entry in the array stores one bit. Among the n columns, k of them store the information bits and are called the *information columns*, while the remaining $n-k$ columns store the parity bits and are called the *parity columns*. Practical array codes are maximum distance separable (MDS), i.e., any k columns suffice to recover the information bits; that is, the system can tolerate any $n-k$ column failures (note that the MDS property here is measured by the column weight instead of the Hamming weight). The number of rows r depends on the code construction. In addition to the MDS property, some important performance metrics need to be considered, including the *encoding complexity* (i.e., the number of XORs needed to construct the parity bits), the *decoding complexity* (i.e., the number of XORs needed to recover the erased columns from surviving ones), and the *update complexity* (i.e., the average number of parity bits affected by a change of a single information bit). In particular, the update complexity affects the performance of small writes and is a crucial metric to storage applications (e.g., databases) with update-intensive workloads. It is desirable to design array codes whose update complexity is as small as possible.

There are many binary MDS array codes in the literature. EVENODD [2] and RDP [3] are two important codes correcting double disk failures. Other binary MDS array codes include X-code [4], Liberation code [5], H-code [6], C-code [7], HV code [8], and Short code [9]. EVENODD is well explored in the literature and has a well-designed algebraic structure that can be extended to have more parity columns [10]. Thus, in this work, we focus on extending EVENODD with improved performance.

We first provide an overview of EVENODD. EVENODD codes are $(p-1)\times(k+2)$ array codes, where $p \geq k$ is prime, the first k columns are information columns, and the last two columns are parity columns[1]. For $i = 0, 1, \ldots, p-2$, let $b_{i,j}$ be the bits stored in column j, where $j = 0, 1, \ldots, k+1$. The parity bits $b_{i,k}$ in column k are computed by

$$b_{i,k} = \sum_{j=0}^{k-1} b_{i,j},$$

and the parity bits $b_{i,k+1}$ in column $k+1$ are computed by

$$b_{i,k+1} = b_{p-1,k+1} + \sum_{j=0}^{k-1} b_{i-j,j},$$

where $b_{p-1,j} = 0$ for $j = 0, 1, \ldots, k-1$, and $b_{p-1,k+1} = \sum_{j=1}^{k-1} b_{p-1-j,j}$. Note that the subscripts above are taken modulo p. It is shown that the encoding complexity of EVENODD is $(p-1)(2k-1)-1$, which is optimal [2]. However, the update complexity of EVENODD is $3-\frac{p+k-2}{k(p-1)}$, which is sub-optimal (note that the optimal update complexity of binary MDS array codes with two parity columns is $2+\frac{k-1}{k(p-1)}$ [11]).

We present EVENODD+, an extended construction of EVENODD. EVENODD+ belongs to MDS codes and provides the same double-fault tolerance as EVENODD. It can also be reduced to EVENODD as a special case. We show that EVENODD+ has lower encoding/decoding/update complexity than that of EVENODD, and the update complexity of EVENODD+ is asymptotically optimal. The design rationale of EVENODD+ is as follows. In the original EVENODD codes, $b_{p-1,k+1}$ is added to each parity bit in column $k+1$, thereby making the update complexity sub-optimal. The main idea of EVENODD+ is to preserve the MDS property but avoid adding a specific bit to each parity bit in column $k+1$. This enables EVENODD+ to reduce the update complexity.

This work was partially supported by the National Natural Science Foundation of China (No. 61701115) and Research Grants Council of Hong Kong (GRF 14216316).
* Corresponding author.

[1]Note that the definition of EVENODD codes here refers to the *shortened* codes of the original EVENODD codes in [2], which we obtain by deleting $p-k$ information columns from the original $(p-1)\times(p+2)$ EVENODD codes. The original EVENODD codes can be viewed as a special case of the shortened EVENODD codes with $k = p$.

TABLE I: EVENODD+$(9,3)$ (note that $b_{8,4} = b_{7,1} + b_{6,2}$).

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

II. New Construction: EVENODD+

We now present EVENODD+ (with two parity columns). Given an odd integer $m \geq k$, we define an $(m-1)\times(k+2)$ array code as follows. For $j = 0, 1, \ldots, k-1$, column j is called an *information column* that stores the information bits $b_{0,j}, b_{1,j}, \ldots, b_{m-2,j}$, and for $j = k, k+1$, column j is called a *parity column* that stores the parity bits $b_{0,j}, b_{1,j}, \ldots, b_{m-2,j}$. The subscripts are taken modulo m throughout the paper unless otherwise specified.

Given the $(m-1)\times k$ information array $b_{i,j}$ for $i = 0, 1, \ldots, m-2$ and $j = 0, 1, \ldots, k-1$, we define an imaginary row $b_{m-1,j} = 0$ for $j = 0, 1, \ldots, k-1$. The bits in column k are computed by

$$b_{i,k} = \sum_{j=0}^{k-1} b_{i,j} \text{ for } 0 \leq i \leq m-2, \tag{1}$$

and the bits in column $k+1$ are computed by

$$b_{i,k+1} = \begin{cases} b_{m-1,k+1} + \sum\limits_{j=0}^{k-1} b_{i-j,j} & \text{for } 0 \leq i \leq 2\lfloor \frac{k}{2} \rfloor - 1, \\ \sum\limits_{j=0}^{k-1} b_{i-j,j} & \text{for } 2\lfloor \frac{k}{2} \rfloor \leq i \leq m-2, \end{cases} \tag{2}$$

where $b_{m-1,k+1} = \sum_{j=1}^{k-1} b_{m-1-j,j}$. We denote the array defined in the equations above as EVENODD+(m,k). The main differences between EVENODD+(m,k) and the original EVENODD codes are two-fold. First, the number of bits in each column is more flexible in EVENODD+(m,k), i.e., m is an odd integer that satisfies the condition in Theorem 1 (see Section III), while p should be a prime number in EVENODD. Second, the parity bits in column $k+1$ are different in both codes. In EVENODD+(m,k), we only add the bit $b_{m-1,k+1}$ to the first $2\lfloor \frac{k}{2} \rfloor$ parity bits in column $k+1$, while in EVENODD, the bit $b_{p-1,k+1}$ is added to all the parity bits in column $k+1$. The above two differences enable EVENODD+(m,k) to achieve asymptotically optimal update complexity and lower encoding/decoding complexity than EVENODD. When $m = k$ is a prime number, EVENODD+(m,m) is reduced to EVENODD. Table I illustrates an example of EVENODD+$(9,3)$, in which the bit $b_{8,4}$ is added to the first two parity bits in column 4.

III. The MDS Property

In this section, we show that the MDS property of EVENODD+(m,k) holds in Theorem 1, which also gives the decoding method for any two column failures.

Theorem 1. *EVENODD+(m,k) is MDS if and only if m is an odd integer such that all divisors of m except 1 are larger than $k-1$.*

Proof. We first show the "if" part: if m is an odd integer such that all divisors of m except 1 are larger than $k-1$, we want to show that EVENODD+(m,k) are MDS codes. It is equivalent to showing that the $k(m-1)$ information bits can be reconstructed after any two column failures. The reconstruction can be divided into three cases: (i) from all k information columns, (ii) from any $k-1$ information columns and one parity column, and (iii) from any $k-2$ information columns and two parity columns.

For case (i), we can obtain the $k(m-1)$ information bits directly from k information columns. Consider case (ii). First, suppose that columns f and $k+1$ are erased, where $0 \leq f \leq k-1$. We can recover the information bits in column f by

$$b_{i,k} + (b_{i,0} + b_{i,1} + \cdots + b_{i,f-1} + b_{i,f+1} + \cdots + b_{i,k-1}) = b_{i,f}$$

according to (1), for $i = 0, 1, \ldots, m-2$. Second, suppose that columns f and k are erased, where $0 \leq f \leq k-1$. The bits $b_{i-f,f}$ for $i = 2\lfloor \frac{k}{2} \rfloor, 2\lfloor \frac{k}{2} \rfloor + 1, \ldots, m-2$ can be recovered by

$$\begin{aligned} \left(\sum_{j=0, j\neq f}^{k-1} b_{i-j,j} \right) + b_{i,k+1} &= \sum_{j=0, j\neq f}^{k-1} b_{i-j,j} + \sum_{j=0}^{k-1} b_{i-j,j} \\ &= b_{i-f,f}. \end{aligned}$$

The first equation above comes from (2). If $f = 0$, the other information bits $b_{i,0}$ can be repaired by

$$\begin{aligned} & b_{i,k+1} + \sum_{j=1}^{k-1} b_{i-j,j} + \sum_{j=1}^{k-1} b_{m-1-j,j} \\ &= \sum_{j=0}^{k-1} b_{i-j,j} + \sum_{j=1}^{k-1} b_{m-1-j,j} + \sum_{j=1}^{k-1} b_{i-j,j} + \sum_{j=1}^{k-1} b_{m-1-j,j} \\ &= b_{i,0}, \end{aligned}$$

where the first equation above comes from (2), for $i = 0, 1, \ldots, 2\lfloor \frac{k}{2} \rfloor - 1$. Otherwise, if $f \geq 1$, we have $0 \leq f-1 \leq k-2 \leq 2\lfloor \frac{k}{2} \rfloor - 1$ and we can recover the bit $b_{m-f-1,f}$ by

$$\begin{aligned} & b_{f-1,k+1} + \sum_{j=0, j\neq f}^{k-1} b_{f-1-j,j} + \sum_{j=1, j\neq f}^{k-1} b_{m-1-j,j} \\ &= \sum_{j=0}^{k-1} b_{f-1-j,j} + \sum_{j=1}^{k-1} b_{m-1-j,j} + \\ & \quad \sum_{j=0, j\neq f}^{k-1} b_{f-1-j,j} + \sum_{j=1, j\neq f}^{k-1} b_{m-1-j,j} \\ &= b_{m-1,f} + b_{m-f-1,f} \\ &= b_{m-f-1,f}, \end{aligned}$$

where the first equation above comes from (2) and the last equation above comes from $b_{m-1,f} = 0$. Now, $b_{m-f-1,f}$ is

known. We can repair $b_{i-f,f}$ by

$$\begin{aligned}&\left(\sum_{j=0,j\neq f}^{k-1} b_{i-j,j}\right)+\left(\sum_{j=1}^{k-1} b_{m-1-j,j}\right)+b_{i,k+1}\\ =&\sum_{j=0,j\neq f}^{k-1} b_{i-j,j}+\sum_{j=1}^{k-1} b_{m-1-j,j}+\sum_{j=0}^{k-1} b_{i-j,j}+\sum_{j=1}^{k-1} b_{m-1-j,j}\\ =&\, b_{i-f,f},\end{aligned}$$

where $i = 0, 1, \ldots, 2\lfloor \frac{k}{2} \rfloor - 1$. Therefore, we can recover column f from case (ii).

Finally, consider case (iii). Without loss of generality, we assume that the indices of two failure information columns are f and g, where $0 \leq f < g \leq k-1$. We want to decode the information bits in columns f and g. We can first compute $b_{m-1,k+1}$ by summing all the parity bits in columns k and $k+1$, i.e.,

$$\begin{aligned}&\sum_{i=0}^{m-2} b_{i,k}+\sum_{i=0}^{m-2} b_{i,k+1}\\ =&\sum_{i=0}^{m-2}\sum_{j=0}^{k-1} b_{i,j}+\sum_{i=0}^{m-2}\sum_{j=0}^{k-1} b_{i-j,j}+\\ &\underbrace{b_{m-1,k+1}+\cdots+b_{m-1,k+1}}_{k-1 \text{ terms if } k \text{ is odd}, k \text{ terms if } k \text{ is even}} \qquad (3)\\ =&\sum_{j=0}^{k-1}\sum_{i=0}^{m-1} b_{i,j}+\sum_{j=0}^{k-1}\sum_{i=0}^{m-1} b_{i-j,j}+\sum_{j=0}^{k-1} b_{m-1-j,j} \qquad (4)\\ =&\, b_{m-1,k+1}. \qquad (5)\end{aligned}$$

In above equations, (3) comes from (1) and (2); (4) comes from $b_{m-1,j} = 0$ for $j = 0, 1, \ldots, k-1$; (5) comes from

$$\{-j, 1-j, \cdots, m-1-j\} = \{0, 1, \cdots, m-1\} \bmod m.$$

We subtract $b_{m-1,k+1}$ from $b_{i,k+1}$ for $i = 0, 1, \ldots, 2\lfloor \frac{k}{2} \rfloor - 1$ and let $b'_{i,k+1} = b_{i,k+1}$ for $i = 2\lfloor \frac{k}{2} \rfloor, 2\lfloor \frac{k}{2} \rfloor + 1, \ldots, m-1$. Then, we obtain

$$b'_{i,k+1} = \sum_{j=0}^{k-1} b_{i-j,j}$$

by (2) for $i = 0, 1, \ldots, m-1$. Next, we subtract $(k-2)(m-1)$ information bits in $k-2$ surviving information columns from bits $b_{i,k}$ and $b'_{i,k+1}$ for $i = 0, 1, \ldots, m-1$ and obtain the following $2m$ bits

$$b_{i,f} + b_{i,g} \text{ and } b_{g-f+i,f} + b_{i,g} \text{ for } i = 0, 1, \ldots, m-1. \qquad (6)$$

Recall that $b_{m-1,f} = b_{m-1,g} = 0$. We can directly obtain

$$\begin{cases} b_{g-f+m-1,f} = b_{g-f+m-1,f} + b_{m-1,g} \\ b_{g-f+m-1,g} = b_{g-f+m-1,f} + (b_{g-f+m-1,f} + b_{g-f+m-1,g}). \end{cases}$$

We compute $b_{2(g-f)+m-1,f}$ and $b_{2(g-f)+m-1,g}$ by

$$\begin{cases} & b_{2(g-f)+m-1,f} \\ = & b_{g-f+m-1,g} + (b_{2(g-f)+m-1,f} + b_{g-f+m-1,g}), \\ & b_{2(g-f)+m-1,g} \\ = & b_{2(g-f)+m-1,f} + (b_{2(g-f)+m-1,f} + b_{2(g-f)+m-1,g}). \end{cases}$$

The information bits $b_{i(g-f)+m-1,f}$ and $b_{i(g-f)+m-1,g}$ can be decoded iteratively for $i = 1, 2, \ldots, m-1$ if

$$\{m-1+i(g-f) \bmod m \mid 1 \leq i \leq m-1\} = \{0, \ldots, m-2\}. \qquad (7)$$

As $1 \leq g-f \leq k-1$ and all divisors of m except 1 are larger than $k-1$, we have that $\gcd(g-f, m) = 1$. First, we prove that if $i \neq j$, then $(m-1+i(g-f)) \bmod m \neq (m-1+j(g-f)) \bmod m$. If $(m-1+i(g-f)) \bmod m = (m-1+j(g-f)) \bmod m$ for $1 \leq i < j \leq m-1$, then there exists an integer ℓ such that

$$m-1+j(g-f) = \ell m + m - 1 + i(g-f).$$

The equation above can be further reduced to

$$(j-i)(g-f) = \ell m.$$

Since $\gcd(g-f, m) = 1$, we have $(j-i) | m$. However, this is impossible due to the fact that $1 \leq i < j \leq m-2$. Similarly, we can prove that, for $1 \leq i \leq m-1$,

$$m-1+i(g-f) \bmod m \neq m-1.$$

Hence, (7) holds. Therefore, we can decode all the information bits in columns f and g, if m is an odd number such that all divisors of m except 1 are larger than $k-1$.

We now show the "only-if" part. If EVENODD+(m, k) is MDS, then we can recover all the information bits from any k columns. Consider the case that we want to recover information columns f and g from the other $k-2$ information columns and two parity columns. By the same procedures of case (iii), we can show that all the information bits in columns f and g can be recovered if (7) holds. As (7) holds only when $\gcd(g-f, m) = 1$, all the divisors of m except 1 should be larger than $k-1$. This completes the proof. □

We show via the example of $m = 9, k = 3$ in Table I the reconstruction method of case (iii). Suppose that information columns f and g are erased, where $0 \leq f < g \leq 2$. We can compute $b_{8,4}$ by summing all parity bits in columns 3 and 4:

$$\sum_{i=0}^{7} b_{i,3} + \sum_{i=0}^{7} b_{i,4} = b_{7,1} + b_{6,2} = b_{8,4}.$$

After subtracting $b_{8,4}$ from $b_{0,4}$ and $b_{1,4}$, we obtain 18 bits $b_{i,3} = b_{i,0} + b_{i,1} + b_{i,2}$ and $b'_{i,4} = b_{i,0} + b_{(i-1) \bmod 9,1} + b_{(i-2) \bmod 9,2}$ for $i = 0, 1, \ldots, 8$. Then, we subtract $b_{i,\ell}$ from $b_{i,3}$ and $b'_{i,4}$ for $i = 0, 1, \ldots, 8$, where $\ell = \{0, 1, 2\} \setminus \{g, f\}$, to obtain 18 bits

$$b_{i,f} + b_{i,g} \text{ and } b_{(g-f+i) \bmod 9,f} + b_{i,g} \text{ for } i = 0, 1, \ldots, 8.$$

As $b_{8,f} = b_{8,g} = 0$, we can obtain $b_{(g-f+8) \bmod 9,f} = b_{(g-f+8) \bmod 9,f} + b_{8,g}$ and $b_{(g-f+8) \bmod 9,g} = b_{(g-f+8) \bmod 9,f} + (b_{(g-f+8) \bmod 9,f} + b_{(g-f+8) \bmod 9,g})$. The other information bits can be recovered iteratively.

We note that in case (iii) of the proof in Theorem 1, the calculation of $b_{m-1,k+1}$, obtained by summing all the $2(m-1)$ parity bits in (3), is a key point. By (3), we can always obtain $b_{m-1,k+1}$ if the number of parity bits in column $k+1$ that contain $b_{m-1,k+1}$ is an even number. This is one of the reasons we add $b_{m-1,k+1}$ to the first $2\lfloor \frac{k}{2} \rfloor$ parity bits in column $k+1$.

However, the number of parity bits in column $k+1$ that contain $b_{m-1,k+1}$ should be no less than $2\lfloor\frac{k}{2}\rfloor$, as we need to ensure that any $k-1$ information columns and column $k+1$ can recover all the information bits.

IV. Complexity Analysis

The next theorem shows the encoding, decoding, and update complexities for EVENODD+(m,k); note that we focus on the decoding complexity for decoding two information erasures.

Theorem 2. *The encoding, decoding, and update complexities of EVENODD+(m,k) are $2km-2m-k$, $2km+2\lfloor\frac{k}{2}\rfloor-2k-2$, and $2+(2\lfloor\frac{k}{2}\rfloor-1)\frac{k-1}{k(m-1)}$, respectively.*

Proof. First, consider the encoding complexity. Computing the bits in column k according to (1) takes $(k-1)(m-1)$ XORs. The bits in column $k+1$ are computed by (2) that involve $(k-1)(m-1)+k-2$ XORs. Thus, the encoding complexity is $2km-2m-k$.

Next, consider the decoding complexity of two information erasures. The decoding process is given in the proof of Theorem 1. First, we can compute $b_{m-1,k+1}$ by (5) that takes $2m-3$ XORs. Then we can obtain the bits in (6) with $2(k-2)(m-1)+2\lfloor\frac{k}{2}\rfloor$ XORs. Finally, we can recover the erased $2(m-1)$ information bits with $2m-3$ XORs. Thus, the decoding complexity is $2km+2\lfloor\frac{k}{2}\rfloor-2k-2$.

Finally, consider the update complexity. If an information bit is changed, we need to update one parity bit in column k and $1+(2\lfloor\frac{k}{2}\rfloor-1)\frac{k-1}{k(m-1)}$ parity bits in column $k+1$ on average. Thus, the update complexity is $2+(2\lfloor\frac{k}{2}\rfloor-1)\frac{k-1}{k(m-1)}$. □

Define the *normalized encoding complexity* as the ratio of encoding complexity to the number of information bits and *normalized decoding complexity* as the ratio of decoding complexity to the number of information bits. By Theorem 2, the normalized encoding complexity of EVENODD+(m,k) is $2-\frac{2m-k}{k(m-1)}$. Recall that the normalized encoding complexity of EVENODD is $2-\frac{p}{k(p-1)}$. Therefore, the normalized encoding complexity of EVENODD+(m,k) is slightly less than that of EVENODD for $m=p>k$.

According to Theorem 2, the normalized decoding complexity of EVENODD+(m,k) is $2+\frac{2\lfloor\frac{k}{2}\rfloor-1}{k(m-1)}$. The normalized decoding complexity of EVENODD is $2+\frac{p-2}{k(p-1)}$. Therefore, the normalized decoding complexity of EVENODD+(m,k) is slightly less than that of EVENODD for $m=p>k$.

The update complexity of EVENODD+(m,k) is $2+(2\lfloor\frac{k}{2}\rfloor-1)\frac{k-1}{k(m-1)}$ by Theorem 2. If $m\gg k$, then the update complexity approaches the optimal value $2+\frac{k-1}{k(m-1)}$ [11] asymptotically in m. Therefore, the update complexity is asymptotically optimal when m is much larger than k. The update complexity of EVENODD is $3-\frac{p+k-2}{k(p-1)}$, which is strictly larger than that of EVENODD+(m,k) if $m=p>k$. Table II shows the update complexity of EVENODD, EVENODD+(m,k), and the code in [11] when $k=7$ and $m=p$ ranges from 7 to 53. We observe that EVENODD+(m,k) has less update complexity than EVENODD when $m>7$, and this advantage increases with m. As $m=49$ is not prime, we do not add the result for EVENODD and the code in [11] in Table II.

TABLE II: Update complexities of EVENODD+$(m,7)$, EVENODD, and the code in [11].

m	EVENODD	EVENODD+$(m,7)$	Code in [11]
7	2.7143	2.7143	2.1429
11	2.7714	2.4286	2.0857
13	2.7857	2.3571	2.0714
17	2.8035	2.2679	2.0536
19	2.8095	2.2381	2.0476
23	2.8182	2.1948	2.0390
29	2.8265	2.1531	2.0306
31	2.8229	2.1429	2.0286
37	2.8333	2.1190	2.0238
41	2.8357	2.1071	2.0214
43	2.8367	2.1020	2.0204
47	2.8385	2.0932	2.0186
49	n/a	2.0893	n/a
53	2.8407	2.0824	2.0165

Although the code in [11] has optimal update complexity, how to decode the information bits when two columns are failed is not presented. Also, the generalization of the code in [11] with more parity columns is not necessarily MDS.

V. Conclusion

In this letter, we present EVENODD+(m,k), a new construction of EVENODD such that its update complexity is asymptotically optimal and the encoding and decoding complexities are also slightly less than that of EVENODD. The main idea for reducing the encoding, decoding, and update complexities is that we only add the bit $b_{m-1,k+1}$ to the first $2\lfloor\frac{k}{2}\rfloor$ parity bits in the second parity column, while $b_{p-1,k+1}$ is added to all the parity bits in the second parity column in the original EVENODD construction. Future work includes the extension of the construction with more parity columns and the design of efficient repair algorithms for column failures.

References

[1] D. A. Patterson, P. Chen, G. Gibson, and R. H. Katz, "Introduction to Redundant Arrays of Inexpensive Disks (RAID)," in *Proc. IEEE COMPCON*, vol. 89, 1989, pp. 112–117.

[2] 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.

[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 *Proc. of USENIX Conf. on File and Storage Technologies (FAST)*, 2004, pp. 1–14.

[4] 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.

[5] J. S. Plank, "The RAID-6 liberation codes," in *Proc. of USENIX Conf. on File and Storage Technologies (FAST)*, 2008, pp. 97–110.

[6] C. Wu, S. Wan, X. He, Q. Cao, and C. Xie, "H-code: A hybrid MDS array code to optimize partial stripe writes in RAID-6," in *Parallel & Distributed Processing Symposium*, 2011, pp. 782–793.

[7] M. Li and J. Shu, "C-codes: Cyclic lowest-density MDS array codes constructed using starters for RAID 6," *IBM Corporation*, 2012.

[8] Z. Shen and J. Shu, "HV code: An all-around MDS code to improve efficiency and reliability of RAID-6 systems," in *IEEE International Conference on Dependable Systems and Networks*, 2014, pp. 550–561.

[9] Y. Fu, J. Shu, X. Luo, Z. Shen, and Q. Hu, "Short code: An efficient RAID-6 MDS code for optimizing degraded reads and partial stripe writes," *IEEE Trans. on Computers*, vol. 66, no. 1, pp. 127–137, 2016.

[10] M. Blaum, J. Brady, J. Bruck, J. Menon, and A. Vardy, "The EVENODD code and its generalization: An effcient scheme for tolerating multiple disk failures in RAID architectures," in *High Performance Mass Storage and Parallel I/O*. Wiley-IEEE Press, 2002, ch. 8, pp. 187–208.

[11] M. Blaum and R. M. Roth, "On lowest density MDS codes," *IEEE Trans. on Information Theory*, vol. 45, no. 1, pp. 46–59, 1999.