Lowest Density MDS Array Codes of Distance 3

Zhijie Huang, Hong Jiang, *Fellow, IEEE*, Ke Zhou, *Member, IEEE*, Yuhong Zhao, and Chong Wang

***Abstract*—We present a new family of maximum-distance separable (MDS) array codes of distance 3, called S(ymmetry)-Code, which has the properties of: 1) optimality in encoding, decoding and update; 2) code length of either p or $p-1$ with p being an odd prime; 3) requiring less I/O cost than most existing lowest density MDS array codes during single-erasure recovery; and 4) approaching the lower bound in I/O cost of single-erasure recovery when the code length is small.**

***Index Terms*—Array codes, maximum-distance separable (MDS) codes, lowest density, storage systems.**

I. Introduction

ARRAY codes have been studied extensively in the past thirty years due to their practical utility in communication and storage systems. The main advantage of these codes is that the encoding and decoding procedures use only simple XOR operations, thus are more efficient than the well-known Reed–Solomon codes in terms of computational complexity [1]. In array codes, symbols are column vectors and hence each codeword is a two-dimensional array. An array code of size $m \times n$ can be regarded as a one-dimensional code defined over the Abelian group $G(2^m)$. Let N be the number of its codewords, then the dimension k can be defined as $k = \log_{2^m} N$, as with the usual one-dimensional codes. Accordingly, the distance d of the code is defined over $G(2^m)$, i.e., each *column* is regarded as a *symbol*, and d refers to the column-wise distance. If the Singleton bound is achieved, i.e., $d=n-k+1$, then the array code is called a maximum-distance separable (MDS) array code.

When array codes are applied to storage systems, the n columns (symbols) of a codeword are stored in n different disks (nodes) respectively. Since every parity bit is constructed from a certain subset of the data bits of the array, whenever a data bit is changed, the parity bits to which it contributes must be updated (read-modify-write) accordingly. Therefore, in addition to the traditional *encoding* and *decoding* complexity, another important performance metric of array codes is *update complexity*—the average number of parity bits that must be updated when a data bit is altered. Update complexity directly dictates the *communication overhead* of small-write operations, hence is particularly crucial when the codes are used in storage applications that require frequent updates of data, particularly in a distributed storage system. In order to minimize the update complexity, a subclass of MDS array codes, called *lowest density* MDS array codes, have gained much attention in the past few years [2]–[6]. These codes can achieve the lower bound of update complexity, at the expense of certain limit in the code length.

In recent years, array codes used in storage systems also take the *I/O cost of single-erasure recovery* (IOCSER) as an additional performance metric. IOCSER refers to the amount of surviving information that needs to be accessed in order to rebuild exactly the lost information, and it deserves attention due to the following reasons: a) 99.75% of the actual recoveries in the production environment are caused by single erasures [7], b) IOCSER directly dictates the *communication overhead* and the reconstruction time of single-erasure recovery. Researchers have been trying to reduce the IOCSER in two ways: one is to develop optimal single-erasure recovery algorithms for existing array codes [8], [9], and the other is to design new codes aiming to minimize the IOCSER, e.g., [10]–[12]. The latter can reduce the IOCSER much more than the former, but the price is that the new codes are usually *unsatisfactory* in terms of encoding, decoding and update.

It is worth stressing that, none of the existing MDS array codes is perfect in that every code requires certain trade-offs in different performance metrics, according to its main application scenarios. In this letter, we construct a new family of MDS array codes of distance 3, called S(ymmetry)-Code. S-Code is a class of *lowest density* MDS array codes with a *regular* geometrical structure, so it enjoys the inherent optimal properties of *lowest density* MDS array codes and it is possible to work out a specially optimized single-erasure decoding algorithm. In general, it has the following attractive properties:

- Optimal encoding, decoding, and update complexity.
- Less IOCSER than existing *lowest density* MDS array codes that have similar code length limit.
- Approaching the lower bound of IOCSER when the code length is small. In particular, the S-Code of length 4 exactly achieves the theoretical lower bound of IOCSER.

II. Symmetric-Code (S-Code)

In S-Code, a codeword is a $(p-1) \times p$ array containing $(p-1) \times (p-2)$ data bits and $2(p-1)$ parity bits, where p is an odd prime. The parity bits are placed along two symmetric diagonals, and each parity bit is constructed from the data bits along the diagonal (anti-diagonal) that traverses it.

A. Encoding Procedure

Throughout this letter, we use $b_{i,j}$ to denote the ith bit in the jth column, where $0 \le i \le p-2$ and $0 \le j \le p-1$, and

Manuscript received February 8, 2015; accepted July 26, 2015. Date of publication August 4, 2015; date of current version October 8, 2015. This work was supported in part by the National Basic Research Program (973 Program) of China under Grant 2011CB302305, by the NSFC under Grant 61232004, and by the U.S. National Science Foundation under Grants CNS-1016609 and CNS-1116606. The associate editor coordinating the review of this paper and approving it for publication was H. Saeedi. *(Corresponding author: Ke Zhou.)*

Z. Huang, K. Zhou, Y. Zhao, and C. Wang are with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: jayzy_huang@hust.edu.cn; k.zhou@hust.edu.cn; yuhongzhao@hust.edu.cn; C_Wang@hust.edu.cn).

H. Jiang is with the Department of Computer Science and Engineering, University of Texas at Arlington, TX 76010 USA (e-mail: hong.jiang@uta.edu).

Digital Object Identifier 10.1109/LCOMM.2015.2464379

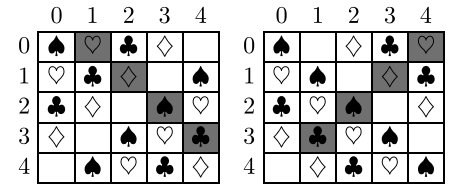

Fig. 1. Diagonal and anti-diagonal parity sets of the S-Code with $p = 5$.

let $\langle x \rangle = x \bmod p$. To facilitate the description, we also assume that there is an *imaginary* 0-row after the last row of the codeword. Then, the parity bits are constructed as follows:

$$b_{j-1,j} = \bigoplus_{t=0,t\neq j}^{p-1} b_{\langle 2j-1-t\rangle,t} \tag{1}$$

$$b_{p-1-j,j} = \bigoplus_{t=0,t\neq j}^{p-1} b_{\langle p-1-2j+t\rangle,t} \tag{2}$$

where $j = 1, 2, \cdots, p-1$. As an example, Fig. 1 shows the encoding rules of the S-Code with $p = 5$.

From the encoding rules of S-Code, it is easy to see that each parity bit is constructed from a certain collection of data bits without involving any other parity bits. Moreover, each data bit is involved in calculating exactly two parity bits. This implies that modifying one data bit results in updating exactly two parity bits. In other words, *S-Code achieves the lower bound 2 of the update complexity for any codes of distance 3*. In addition, since the last row is just an imaginary 0-row, each parity bit is constructed from only $p-2$ data bits in practice. Therefore, S-Code requires only $p-3$ XORs per parity bit in encoding, *which achieves the lower bound of the encoding complexity for any codes of distance 3 with code length p*.

Observe that the first column does not contain any parity bits, thus S-Code can be shortened by assuming that the first column is an imaginary zero column. In other words, the code length of S-Code can be either p or $p-1$.

B. MDS Property

To prove the MDS property of S-Code, we first provide two Lemmas that will be used in the proof.

Lemma 1: In the sequence of numbers $\{(p-1+k\delta) \bmod p, k = 0, 1, \cdots, p\}$, with p being prime and $0 < \delta < p$, the endpoints are both equal to $p-1$, and all numbers $0, 1, \cdots, p-2$ occur exactly once in the sequence [13].

Lemma 2: Let u and v be the x-th and y-th numbers in the sequence defined in Lemma 1, if $0 \le u, v \le p-2$, and $u+v = p-2$, then $x+y = p$.

Proof: From $u \equiv p-1+x\delta \pmod p$ and $v \equiv p-1+y\delta \pmod p$, we have $u+v \equiv 2p-2+(x+y)\delta \equiv (x+y)\delta - 2 \pmod p$. If $u+v = p-2$, the above equation can be reduced to $p-2 \equiv (x+y)\delta - 2 \pmod p$, i.e., $(x+y)\delta \equiv 0 \pmod p$. Since p is a prime number and $0 < \delta < p$, we have $x+y \equiv 0 \pmod p$. On the other hand, from $0 \le u, v \le p-2$, and Lemma 1, we have $1 \le x, y \le p-1$, thus $2 \le x+y \le 2p-2$. From the above, it can be easily deduced that $x+y = p$. □

Theorem 1: For any odd prime p, S-Code has column distance of 3, i.e., it is MDS.

Proof: Since S-Code is a class of linear codes, its column distance is equal to its minimum column weight. Thus we only need to prove that a valid codeword of S-Code has at least three nonzero columns. First, from the construction of S-Code, each parity bit is obtained along a certain diagonal, so it is clearly impossible to have only one nonzero column.

Now suppose that there is a valid codeword that contains exactly two nonzero columns—the lth and rth columns, where $0 \le l < r \le p-1$. Let $\delta = r-l$, obviously $1 \le \delta \le p-1$, and let $\rho(x) = \langle p-1+x\delta \rangle$. Then, according to Lemma 1, $b_{0,j}, b_{1,j}, \cdots, b_{p-2,j}$ can be rearranged as $b_{\rho(1),j}, b_{\rho(2),j}, \cdots, b_{\rho(p-1),j}$, where $j = l, r$. And these two sequences can be further rearranged as $b_{\rho(1),r}, b_{\rho(2),l}, b_{\rho(3),r}, \cdots, b_{\rho(p-1),l}$ (Seq. A) and $b_{\rho(1),l}, b_{\rho(2),r}, b_{\rho(3),l}, \cdots, b_{\rho(p-1),r}$ (Seq. B). We distinguish between the following two cases.

Case 1: $l = 0$ and $1 \le r \le p-1$. The rth column contains two parity bits $b_{r-1,r}$ and $b_{p-1-r,r}$. Note that $\rho(1) = r-1$ and $\rho(p-1) = p-1-r$, thus the two parity bits are the *first* element of Seq. A and the *last* element of Seq. B respectively. Observe that $b_{\rho(1),l}$ and $b_{p-1,r}$ are in the same diagonal, and that $b_{p-1,r} = 0$, we have $b_{\rho(1),l} = 0$. Since $b_{\rho(1),l}$ and $b_{\rho(2),r}$ are in the same anti-diagonal, we can further deduce that $b_{\rho(2),r} = 0$. This zigzag recursion stops at $b_{\rho(p-1),r}$, which is a parity bit that only lies in anti-diagonal. Similarly, since $b_{p-1-\delta,l}$ and $b_{p-1,r}$ are in the same anti-diagonal, we have $b_{p-1-\delta,l} = 0$, i.e., $b_{\rho(p-1),l} = 0$. By parity of reasoning, we can deduce that every element of Seq. A is also zero.

Case 2: $1 \le l < r \le p-1$. There are four parity bits: $b_{l-1,l}$, $b_{p-1-l,l}$, $b_{r-1,r}$, and $b_{p-1-r,r}$. According to Lemma 1, there must be x and y such that $\rho(x) = l-1$ and $\rho(y) = p-1-l$. Since $l-1+(p-1-l) = p-2$, according to Lemma 2, we have $x+y = p$. Additionally, notice that $l-1+\delta = r-1$ and $p-1-l-\delta = p-1-r$, we have $\rho(x+1) = r-1$ and $\rho(y-1) = p-1-r$.

If x is odd, then y must be even (since $x+y = p$), moreover, $x+1$ is even and $y-1$ is odd. In this case, the two parity bits $b_{\rho(y),l}$ and $b_{\rho(y-1),r}$ nicely fall into Seq. A, while $b_{\rho(x),l}$ and $b_{\rho(x+1),r}$ nicely fall into Seq. B. Then, observe that $b_{p-1,l}$ and $b_{\rho(1),r}$ are in the same anti-diagonal and $b_{p-1,l} = 0$, we have $b_{\rho(1),r} = 0$. If $b_{\rho(1),r}$ is not a parity bit, i.e., $y \neq 2$, then $b_{\rho(1),r}$ and $b_{\rho(2),l}$ are in the same diagonal, thus we can further deduce that $b_{\rho(2),l} = 0$. By parity of reasoning, we have $b_{p-1,l} = b_{\rho(1),r} = \cdots = b_{\rho(y-1),r} = 0$. Similarly, starting from imaginary 0-bit $b_{\rho(p),r}$, we can deduce that $b_{\rho(p-1),l} = \cdots = b_{\rho(y),l} = 0$ using anti-diagonal and diagonal parities alternately. Since $b_{\rho(y-1),r}$ and $b_{\rho(y),l}$ are adjacent elements in Seq. A, we have $b_{\rho(1),r} = b_{\rho(2),l} = \cdots = b_{\rho(p-1),l} = 0$. Similarly, it can be easily deduced that every element of Seq. B also equals zero.

If x is even, then y and $x+1$ are odd, thus $y-1$ is even. Therefore, $b_{\rho(y),l}$ and $b_{\rho(y-1),r}$ nicely fall into Seq. B, while $b_{\rho(x),l}$ and $b_{\rho(x+1),r}$ nicely fall into Seq. A. Similarly, we can deduce that every element of Seq. A and Seq. B equals zero.

From the above, it is clear that the minimum column weight of S-Code is at least 3. But clearly there is a codeword of column weight 3, thus the column distance of S-Code is 3. □

Remark: We refer to the zigzag recursions in the proof as *decoding chains*, since they actually indicate the reconstruction sequence of the missing bits in decoding.

III. Efficient Decoding Algorithms

In this section, we give two efficent decoding algorithms for correcting two erasures and one erasure respectively.

A. Correcting Two Erasures

According to (1) and (2), the parity sets are along diagonals $\{(x,y)|x+y\equiv 2j-1(\mathrm{mod}\ p)\}$ and $\{(x,y)|x-y\equiv p-1-2j(\mathrm{mod}\ p)\}$ respectively, where $j=1,2,\cdots,p-1$. Thus, according to Lemma 1, the parity constraints of S-Code can also be expressed as:

$$\bigoplus_{t=0}^{p-1} b_{\langle u-t\rangle,t}=0 \tag{3}$$

$$\bigoplus_{t=0}^{p-1} b_{\langle u+t\rangle,t}=0 \tag{4}$$

where $u=0,1,\cdots,p-2$. Now we present the formal decoding algorithm, noting that its correctness can be easily deduced from the proof of Theorem 1.

Algorithm 1 (Two-Erasure Decoding Algorithm): Assume that the lth and rth columns have been erased, where $0\le l<r\le p-1$. First, we calculate the diagonal syndromes $S^{(1)}=S_0^{(1)},S_1^{(1)},\cdots,S_{p-2}^{(1)}$, and the anti-diagonal syndromes $S^{(-1)}=S_0^{(-1)},S_1^{(-1)},\cdots,S_{p-2}^{(-1)}$ as follows:

$$S_u^{(1)}=\bigoplus_{t=0,t\neq l,r}^{p-1} b_{\langle u-t\rangle,t} \tag{5}$$

$$S_u^{(-1)}=\bigoplus_{t=0,t\neq l,r}^{p-1} b_{\langle u+t\rangle,t} \tag{6}$$

where $u=0,1,\cdots,p-2$. Then, from (3)–(6) we have

$$b_{\langle u-l\rangle,l}\oplus b_{\langle u-r\rangle,r}=S_u^{(1)} \tag{7}$$

$$b_{\langle u+l\rangle,l}\oplus b_{\langle u+r\rangle,r}=S_u^{(-1)} \tag{8}$$

where $u=0,1,\cdots,p-2$.

Next, according to the values of l and r, we have the following two cases:

a) $l=0$ and $1\le r\le p-1$. First, according to (7) and (8) we have $b_{r-1,0}\oplus b_{p-1,r}=S_{r-1}^{(1)}$ and $b_{p-1-r,0}\oplus b_{p-1,r}=S_{p-1-r}^{(-1)}$. Since $b_{p-1,r}$ is an imaginary 0-bit, we actually have determined the values of $b_{r-1,0}$ and $b_{p-1-r,0}$. Once $b_{r-1,0}$ and $b_{p-1-r,0}$ are determined, all the other missing bits can be retrieved along two decoding chains as discussed in Theorem 1, using (7) and (8) alternately.

b) $1\le l<r\le p-1$. Like the last case, from $b_{p-1,l}=0$ we can deduce that $b_{p-1-(r-l),r}=S_{l-1}^{(1)}$ and $b_{r-l-1,r}=S_{p-1-l}^{(-1)}$. From $b_{p-1,r}=0$ we can deduce that $b_{p-1-(r-l),l}=S_{p-1-r}^{(-1)}$ and $b_{r-l-1,l}=S_{r-1}^{(1)}$. Then, as discussed in Theorem 1, start from these four bits, we can retrieve all the other missing bits along four decoding chains respectively, using (7) and (8) alternately in each decoding chain. □

In the above algorithm, every missing bit is retrieved by XORing the other $p-1$ bits along the diagonal or anti-diagonal that traverses it. Since there is an imaginary 0-bit among the $p-1$ bits, we need only $p-3$ XOR operations for reconstructing each missing bit. Therefore, *the decoding complexity of S-Code is also optimal*.

B. Optimal Recovery From a Single Erasure

If only one column is erased, we can reduce the I/O cost by reusing certain surviving bits in the reconstruction. Observe that diagonals of different slopes necessarily intersect with each other in the $p\times p$ array, if we reconstruct some lost data bits using diagonal parities and reconstruct the rest using anti-diagonal parities, then the surviving bits at the intersection points of those selected diagonals and anti-diagonals are used twice in the reconstruction. In order to maximize the number of intersection points, we should *reconstruct half of the missing bits using diagonal parities, and reconstruct the other half using anti-diagonal parities*. Moreover, we should also try to avoid generating intersection points lying in the $(p-1)$th row, i.e., the imaginary 0-row. To facilitate the following discussion, we first present a useful theorem.

Theorem 2: For the f-th column, where $0\le f\le p-1$, let $b_{x,y}$ be the intersection point of the diagonal that crosses $b_{i,f}$ and the anti-diagonal that crosses $b_{j,f}$, then

a) $b_{x,y}$ lies in the $(p-1)$th row, if $i+j=p-2$.
b) $b_{x,y}$ lies in the 0-th column, if $j-i\equiv 2f(\mathrm{mod}\ p)$.

Proof: The diagonal of slope 1 that crosses $b_{i,f}$ is $\{(x,y)|x+y\equiv i+f(\mathrm{mod}\ p)\}$, and the diagonal of slope -1 that crosses $b_{j,f}$ is $\{(x,y)|x-y\equiv j-f(\mathrm{mod}\ p)\}$. Since $b_{x,y}$ is the intersection point, the following equations hold:

$$x+y\equiv i+f \pmod p$$

$$x-y\equiv j-f \pmod p$$

For a), adding the two equations above, we have $2x\equiv i+j(\mathrm{mod}\ p)$, i.e., $2x\equiv p-2(\mathrm{mod}\ p)$. Since $p-2$ is an odd number, we have $x\equiv\frac{p-2}{2}\equiv\frac{p-2+p}{2}\equiv p-1(\mathrm{mod}\ p)$, i.e., $b_{x,y}$ lies in the $(p-1)$th row. For b), the difference of the two equations is $-2y\equiv j-i-2f(\mathrm{mod}\ p)$. If $j-i\equiv 2f(\mathrm{mod}\ p)$, we have $y\equiv 0(\mathrm{mod}\ p)$, i.e., $b_{x,y}$ lies in the 0-th column. □

According to Theorem 2, *it is best to reconstruct the ith missing bit and the $(p-2-i)$th missing bit of the erased column using the same type of parities*.

For shortened S-Code, since the 0-th column of the array is an imaginary 0-column, we should also minimize the number of intersection points that lie in this column. Now suppose that the f-th column is erased, clearly $1\le f\le p-1$. *To facilitate the following description, we let* $\varphi(x)=\langle p-1+x\cdot f\rangle$. Then, according to Lemma 1, the missing bits occur exactly once in the sequence $b_{\varphi(1),f},b_{\varphi(2),f},\cdots,b_{\varphi(p-1),f}$, which can be divided into $(p-1)/2$ pairs $\{(b_{\varphi(k),f},b_{\varphi(p-k),f})|k=1,3,\cdots,p-2\}$. Since $\varphi(k)+\varphi(p-k)\equiv 2p-2+pf\equiv p-2(\mathrm{mod}\ p)$, according to Theorem 2 it is best to reconstruct $b_{\varphi(k),f}$ and $b_{\varphi(p-k),f}$ using the same type of parities. Following this principle, we have another theorem.

Theorem 3: If we reconstruct two adjacent pairs of missing bits in $\{(b_{\varphi(k),f}, b_{\varphi(p-k),f})|k = 1, 3, \cdots, p-2\}$ using different types of parities, then the corresponding four diagonals have at least one intersection point in the 0-th column.

Proof: Suppose that the two adjacent pairs of missing bits are $(b_{\varphi(k),f}, b_{\varphi(p-k),f})$ and $(b_{\varphi(k+2),f}, b_{\varphi(p-k-2),f})$, where k is odd and $1 \leq k \leq p-4$. First, if we reconstruct the first pair using diagonal parities and reconstruct the second pair using anti-diagonal parities, then according to Theorem 2, the intersection point of the diagonal that crosses $b_{\varphi(k),f}$ and the anti-diagonal that crosses $b_{\varphi(k+2),f}$ lies in the 0-th column. Similarly, if we reconstruct the first pair using anti-diagonal parities and reconstruct the second pair using diagonal parities, then the intersection point of the diagonal that crosses $b_{\varphi(p-k-2),f}$ and the anti-diagonal that crosses $b_{\varphi(p-k),f}$ lies in the 0-th column. □

According to the above theorem, in order to minimize the number of intersection points that lie in the 0-th column, *it is best to reconstruct every two adjacent pairs of missing bits using the same type of parities*.

From the above, a formal algorithm for efficiently rebuilding a single erasure can be described as follows:

Algorithm 2 (Single-Erasure Decoding Algorithm): Assume that the f-th column is erased. Let $m = (p-1)/2$, then we distinguish between the following two cases:

$f = 0$. First, divide the missing bits into m pairs $\{(b_{i,f}, b_{p-2-i,f})|i = 0, 1, \cdots, m-1\}$. If m is even, then reconstruct the first $m/2$ pairs using diagonal parities and reconstruct the other $m/2$ pairs using anti-diagonal parities. Otherwise, execute the following three steps: 1) reconstruct the first pair of missing bits using different types of parities, 2) reconstruct the next $(m-1)/2$ pairs of missing bits using diagonal parities, and 3) reconstruct the remaining $(m-1)/2$ pairs of missing bits using anti-diagonal parities.

$1 \leq f \leq p-1$. First, divide the missing bits into m pairs $\{(b_{\varphi(k),f}, b_{\varphi(p-k),f})|k = 1, 3, \cdots, p-2\}$. Obviously, the first pair is a pair of missing parity bits, which can be retrieved according to the encoding rules. If m is even, we reconstruct the $m-1$ pairs of missing data bits as follows: 1) reconstruct the first pair of missing data bits using different types of parities, 2) reconstruct the next $(m-2)/2$ pairs of missing data bits using diagonal parities, and 3) reconstruct the remaining $(m-2)/2$ pairs of missing data bits using anti-diagonal parities. Otherwise, $m-1$ must be even. Then, reconstruct the first $(m-1)/2$ pairs of missing data bits using diagonal parities and reconstruct the other $(m-1)/2$ pairs of missing data bits using anti-diagonal parities. □

To evaluate the performance of S-Code in terms of single-erasure recovery, we compare its I/O cost with that of existing *lowest density* MDS array codes and *repair-optimal* MDS array codes. First, we introduce the notion of *rebuilding ratio* [10], which refers to the ratio of *the number of surviving bits required for recovery* to *the number of all surviving bits*. Since the rebuilding ratio generally depends on the position of the erased column, we take the *average rebuilding ratio* (ARR) to evaluate a given code. Table I shows the main results.

It is clear that the ARR of S-Code is less than that of other *lowest density* MDS array codes in most cases, and is very close to that of *repair-optimal* MDS array codes when the code length is small. In particular, the S-Code of length 4 exactly achieves the theoretical lower bound in ARR. Note that codes [10]–[12] are specially designed for minimizing the ARR and hence outperform *lowest density* MDS array codes in single-erasure recovery, however, *they are usually inferior to the latter in terms of encoding, decoding, and update*.

TABLE I
AVERAGE REBUILDING RATIOS OF LOWEST DENSITY MDS ARRAY CODES AND REPAIR-OPTIMAL MDS ARRAY CODES

Code length	4	5	6	7	10	11	∞
X-Code[2][9]	—	0.60	—	0.62	—	0.65	0.75
RDP[13][8]	0.54	0.60	0.63	0.64	—	0.68	0.75
P-Code[14]	**0.50**	0.63	0.60	0.67	0.67	0.70	0.75
S-Code	**0.50**	0.60	0.60	0.61	0.64	0.66	0.75
[10] and [11]	0.58	0.60	0.60	0.60	0.58	0.57	0.50
MDR[12]	0.54	0.55	0.55	0.55	0.54	0.54	0.50

IV. CONCLUSION

We have presented a new family of lowest density MDS array codes for correcting double erasures in storage systems. These codes achieve the lower bounds of encoding, decoding and update complexity when the code length is p or $p-1$ with p being an odd prime. For single erasures, our decoding algorithm yields the best recovery scheme and it requires less I/O cost than most existing lowest density MDS array codes. Moreover, the I/O cost either achieves or approaches the theoretical lower bound when the code length is small.

REFERENCES

[1] M. Blaum, P. G. Farrell, and H. C. van Tilborg, "Chapter on array codes," *Handbook of Coding Theory*, vol. 2, pp. 1855–1909, 1998.

[2] L. Xu and J. Bruck, "X-code: MDS array codes with optimal encoding," *IEEE Trans. Inf. Theory*, vol. 45, no. 1, pp. 272–276, Jan. 1999.

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

[4] E. Louidor and R. Roth, "Lowest density mds codes over extension alphabets," *IEEE Trans. Inf. Theory*, vol. 52, no. 7, pp. 3186–3197, Jul. 2006.

[5] Y. Cassuto and J. Bruck, "Cyclic lowest density MDS array codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 4, pp. 1721–1729, Apr. 2009.

[6] M. Sandell and F. Tosato, "Lowest density MDS array codes on incomplete graphs," in *Proc. 51st Annu. Allerton Conf. Commun., Control, Comput.*, Oct. 2013, pp. 645–652.

[7] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you?" in *Proc. FAST*, 2007, pp. 1–17.

[8] L. Xiang, Y. Xu, J. Lui, and Q. Chang, "Optimal recovery of single disk failure in RDP code storage systems," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 1, pp. 119–130, Jun. 2010.

[9] S. Xu *et al.*, "Single disk failure recovery for x-code-based parallel storage systems," *IEEE Trans. Comput.*, vol. 63, no. 4, pp. 995–1007, Apr. 2014.

[10] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597–1616, Mar. 2013.

[11] E. En Gad, R. Mateescu, F. Blagojevic, C. Guyot, and Z. Bandic, "Repair-optimal MDS array codes over gf(2)," in *Proc. IEEE ISIT*, Jul. 2013, pp. 887–891.

[12] Y. Wang, X. Yin, and X. Wang., "MDR codes: A new class of raid-6 codes with optimal rebuilding and encoding," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 1008–1018, May 2014.

[13] P. Corbett *et al.*, "Row-diagonal parity for double disk failure correction," in *Proc. 3rd USENIX Conf. File Storage Technol.*, 2004, pp. 1–14.

[14] C. Jin, H. Jiang, D. Feng, and L. Tian, "P-code: A new raid-6 code with optimal properties," in *Proc. 23rd Int. Conf. Supercomput. ACM*, 2009, pp. 360–369.