Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-024-01440-8

MDS array codes with efficient repair and small sub-packetization level

Lei Li[1] · Xinchun Yu[2] · Chenhao Ying[1] · Liang Chen[3] · Yuanyuan Dong[3] · Yuan Luo[1]

Received: 16 December 2023 / Revised: 25 May 2024 / Accepted: 1 June 2024

Abstract
Modern data centers use erasure codes to provide high storage efficiency and fault tolerance. Reed–Solomon code is commonly deployed in large-scale distributed storage systems due to its ease of implementation, but it consumes massive bandwidth during node repair. Minimum storage regenerating (MSR) codes is a class of maximum distance separable (MDS) codes that achieve the lower bound on repair bandwidth. However, an exponential sub-packetization level is inevitable for MSR codes, resulting in massive disk I/O consumption during node repair. Disk I/O is becoming the bottleneck of the performance in data centers where the storage system needs to frequently provide high-speed data access to clients. In this paper, we consider disk I/O as an important metric to evaluate the performance of a code and construct MDS array codes with efficient repair under small sub-packetization level. Specifically, two explicit families of MDS codes with efficient repair are proposed at the sub-packetization level of $\mathcal{O}(r)$, where r denotes the number of parities. The first family of codes are constructed over a finite field $\mathbb{F}_{q^m}$ where $q \geq n$ is a prime power, $m > r(l-1)+1$, n and l denote the code length and sub-packetization level, respectively. The second family of codes are built

Communicated by T. Feng.

✉ Yuan Luo
yuanluo@sjtu.edu.cn

Lei Li
staroverseas@sjtu.edu.cn

Xinchun Yu
yuxinchun@sz.tsinghua.edu.cn

Chenhao Ying
yingchenhao@sjtu.edu.cn

Liang Chen
cl304641@alibaba-inc.com

Yuanyuan Dong
yuanyuan.dyy@alibaba-inc.com

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Dongchuan Road, Shanghai 200240, China

[2] Institute of Data and Information, Shenzhen International Graduate School of Tsinghua University, Lishui Road, Shenzhen 518055, Guangdong, China

[3] Alibaba Group, West Wenyi Road, Hangzhou 310030, Zhejiang, China

Published online: 23 July 2024

upon a special binary polynomial ring where the computation operations during node repair and file reconstruction are only XORs and cyclic shifts, avoiding complex multiplications and divisions over large finite fields.

Keywords Distributed storage system · MDS array code · Sub-packetization level · Repair bandwidth

Mathematics Subject Classification 94A24

1 Introduction

Distributed storage systems (DSS), such as the Google File System [5], Hadoop Distributed File System [20], and Microsoft Azure [3], are built upon a large number of individually unreliable nodes to store and analyze massive data, where node failures may occur as daily events. To provide data availability and reliability in the face of node failures, data redundancy is usually introduced into the storage system. The traditional scheme for introducing redundancy is triple replication. In order to save storage costs, erasure codes (ECs) are introduced as an alternative to replication since they achieve the higher reliability for fixed redundancy. Maximum distance separable (MDS) codes, such as Reed–Solomon code [15], are a class of EC that achieve the optimal trade-off between storage efficiency and fault tolerance. For a file of size $\mathcal{M}$, the storage system using an (n, k) MDS code first divides the file into k packets, each of size $\frac{\mathcal{M}}{k}$, and then encodes them into n packets which are distributed over n distinct storage nodes. The MDS property guarantees reconstruction of the original file as long as any k out of these n packets are accessible.

In real-world data centers, single-node failure is the most frequent failure pattern and it is of great significance to repair the failed node in a timely manner. During node repair, the amount of data communicated from helper nodes to the replacement node is defined as repair bandwidth in the literature. For systems using an (n, k) scalar MDS code, such as RS code, the failed node can be repaired by accessing and communicating the data stored on any k out of n surviving nodes, i.e., the repair bandwidth is k times of the data stored on the failed node. The repair problem was first formulated in the seminal work [4], wherein a trade-off between storage and repair bandwidth was derived. The two extremal points on the optimal trade-off curve, corresponding to the best storage efficiency and the minimum repair bandwidth, are called minimum storage regenerating (MSR) codes and minimum bandwidth regenerating (MBR) codes, respectively. MSR codes have drawn much attraction due to their optimal storage efficiency, and in our previous work [8] we have proposed binary MSR codes over the same binary polynomial ring, i.e., $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$ where p is a prime, as used in the second family of codes in the present paper. For more details about the constructions of MSR codes, one can refer to [1, 2, 10–13, 17, 18, 21–27, 29–31].

Although various constructions of MSR codes have been proposed in the last decade, MSR code is rarely deployed in real-world data centers because of its large sub-packetization level. The large sub-packetization level is not friendly to metadata management and will restrict the minimum size of files that can be handled by the code. An additional and more severe problem of MSR codes is the large amount of disk I/Os (random data access) consumed during node repair [19]. For disk I/O intensive applications such as cloud computing, frequent requests for data read and write need to be granted in a timely manner. Consequently, the disk I/O performance is becoming the bottleneck that restricts the performance of DSS.

MDS array codes that provide small repair bandwidth as well as small sub-packetization level are of great significance for real-world data centers. In [14], the authors proposed a class of MDS array codes of sub-packetization level $l = r$, where r is the number of parity nodes. The same structure as in [14] was used in [6] over the binary polynomial ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$ to produce binary MDS array codes with small sub-packetization level and efficient repair, where the ability to repair with busy-node of the structure was also presented. In [28], the authors obtained the lower bound on repair bandwidth for MDS array codes with $l = r = 2$ and presented explicit constructions of codes with degraded read friendly. In our previous work [9], we constructed MDS array codes with efficient repair for systematic nodes where the sub-packetization level $l = 2$ and the redundancy $r \geq 2$. In the present paper, we propose two families of MDS array codes with small sub-packetization level $l = \mathcal{O}(r)$ and efficient repair for all nodes. The first family of codes can be viewed as the generalization of the codes with sub-packetization level $l = r$ in [14]. The second family of codes is constructed over the binary polynomial ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$ and is the generalization of the codes in [6]. In addition, for the second family of codes, we propose an algorithm to obtain the minimum prime p that guarantees the MDS property of the code.

The rest of this paper is organized as follows. Section 2 provides some preliminaries that are used in the paper. In Sect. 3, we present two general constructions of MDS array codes over finite field and binary polynomial ring $\mathcal{R} = \mathbb{F}_2[x]/(1+x+x^2+\cdots+x^{p-1})$, following two toy examples respectively. Evaluations of the proposed codes are given in Sect. 4 and Sect. 5 concludes the paper.

2 Preliminaries

Given two integers i and j with $i < j$, define $[i] := \{1, 2, \cdots, i\}$ and $[i, j] := \{i, i + 1, \cdots, j\}$. We use $i \mid j$ and $i \nmid j$ to denote that i divides j and i does not divide j, respectively. Following the literature of codes for distributed storage, we use the two words "coordinate" and "node" interchangeably. In this paper, we denote by $r := n - k$ the number of parity nodes of an (n, k) code.

2.1 MDS array code

Let $\mathbb{F}_q$ be a finite field of size q, where q is a prime power. For a distributed storage system using an (n, k) MDS array code of sub-packetization level l, the codeword can be written as $(C_1, C_2, \ldots, C_n)$. Each coordinate of the codeword $C_i = (c_{i,1}, c_{i,2}, \ldots, c_{i,l})^T \in \mathbb{F}_q^l$ is a column vector of length l over the field $\mathbb{F}_q$. In this paper, we define an (n, k, l) MDS array code $\mathcal{C}$ by its parity-check equations

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} H_{t,i} C_i = \mathbf{0}, t \in [r]\}, \tag{1}$$

where $H_{t,i}, t \in [r], i \in [n]$ are $l \times l$ matrices over some field $\mathbb{F}$ and $\mathbf{0}$ in boldface is a column vector of length rl over $\mathbb{F}$. The parity-check matrix of the code $\mathcal{C}$ can be written as

$$H = \begin{bmatrix} H_{1,1} & H_{1,2} & \ldots & H_{1,n} \\ H_{2,1} & H_{2,2} & \ldots & H_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ H_{r,1} & H_{r,2} & \ldots & H_{r,n} \end{bmatrix}. \tag{2}$$

Obviously, H is an $rl \times nl$ matrix over $\mathbb{F}$ and each row corresponds to a parity-check equation. In this paper, we say that every l rows in H form a parity-check group, i.e., H consists of r parity-check groups, each of which can be written as $\{H_{t,i} : i \in [n]\}$ for some $t \in [r]$. The code $\mathcal{C}$ defined by (1) and (2) is MDS if any $r \times r$ sub-block matrix of H is invertible.

2.2 A binary polynomial ring

For the binary field $\mathbb{F}_2$ and an odd prime p, let $\mathcal{R}$ be the ring of polynomials of degree less than $p-1$ over $\mathbb{F}_2$ where the multiplication is modulo $1 + x + x^2 + \cdots + x^{p-1}$, i.e., $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$. Let $\mathcal{R}^*$ denote the multiplicative group of polynomials in $\mathcal{R}$, which are relatively prime to $1 + x + x^2 + \cdots + x^{p-1}$. Clearly, the multiplication operation in $\mathcal{R}$ is commutative.

In Sect. 3.3 of the present paper, we use special elements in $\mathcal{R}^*$ to construct the codes with desired properties. It is beneficial to introduce some of the elements in $\mathcal{R}^*$ before giving the code constructions. As $\gcd(x, 1 + x + x^2 + \cdots + x^{p-1}) = 1$, where gcd() is the greatest common divisor of the input arguments, we conclude that $x \in \mathcal{R}^*$. Obviously, for any $i \in [p-2]$, we have $x^i \in \mathcal{R}^*$. Note that $x^p - 1 = (x-1)(1 + x + x^2 + \cdots + x^{p-1})$ and for $1 \le i \in [p-2]$,

$$\gcd(x^i - 1, x^p - 1) = x^{\gcd(i,p)} - 1 = x - 1.$$

Since p is not the characteristic of $\mathbb{F}_2$, we have that 1 is not a root of $1+x+x^2+\cdots+x^{p-1} = 0$, i.e., $\gcd(x-1, 1+x+x^2+\cdots+x^{p-1}) = 1$. As a result, we have $\gcd(x^i - 1, 1+x+x^2+\cdots+x^{p-1}) = 1$, meaning $x^i - 1 \in \mathcal{R}^*$ and $x^i - x^j$ is also in $\mathcal{R}^*$ with $i \neq j \in [p-2]$.

3 Code construction

In this section, we present the constructions of two families of MDS array codes with efficient repair over finite field and the binary polynomial ring, respectively. The codes share the same structure, and both of them have smaller sub-packetization level compared with related works.

3.1 A toy example

Before giving the general construction of the code, we first present an example code to highlight the core structure of the code.

Example 1 For $n = 8, k = 5, l = 2$, the parity-check matrix of the code is

$$H_{(8,5,2)} = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
\hline
\lambda_1 & \psi_1 & \lambda_2 & \psi_2 & \lambda_3 & 0 & \lambda_4 & 0 & \lambda_5 & 0 & \lambda_6 & 0 & \lambda_7 & 0 & \lambda_8 & 0 \\
0 & \lambda_1 & 0 & \lambda_2 & \psi_3 & \lambda_3 & \psi_4 & \lambda_4 & 0 & \lambda_5 & 0 & \lambda_6 & 0 & \lambda_7 & 0 & \lambda_8 \\
\hline
\lambda_1^2 & 0 & \lambda_2^2 & 0 & \lambda_3^2 & 0 & \lambda_4^2 & 0 & \lambda_5^2 & \psi_5 & \lambda_6^2 & \psi_6 & \lambda_7^2 & 0 & \lambda_8^2 & 0 \\
0 & \lambda_1^2 & 0 & \lambda_2^2 & 0 & \lambda_3^2 & 0 & \lambda_4^2 & 0 & \lambda_5^2 & 0 & \lambda_6^2 & \psi_7 & \lambda_7^2 & \psi_8 & \lambda_8^2
\end{array}\right], \qquad (3)$$

where the 16 nonzero entries $\lambda_1 = 197$, $\lambda_2 = 43$, $\lambda_3 = 219$, $\lambda_4 = 250$, $\lambda_5 = 130$, $\lambda_6 = 222$, $\lambda_7 = 147$, $\lambda_8 = 39$, $\psi_1 = 50$, $\psi_2 = 101$, $\psi_3 = 184$, $\psi_4 = 202$, $\psi_5 = 192$, $\psi_6 = 78$, $\psi_7 = 129$, $\psi_8 = 22$ are drawn from the field $\mathbb{F}_{2^8}$ with primitive polynomial equal to $x^8 + x^4 + x^3 + x^2 + 1$. Note that these "integers" $197, 43, \ldots, 22$ are not really integers drawn from [0, 255], they are the decimal representation of binary numbers formed by the coefficients of polynomials corresponding to the elements. With these assignments and the help of a computer problem, one can verify that any 3×3 sub-block matrix of $H_{(8,5,2)}$ is nonsingular, i.e., the code in this example is MDS code. The parity-check matrix can be viewed as the sum of a block Vandermonde matrix V and a perturbation matrix P with

$$V = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
\hline
\lambda_1 & 0 & \lambda_2 & 0 & \lambda_3 & 0 & \lambda_4 & 0 & \lambda_5 & 0 & \lambda_6 & 0 & \lambda_7 & 0 & \lambda_8 & 0 \\
0 & \lambda_1 & 0 & \lambda_2 & 0 & \lambda_3 & 0 & \lambda_4 & 0 & \lambda_5 & 0 & \lambda_6 & 0 & \lambda_7 & 0 & \lambda_8 \\
\hline
\lambda_1^2 & 0 & \lambda_2^2 & 0 & \lambda_3^2 & 0 & \lambda_4^2 & 0 & \lambda_5^2 & 0 & \lambda_6^2 & 0 & \lambda_7^2 & 0 & \lambda_8^2 & 0 \\
0 & \lambda_1^2 & 0 & \lambda_2^2 & 0 & \lambda_3^2 & 0 & \lambda_4^2 & 0 & \lambda_5^2 & 0 & \lambda_6^2 & 0 & \lambda_7^2 & 0 & \lambda_8^2
\end{array}\right]$$

and

$$P = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & \psi_1 & 0 & \psi_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \psi_3 & 0 & \psi_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \psi_5 & 0 & \psi_6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \psi_7 & 0 & \psi_8 & 0
\end{array}\right].$$

In this example, the $n = 8$ nodes are divided into 4 groups of equal size. There are $rl = 6$ parity-check equations in total and each row of $H_{(8,5,2)}$ defines one parity-check equation of the code. For the repair of node in the first group, i.e., node 1 or node 2, we use the first and the third rows of $H_{(8,5,2)}$ which form the following linear system

$$\begin{aligned}
& c_{1,1} + c_{2,1} + \sum_{j=3}^{8} c_{j,1} = 0 \\
& \lambda_1 c_{1,1} + \psi_1 c_{1,2} + \lambda_2 c_{2,1} + \psi_2 c_{2,2} + \sum_{j=3}^{8} \lambda_i c_{j,1} = 0
\end{aligned}. \tag{4}$$

Rewrite (4) in the matrix form, we have

$$H^{(1,3)} C^T = \mathbf{0}, \tag{5}$$

where

$$H^{(1,3)} = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\lambda_1 & \psi_1 & \lambda_2 & \psi_2 & \lambda_3 & 0 & \lambda_4 & 0 & \lambda_5 & 0 & \lambda_6 & 0 & \lambda_7 & 0 & \lambda_8 & 0
\end{array}\right]$$

and $C = [c_{1,1}, c_{1,2}, c_{2,1}, c_{2,2}, \cdots, c_{8,1}, c_{8,2}]^T$.

The blocks $\begin{bmatrix} 1 & 0 \\ \lambda_1 & \psi_1 \end{bmatrix}$ and $\begin{bmatrix} 1 & 0 \\ \lambda_2 & \psi_2 \end{bmatrix}$ in $H^{(1,3)}$, corresponding to $[c_{1,1}, c_{1,2}]^T$ and $[c_{2,1}, c_{2,2}]^T$, respectively, are invertible and thus, node 1 and node 2 are repairable with $H^{(1,3)}$. We take the repair of node 1 for example. By accessing and communicating the symbols $\{c_{j,1} : j \in [2, 8]\}$, the lost symbol $c_{1,1}$ can be computed through the first equation in (4). By accessing and communicating $c_{2,2}$ further, the lost symbol $c_{1,2}$ can be obtained through the second equation in (4). There are eight symbols accessed and communicated in the repair procedure, achieving a 20% reduction compared with the trivial repair of MDS code. Here, we say that $H^{(1,3)}$ is a *repair pattern*, which is obtained by selecting the first and third rows of $H_{(8,5,2)}$. Similarly, the repair pattern for node 3 and node 4 is $H^{(2,4)}$, which is obtained by selecting the second and the fourth rows of $H_{(8,5,2)}$. For node 5 and node 6, the repair pattern is $H^{(1,5)}$. For node 7 and node 8, the repair pattern is $H^{(2,6)}$. There are four repair patterns in total and thus, the $n = 8$ nodes are divided into four groups, each of which contains two nodes. We say that the repair pattern $H^{(1,3)}$ and $H^{(2,4)}$ form a *repair pattern cluster* (abbreviated as RPC). The other RPC consists of $H^{(1,5)}$ and $H^{(2,6)}$.

Note that the MDS property of the example code is not naturally obtained when the 16 distinct nonzero entries $\{\lambda_i : i \in [8]\}$ and $\{\psi_i : i \in [8]\}$ in $H_{(8,5,2)}$ are randomly chosen from the field $\mathbb{F}_{2^8}$. Actually, the values of the nonzero entries in $H_{(8,5,2)}$ are obtained with the help of a computer program to attain the MDS property. We use this example only to show the basic structure of the code and the explicit construction with MDS property will be presented in the next subsection.

3.2 General construction of MDS array code with small sub-packetization level

In this subsection, we construct MDS array code with small sub-packetization level of l with $(l-1) \mid (r-1)$, i.e., $l-1$ divides $r-1$, which is a generalization of the code with $l = r$ in [14].

Construction 1 Given integers n, r and l such that $(l-1) \mid (r-1)$, let $g = l(r-1)/(l-1)$. Assume that $g \mid n$ and $n = gs$. The parity-check matrix H of the code with parameters (n, k, l) is constructed based on a block Vandermonde matrix V which can be written as

$$V = \left[\begin{array}{cccc|cccc|c|cccc}
1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & & 1 & 0 & \cdots & 0 \\
0 & 1 & \cdots & 0 & 0 & 1 & \cdots & 0 & & 0 & 1 & \cdots & 0 \\
\vdots & \cdots & \ddots & \vdots & \vdots & \cdots & \ddots & \vdots & \cdots & \vdots & \cdots & \ddots & \vdots \\
0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 1 & & 0 & \cdots & 0 & 1 \\
\hline
\lambda_1 & 0 & \cdots & 0 & \lambda_2 & 0 & \cdots & 0 & & \lambda_n & 0 & \cdots & 0 \\
0 & \lambda_1 & \cdots & 0 & 0 & \lambda_2 & \cdots & 0 & & 0 & \lambda_n & \cdots & 0 \\
\vdots & \cdots & \ddots & \vdots & \vdots & \cdots & \ddots & \vdots & \cdots & \vdots & \cdots & \ddots & \vdots \\
0 & \cdots & 0 & \lambda_1 & 0 & \cdots & 0 & \lambda_2 & & 0 & \cdots & 0 & \lambda_n \\
\hline
 & \vdots & & & & \vdots & & & \vdots & & \vdots & & \\
\hline
\lambda_1^{r-1} & 0 & \cdots & 0 & \lambda_2^{r-1} & 0 & \cdots & 0 & & \lambda_n^{r-1} & 0 & \cdots & 0 \\
0 & \lambda_1^{r-1} & \cdots & 0 & 0 & \lambda_2^{r-1} & \cdots & 0 & & 0 & \lambda_n^{r-1} & \cdots & 0 \\
\vdots & \cdots & \ddots & \vdots & \vdots & \cdots & \ddots & \vdots & \cdots & \vdots & \cdots & \ddots & \vdots \\
0 & \cdots & 0 & \lambda_1^{r-1} & 0 & \cdots & 0 & \lambda_2^{r-1} & & 0 & \cdots & 0 & \lambda_n^{r-1}
\end{array}\right].$$

The matrix V consists of $r \times n$ blocks, each of which is an $l \times l$ diagonal matrix where the diagonal entries are $\lambda_i^{t-1}, i \in [n], t \in [r]$. $\lambda_1, \ldots, \lambda_n$ are n distinct elements drawn from field $\mathbb{F}$. Let $\mathbb{E}$ be an extension field of $\mathbb{F}$ where $\mathbb{E}$ is generated by an element $\psi \in \mathbb{E}$, i.e., $\mathbb{E} = \mathbb{F}(\psi)$, and the degree of extension $[\mathbb{E} : \mathbb{F}] \geq r(l-1)+1$. The parity-check matrix H is obtained by substituting 0 of specific positions in V with ψ. The n storage nodes are divided into g groups of equal size s. The last $r-1$ parity-check groups in V are divided into $\frac{r-1}{l-1}$ *parity-check clusters* (abbreviated as PCCs), each of which contains $l-1$ parity-check groups. Each PCC and the first parity-check group in V form a RPC. Each RPC contains l repair patterns and each repair pattern repair a group (s) of nodes. Obviously, there are $\frac{r-1}{l-1}$ RPCs, consisting of $\frac{l(r-1)}{l-1}$ repair patterns. The positions of ψ in H are as follows.

For the first parity-check group of the first PCC, blocks $[s]$ contain ψ at the position of $(1, 2)$, i.e.,

$$H_{2,i} = \begin{bmatrix} \lambda_i & \psi & \cdots & 0 \\ 0 & \lambda_i & \cdots & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_i \end{bmatrix} \tag{6}$$

for $i \in [s]$. Blocks $[s+1, 2s]$ contain ψ at the position of $(2, 3)$, i.e.,

$$H_{2,i} = \begin{bmatrix} \lambda_i & 0 & \cdots & 0 \\ 0 & \lambda_i & \psi & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_i \end{bmatrix} \tag{7}$$

for $i \in [s+1, 2s]$. And so on, blocks $[(l-2)s+1, (l-1)s]$ contain ψ at the position of $(l-1, \overline{l})$ and blocks $[(l-1)s+1, ls]$ contain ψ at the position of $(l, \overline{l+1})$ where for some positive integer m, $\overline{m} = \begin{cases} m, & m \le l \\ m-l, & m > l \end{cases}$. The rest blocks are diagonal matrices which are the same as that in V.

For the second parity-check group of the first PCC, blocks $[s]$ contain ψ at the position of $(1, \overline{3})$. Blocks $[s+1, 2s]$ contain ψ at the position of $(2, \overline{4})$. And so on, blocks $[(l-2)s+1, (l-1)s]$ contain ψ at the position of $(l-1), \overline{l+1}$ and blocks $[(l-1)s+1, ls]$ contain ψ at the position of $[(l, \overline{l+2})]$. The rest blocks are diagonal matrices which are the same as that in V.

For the $(l-1)$-th (last) parity-check group of the first PCC, blocks $[s]$ contain ψ at the position of $(1, \overline{l})$. Blocks $[s+1, 2s]$ contain ψ at the position of $(2, \overline{l+1})$. And so on, blocks $[(l-2)s+1, (l-1)s]$ contain ψ at position $(l-1), \overline{2l-2}$ and blocks $[(l-1)s+1, ls]$ contain ψ at position $(l, \overline{2l-1})$. The rest blocks are diagonal matrices which are the same as that in V.

We now provide a more rigorous description of the structure of the code by presenting the blocks $H_{t,i}$ of its parity check matrix with a general formula. For any $t \in [2, r]$, we write

$$t = (\tau, \eta) = (\tau-1)(l-1) + \eta + 1,$$

where $\tau \in [\frac{r-1}{l-1}]$ and $\eta \in [l-1]$. And, for $i \in [n]$, we write

$$i = (a, b, c) = (a-1)ls + (b-1)s + c,$$

where $a \in [\frac{r-1}{l-1}]$, and $b \in [l]$, $c \in [s]$. In other words, we use τ, η to denote the index of the PCC and the index of the parity check group within each PCC, respectively. Similarly, we use a, b, c to denote the index of cluster, the index of node group within each cluster and the index of node within each node group, respectively. Then, the entry of parity check sub-matrices $H_{t,i}$ at the xth row and the yth column is

$$H_{t,i}(x, y) = \begin{cases} \lambda_i^{t-1} & \text{if } x = y \\ \psi & \text{if } \tau = a, x = b, y = \overline{b+\eta} \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for the first PCC, the first ls blocks contain ψ and the other blocks remain unchanged. Generally, for the τ-th PCC where $\tau \in [\frac{r-1}{l-1}]$, the blocks that contain ψ are $[(\tau-1)ls+1, \tau ls]$.

The positions of ψ in these ls blocks in the τ-th PCC follow the same pattern as in the first PCC. As shown in parity-check matrix $H_{(8,5,2)}$ of the example (3), the positions of $\psi_5, \psi_6, \psi_7, \psi_8$ are (1, 2), (1, 2), (2, 1), (2, 1), which are same to the positions of $\psi_1, \psi_2, \psi_3, \psi_4$. Note that multiple distinct perturbation elements $\psi_1, \psi_2, \ldots, \psi_8$ are used in *Example* 1 while only one perturbation element ψ is used in *Construction* 1. The example shows the basic structure of the code, i.e., the Vandermonde structure and the positions of the perturbation elements. It is easier to obtain the example code with MDS property over $\mathbb{F}_{256}$ through a computer program when there are multiple perturbation elements.

The parity-check matrix of an (n, k, l) array code with $(l-1) \mid (r-1)$ is explicitly given in *Construction* 1. We now present the MDS property of the code in the following Theorem 1.

Theorem 1 *The code constructed in Construction 1 is MDS code.*

Proof To prove the MDS property of the code is equivalent to prove that every r block columns of the parity-check matrix H is invertible over $\mathbb{E}$. According to *Construction* 1, it is not hard to find that there are $l-1$ blocks containing ψ in each block column. For $\mathcal{I} \subset [n]$ with $|\mathcal{I}| = r$, the determinant of the $r \times r$ block matrix $f_{\mathcal{I}}(\psi) = \det(H_{\mathcal{I}})$ can be seen as a polynomial of ψ over $\mathbb{F}$. Clearly, the degree of $f_{\mathcal{I}}(\psi)$ is at most $r(l-1)$. Let $m(\psi)$ be the minimal polynomial of ψ over $\mathbb{F}$. We have that the degree of $m(\psi)$ is at least $r(l-1)+1$ as $\mathbb{E} = \mathbb{F}(\psi)$ and $[\mathbb{E} : \mathbb{F}] \geq r(l-1)+1$. Note that $f_{\mathcal{I}}(0) = \det(V_{\mathcal{I}}) \neq 0$ because V is the parity-check matrix of an (n, k, l) MDS array code, meaning $f_{\mathcal{I}}(\psi)$ is a nonzero polynomial. As a result, $f_{\mathcal{I}}(\psi) \neq 0$, meaning that $H_{\mathcal{I}}$ is invertible over $\mathbb{E}$ for any $\mathcal{I} \subset [n]$ with $|\mathcal{I}| = r$. This completes the proof. □

In the following Theorem 2, we present the repair bandwidth for single-node failure of the code constructed in *Construction* 1.

Theorem 2 *For n, k, l with $(l-1) \mid (r-1)$ and $\frac{l(r-1)}{l-1} \mid n$ where $r = n-k$, the repair bandwidth of the code constructed in Construction 1 is $\gamma = \left(\frac{n(l-1)}{l(r-1)} - 1\right) l + n - s$.*

Proof Note that the n storage nodes are divided into $g = \frac{l(r-1)}{l-1}$ groups, each of size $s = n/g$. Also, there are $\frac{r-1}{l-1}$ RPCs, each of which contributes l repair patterns and each repair pattern is responsible for the repair of a group of nodes. Without loss of generality, we take the repair of node 1 as an example. The first rows of the l parity-check groups in the first RPC is selected and we obtain the repair pattern $H^{(1,l+1,2l+1,\cdots,(l-1)l+1)}$. In this repair pattern, the blocks A_i corresponding to node $i \in [s]$ and A_j corresponding to node $j \in [s+1, n]$ are

$$A_i = \begin{bmatrix} 1 & 0 & \cdots & & 0 \\ \lambda_i & \psi & \cdots & & 0 \\ \lambda_i^2 & 0 & \psi & \cdots & 0 \\ \vdots & \cdots & & \ddots & \vdots \\ \lambda_i^{l-1} & 0 & \cdots & & \psi \end{bmatrix}, A_j = \begin{bmatrix} 1 & 0 & \cdots & & 0 \\ \lambda_j & 0 & \cdots & & 0 \\ \lambda_j^2 & 0 & 0 & \cdots & 0 \\ \vdots & \cdots & & \ddots & \vdots \\ \lambda_j^{l-1} & 0 & \cdots & & 0 \end{bmatrix}.$$

Thus, by accessing and communicating all the symbols $c_{i,1}, c_{i,2}, \ldots, c_{i,l}$ in nodes $i \in [2, s]$ and the first symbol $c_{j,1}$ in nodes $j \in [s+1, n]$, the lost symbols $c_{1,1}, c_{1,2}, \ldots, c_{2,l}$ can be computed through the linear system defined by $H^{(1,l+1,2l+1,\cdots,(l-1)l+1)}$. During the repair procedure, the amount of symbols accessed and communicated is $(s-1)l + n - s$. One can verify that the repair bandwidth of the code constructed in *Construction* 1 is

$$\gamma = (s-1)l + n - s = \left(\frac{n(l-1)}{l(r-1)} - 1\right) l + n - s \tag{8}$$

for every node $i \in [n]$. This completes the proof. $\square$

Remark 1 For $g \nmid n$, where $g = l(r-1)/(l-1)$, the n storage nodes are divided into g groups where $n(\text{mod} \quad g)$ groups are of size $\lceil n/g \rceil$ and the remaining groups are of size $\lfloor n/g \rfloor$, i.e., the codes with $g \nmid n$ can be directly obtained by applying *shortening technique* on codes constructed through *Construction* 1. According to the repair procedure presented above, one can easily find that the amount of data access is equal to the amount of data communicated during node repair, which is good because this means the disk I/O consumption is also small.

3.3 Constructing MDS array code over binary field

In the previous subsection, explicit construction of (n, k, l) MDS array code is presented where the size of the field is at least $n^{(n-k)(l-1)+1}$. In this subsection, we construct MDS array code over the binary field. The construction relies on the binary polynomial ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$ where $p > n$ is a prime. For the code constructed in this subsection, each coordinate of a codeword can be viewed as either a column vector of length pl over $\mathbb{F}_2$ or a column vector of length l over the ring $\mathcal{R}$. The parity check matrix of the code is an $r \times n$ block matrix over the ring $\mathcal{R}$ where each block is of size $l \times l$.

The codes in this subsection and previous subsection share the same structure and the main difference between them is the alphabet. We now give an example of binary array code which is obtained by substituting the entries of the parity check matrix in *Example* 1 with polynomials in a binary polynomial ring.

Example 2 For $n = 8, k = 5$ and $l = 2$ (sub-packetization level over the ring), the parity check matrix is

$$H = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline x & 1 & x^2 & 1 & x^3 & 0 & x^4 & 0 & x^5 & 0 & x^6 & 0 & x^7 & 0 & x^8 & 0 \\ 0 & x & 0 & x^2 & 1 & x^3 & 1 & x^4 & 0 & x^5 & 0 & x^6 & 0 & x^7 & 0 & x^8 \\ \hline x^2 & 0 & x^4 & 0 & x^6 & 0 & x^8 & 0 & x^{10} & 1 & x^{12} & 1 & x^{14} & 0 & x^{16} & 0 \\ 0 & x^2 & 0 & x^4 & 0 & x^6 & 0 & x^8 & 0 & x^{10} & 0 & x^{12} & 1 & x^{14} & 1 & x^{16} \end{array}\right], \tag{9}$$

where the entries in H are drawn from the ring $\mathbb{F}_2[x]/(1+x+x^2+\cdots+x^{10})$. Similar to the parity check matrix in *Example* 1, H in (9) can be viewed as the sum of a block Vandermonde matrix V and a perturbation matrix P over the ring, where

$$V = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline x & 0 & x^2 & 0 & x^3 & 0 & x^4 & 0 & x^5 & 0 & x^6 & 0 & x^7 & 0 & x^8 & 0 \\ 0 & x & 0 & x^2 & 0 & x^3 & 0 & x^4 & 0 & x^5 & 0 & x^6 & 0 & x^7 & 0 & x^8 \\ \hline x^2 & 0 & x^4 & 0 & x^6 & 0 & x^8 & 0 & x^{10} & 0 & x^{12} & 0 & x^{14} & 0 & x^{16} & 0 \\ 0 & x^2 & 0 & x^4 & 0 & x^6 & 0 & x^8 & 0 & x^{10} & 0 & x^{12} & 0 & x^{14} & 0 & x^{16} \end{array}\right]$$

and

$$P = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{array}\right].$$

The repair patterns in this example are the same to that in *Example* 1 and we only present the repair of node 1 for simplicity. To repair node 1, the first and the third rows of H in (9) are selected to form the repair pattern $H^{(1,3)}$ which can be written as

$$H^{(1,3)} = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ x & 1 & x^2 & 1 & x^3 & 0 & x^4 & 0 & x^5 & 0 & x^6 & 0 & x^7 & 0 & x^8 & 0 \end{array}\right].$$

Obviously, the block $\begin{bmatrix} 1 & 0 \\ x & 1 \end{bmatrix}$ is invertible in the ring $\mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{10})$. Two polynomials stored on node 2 and the first polynomials stored on nodes [3, 8] are accessed and communicated to recover the data stored on node 1. Note that the MDS property of the code in this example is not verified and we will give the conditions under which the MDS property is guaranteed in the sequel.

Construction 2 Given integers n, r and l such that $(l-1) \mid (r-1)$ and $\frac{l(r-1)}{l-1} \mid n$, the parity check matrix H of the $(n, k = n - r, (p-1)l)$ binary array code is obtained by substituting the entries in the parity check matrix of the code constructed in *Construction* 1 with special elements in the binary polynomial ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$. Specifically, define $x^0 = 1 \in \mathcal{R}$, and $\lambda_i \in \mathbb{F}$ is replaced by $x^{i-1} \in \mathcal{R}$ and ψ is replaced by $1 \in \mathcal{R}$.

The code constructed in *Construction* 2 has MDS property if any k out of the n coordinates can recover the whole codeword. It is equivalent to show that any r block columns of the parity check matrix is invertible over the ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots x^{p-1})$. In the following Theorem 3, we describe the conditions under which the MDS property of the binary array code is guaranteed.

Theorem 3 *Assume* $1 + x + x^2 + \cdots + x^{p-1}$ *can be factorized as a product of* t *irreducible polynomials* $f_1(x), f_2(x), \ldots, f_t(x)$ *over* $\mathbb{F}_2$*, i.e.,*

$$1 + x + x^2 + \cdots + x^{p-1} = f_1(x) \cdot f_2(x) \cdots f_t(x),$$

where $\deg(f_1(x)) \le \deg(f_2(x)) \le \cdots \le \deg(f_t(x))$. *If the degree of* $f_1(x)$ *satisfies*

$$\deg(f_1(x)) > rl(r-1)\left(\frac{n-r}{2} + \frac{2r-1}{6}\right),$$

then the $(n, k, (p-1)l)$ *binary array code constructed in Construction* 2 *is an MDS code.*

Proof To prove the MDS property of the binary array code constructed in *Construction* 2, we should show that for any $\mathcal{I} \in [n]$ with $|\mathcal{I}| = r$, the determinant of the r block columns $H_{\mathcal{I}}$ is invertible over the ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$. Obviously, the determinant can be viewed as a polynomial over $\mathbb{F}_2[x]$ and it is not difficult to verify that the polynomial is a non-zero polynomial. According to the Chinese remainder theorem [7], the ring $\mathcal{R}$ is isomorphic to the direct sum of the t rings $\mathbb{F}_2[x]/f_1(x), \mathbb{F}_2[x]/f_2(x), \ldots, \mathbb{F}_2[x]/f_t(x)$. If the degree of the determinant $|H_{\mathcal{I}}|$ is smaller that $\deg(f_1(x))$, then the determinant is invertible over each of the t rings $\mathbb{F}_2[x]/f_1(x), \mathbb{F}_2[x]/f_2(x), \ldots, \mathbb{F}_2[x]/f_t(x)$ and is also invertible over the ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$.

As a result, we only need to find out the maximum degree of the determinants of $H_{\mathcal{I}}$ with $\mathcal{I} \subset [n]$ and $|\mathcal{I}| = r$. According to the structure of H, it easy to find that the maximum degree of the determinant is achieved when $\mathcal{I} = [n - r + 1, n]$, i.e., the last r block columns are selected to calculate the maximum degree. The degree of the determinant of the last r block columns is

$$
\begin{aligned}
&(n-r+1)l+(n-r+2)2l+\cdots+(n-1)(r-1)l\\
&\quad=l\left((n-r+1)+2(n-r+2)+\cdots+(r-1)(n-r+r-1)\right)\\
&\quad=l\left((n-r)\sum_{i=1}^{r-1}i+\sum_{i=1}^{r-1}i^2\right)\\
&\quad=l\left((n-r)\frac{r(r-1)}{2}+\frac{r(r-1)(2r-1)}{6}\right)\\
&\quad=rl(r-1)\left(\frac{n-r}{2}+\frac{2r-1}{6}\right)
\end{aligned}
.
$$

This completes the proof. □

The repair procedure of the code constructed in *Construction* 2 is similar to the code constructed in *Construction* 1 as the two codes share the same core structure. In the following Theorem 4, we present the repair bandwidth for single-node failure of the code constructed in *Construction* 2.

Theorem 4 *Given integers n, k and l with $(l-1) \mid (r-1)$ and $\frac{l(r-1)}{l-1} \mid n$ where $r=n-k$, the repair bandwidth of the (n,k,pl) binary MDS array code constructed in Construction 2 is $\gamma=\left(\frac{n(l-1)}{l(r-1)}-1\right)l+n-s$ over the ring $\mathcal{R}=\mathbb{F}_2[x]/(1+x+x^2+\cdots+x^{p-1})$ or equally $\gamma(p-1)$ bits (over $\mathbb{F}_2$).*

Proof The proof follows similar arguments to that of Theorem 2. For the sake of completeness, we present the repair procedure of the binary MDS array code to obtain the repair bandwidth. Note that the n block columns of the parity check matrix is divided into $g=\frac{l(r-1)}{(l-1)}$ groups, each of size $s=n/g$. Similar to the code in *Construction* 1, there are $\frac{r-1}{l-1}$ RPCs in the parity check matrix, each of which contributes l repair pattern and each repair pattern is responsible for the repair of a group of nodes. Without loss of generality, we take the repair of node 1 for example. The first rows of the l parity check groups in the first RPC is selected to form the repair pattern $H^{(1,l+1,2l+1,\ldots,(l-1)l+1)}$. In this repair pattern, the blocks A_i corresponding to node $i\in[s]$ and A_j corresponding to node $j\in[s+1,n]$ are

$$
A_i=\begin{bmatrix} 1 & 0 & \cdots & & 0\\ x^i & 1 & \cdots & & 0\\ x^{2i} & 0 & 1 & \cdots & 0\\ \vdots & & \cdots & \ddots & \vdots\\ x^{i(l-1)} & 0 & \cdots & & 1\end{bmatrix},\ A_j=\begin{bmatrix} 1 & 0 & \cdots & & 0\\ x^j & 0 & \cdots & & 0\\ x^{2j} & 0 & 0 & \cdots & 0\\ \vdots & & \cdots & \ddots & \vdots\\ x^{j(l-1)} & 0 & \cdots & & 0\end{bmatrix}.
$$

Clearly, by accessing and communicating all the polynomials $c_{i,1},c_{i,2},\ldots,c_{i,l}$ in nodes $i\in[2,s]$ and the first polynomial $c_{j,1}$ in nodes $j\in[s+1,n]$, the lost polynomials $c_{1,1},c_{1,2},\ldots,c_{1,l}$ can be computed since A_1 is invertible over the ring $\mathcal{R}=\mathbb{F}_2[x]/(1+x+x^2+\cdots x^{p-1})$. The amount of polynomials accessed and communicated is $(s-1)l+n-s$ during the repair of node 1 and one can verify that the repair bandwidth (over $\mathcal{R}$) of the code constructed in *Construction* 2 is

$$
\gamma=(s-1)l+n-s=\left(\frac{n(l-1)}{l(r-1)}-1\right)l+n-s
$$

for every node $i\in[n]$. As each polynomial $\mathcal{R}$ is represented by a vector of length p over $\mathbb{F}_2$, the repair bandwidth of the code in terms of bit is $\gamma(p-1)$. This completes the proof. □

Remark 2 For $(n, k = n - r, pl)$ binary MDS array code with $(l-1) \mid (r-1)$ and $\frac{l(r-1)}{l-1} \nmid n$, the n storage node are divided into $g = \frac{l(r-1)}{(l-1)}$ groups where $n(\text{mod } g)$ groups are of size $\lceil n/g \rceil$ and the remaining groups are of size $\lfloor n/g \rfloor$, i.e, the code with $g \nmid n$ can be obtained by applying *shortening technique* on the code constructed through *Construction* 2. Also, the amount of data access is equal to the amount of data communicated during node repair.

In the following, we propose a scheme for finding the minimum value of p which guarantees the MDS property of the code constructed in *Construction* 2. Specifically, given an input prime p, the following Algorithm 1 outputs the minimum degree of the irreducible polynomials of $f_1(x), f_2(x), \ldots, f_t(x)$ without factoring $1 + x + x^2 + \cdots + x^{p-1}$, where $1 + x + x^2 + \cdots + x^{p-1} = f_1(x) f_2(x) \cdots f_t(x)$. By repeatedly using the algorithm and increasing the value of p every time, we can obtain the minimum value of p which satisfies the corresponding MDS condition in Theorem 3.

Algorithm 1 Find the degree of $f_1(x)$ for a prime p

```
Input: a prime $p$
Output: the degree of $f_1(x)$
  $\mathcal{S} \leftarrow \{1, 2, \ldots, p-1\}$
  $\mathcal{D} \leftarrow \emptyset$
  while $\mathcal{S} \neq \emptyset$ do
     $i \leftarrow \min(\mathcal{S})$                          ▷ $i$ is the minimum value in $\mathcal{S}$
     $d \leftarrow 1$
     $\mathcal{C} \leftarrow \emptyset$
     while $i2^d \neq i \pmod p$ do
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{i \times 2^{d-1} (\text{mod } p)\}$
        $d \leftarrow d + 1$
     end while
     $\mathcal{D} \leftarrow \mathcal{D} \cup \{|\mathcal{C}|\}$
     $\mathcal{S} \leftarrow \mathcal{S} \backslash \mathcal{C}$
  end while
  return $\min(\mathcal{D})$            ▷ The minimum value in $\mathcal{D}$ is the degree of $f_1(x)$
```

We now give an explanation of the algorithm. As $x^p - 1 = (x-1)(1 + x + x^2 + \cdots + x^{p-1}) = (x-1) f_1(x) f_2(x) \ldots f_t(x)$, the problem of finding the minimum degree of the irreducible polynomial factor of $1 + x + x^2 + \cdots + x^{p-1}$ translates into the problem of finding the minimum degree of the irreducible polynomial factor of $x^p - 1$ except $x - 1$. According to [16], the degree of $f_1(x)$ can be written as

$$\deg(f_1(x)) = \min(|\mathcal{C}_1|, |\mathcal{C}_2|, \ldots, |\mathcal{C}_t|) \tag{10}$$

where $\mathcal{C}_i, i \in [t]$ is the i-th cyclotomic coset and $|\mathcal{C}_i|$ denotes the cardinality of $\mathcal{C}_i$. Let $\mathbb{F}_{2^s}$ be the splitting field of $x^p - 1$ over $\mathbb{F}_2$, where s is the smallest positive integer for which $p \mid (2^s - 1)$. Let β be a primitive element of $\mathbb{F}_{2^s}$, from which we can determine a primitive p-th root of unity:

$$\omega = \beta^{\frac{2^s - 1}{p}}. \tag{11}$$

Thus, the roots of $x^p - 1 = 0$ over $\mathbb{F}_{2^s}$ can be written as $1, \omega, \omega^2, \cdots, \omega^{p-1}$. For some $i \in [p-1]$, the conjugates whose exponents constitute a cyclotomic coset are

$$\omega^i, \omega^{2i}, \omega^{4i}, \cdots, \omega^{2^{d-1} i} \tag{12}$$

where d is the smallest positive integer for which $\omega^{2^d i} = \omega^i$. From the fact that

$$\omega^{2^d i} = \omega^i \iff p \mid \left(i2^d - i\right) \iff i2^d = i \pmod{p}, \tag{13}$$

we can write the irreducible polynomial for the conjugates (12) as

$$m_i(x) = \left(x - \omega^i\right)\left(x - \omega^{2i}\right) \cdots \left(x - \omega^{i2^{d-1}}\right) \tag{14}$$

and the i-th cyclotomic coset as

$$\mathcal{C}_i = \left\{i, 2i, 2^2 i, \cdots, i2^{d-1}\right\} \tag{15}$$

where d is the smallest positive integer for which $i2^d = i \pmod{p}$. Obviously, $\mathcal{C}_i$ can be determined without the primitive p-th root of unity β, implying that we need not really factor the polynomial $x^p - 1$(or $1 + x + x^2 + \cdots + x^{p-1}$) to obtain the degree of $f_1(x)$.

4 Evaluation

The most related works to the present paper in the literature are [6, 9, 14, 28]. Comparisons between these related codes and the codes in the present paper are presented in the sequel.

In [14], the authors constructed MDS array codes of sub-packetization level $l = n - k = r$ (*Construction 13*) where the size of field satisfies that $\mathbb{F} \geq n^{(r-1)l+1}$. In the present paper, the codes constructed in Subsect. 3.2 (*Construction* 1) can be viewed as a generalization of the codes in [14] (*Construction 13*), where the sub-packetization level l satisfies that $(l-1) \mid (r-1)$. Thus, we make an improvement by providing a more flexible sub-packetization level.

Based on the same structure, the authors of [6] constructed binary MDS array code over the ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$ and showed the code's ability of repairing single-node failure with one or more busy nodes. For repairing single-node failure with fewer helper nodes, one can refer to [6] for details. When p is a prime number such that 2 is primitive in $\mathbb{F}_p$, the authors provided a method of obtaining the minimum value of p through a computer search program that could check whether each determinant was a multiple of $1 + x^p$ or not. In Subsect. 3.3 (*Construction* 2), we generalize the codes in [6] by constructing binary MDS codes with more flexible sub-packetization level over the same ring $\mathcal{R}$. The ability of repairing single-node failure with one or more busy nodes, although not presented in the paper, is straightforwardly obtained by our codes. Besides, we propose an algorithm to find the minimum value of p that guarantees the MDS property of the code, no matter whether 2 is primitive of $\mathbb{F}_p$ or not. Note that with the algorithm, one can easily find the minimum value of p without really factoring the polynomial $1 + x + x^2 + \cdots + x^{p-1}$.

In our previous work [9], we constructed MDS array code of sub-packetization level $l = 2$ where only the information nodes can be repaired with reduced bandwidth. In [28], the authors derived the lower bound of repair bandwidth for MDS array code with $l = r = 2$ and presented an explicit construction of the so called "DRF code" which achieved the lower bound. For fixed $l > 2$, the lower bound on repair bandwidth is still an open problem. In the present paper, we make a significate contribution by giving explicit constructions of MDS array codes with more flexible l (and r), where both the information and parity nodes can be repaired with reduced bandwidth.

5 Conclusion

In this paper, we propose two families of MDS array codes with small sub-packetization level of $\mathcal{O}(r)$. The first family of codes with $(n, k = n - r, l)$ are constructed over the finite field whose size is at least $n^{r(l-1)+1}$ and the sub-packetization level l satisfies that $(l - 1) \mid (r - 1)$. The second family of codes are binary MDS array codes constructed over a special polynomial ring $\mathcal{R} = \mathbb{F}_2[x]/(1 + x + x^2 + \cdots + x^{p-1})$ where p is a prime. Efficient repair procedure is explicitly presented for both of the codes. For the second family of codes, we give a sufficient condition for p under which the MDS property of the codes can be guaranteed. Moreover, we develop an explicit algorithm to find the minimum value of p where one need not really factor the polynomial $1 + x + x^2 + \cdots + x^{p-1}$.

Acknowledgements This work was supported in part by National Key R&D Program of China under Grant 2022YFA1005000, National Natural Science Foundation of China under Grant 62171279, and Fundings of SJTU-Alibaba Joint Research Lab on Cooperative Intelligent Computing.

Author Contributions All authors contributed to the study implementation and design. Research preparation, paper collection and analysis were performed by [Lei Li], [Xinchun Yu], [Chenhao Ying] and [Yuan Luo]. Disk I/O problems are analysed by [Liang Chen] and [Yuanyuan Dong]. [Xinchun Yu] and [Lei Li] worked together to propose Algorithm 1. The first draft of the manuscript was written by all authors and all authors commented on previous versions of the manuscript. All authors of the manuscript read and approved the final manuscript.

Funding This work was supported in part by National Key R&D Program of China under Grant 2022YFA1005000, National Natural Science Foundation of China under Grant 62171279, and Fundings of SJTU-Alibaba Joint Research Lab on Cooperative Intelligent Computing.

Data availability The data and materials generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Code availability The code generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

Ethical approval Not applicable.

Consent to participate All authors agreed with the content.

Consent for publication All authors agreed with the publication.

References

1. Cadambe V.R., Huang C., Li J.: Permutation code: optimal exact-repair of a single failed node in MDS code based distributed storage systems. In: 2011 IEEE International Symposium on Information Theory Proceedings, pp. 1225–1229 (2011).
2. Cadambe V.R., Jafar S.A., Maleki H., Ramchandran K., Suh C.: Asymptotic interference alignment for optimal repair of MDS codes in distributed storage. IEEE Trans. Inf. Theory **59**(5), 2974–2987 (2013).
3. Calder B., Wang J., Ogus A., Nilakantan N., Skjolsvold A., McKelvie S., Xu Y., Srivastav S., Wu J., Simitci H., et al.: Windows Azure Storage: a highly available cloud storage service with strong consistency. In: Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 143–157 (2011).

4. Dimakis A.G., Godfrey P.B., Wu Y., Wainwright M.J., Ramchandran K.: Network coding for distributed storage systems. IEEE Trans. Inf. Theory **56**(9), 4539–4551 (2010).
5. Ghemawat S., Gobioff H., Leung S.-T.: The google file system. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, pp. 29–43 (2003).
6. Hou H., Han Y.S., Bai B., Zhang G.: Towards efficient repair and coding of binary MDS array codes with small sub-packetization. In: 2022 IEEE International Symposium on Information Theory, pp. 3132–3137 (2022).
7. Hou H., Han Y.S., Lee P.P., Hu Y., Li H.: A new design of binary MDS array codes with asymptotically weak-optimal repair. IEEE Trans. Inf. Theory **65**(11), 7095–7113 (2019).
8. Li L., Ying C., Chen L., Dong Y., Luo Y.: New constructions of binary MDS array codes with optimal repair bandwidth. In: 2023 IEEE International Symposium on Information Theory, pp. 2045–2050 (2023).
9. Li L., Ying C., Yu X., Chen L., Dong Y., Luo Y.: Constructing MDS array codes with small repair bandwidth under sub-packetization two. In: 2023 15th International Conference on Wireless Communications and Signal Processing (2022).
10. Li J., Tang X., Tian C.: A generic transformation to enable optimal repair in MDS codes for distributed storage systems. IEEE Trans. Inf. Theory **64**(9), 6257–6267 (2018).
11. Liu Y., Li J., Tang X.: A generic transformation to enable optimal repair/access MDS array codes with multiple repair degrees. IEEE Trans. Inf. Theory (2023). https://doi.org/10.1109/TIT.2023.3248288.
12. Rashmi K.V., Shah N.B., Kumar P.V.: Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction. IEEE Trans. Inf. Theory **57**(8), 5227–5239 (2011).
13. Rawat A.S., Koyluoglu O.O., Vishwanath S.: Progress on high-rate MSR codes: enabling arbitrary number of helper nodes. In: 2016 Information Theory and Applications Workshop, pp. 1–6 (2016).
14. Rawat A.S., Tamo I., Guruswami V., Efremenko K.: MDS code constructions with small sub-packetization and near-optimal repair bandwidth. IEEE Trans. Inf. Theory **64**(10), 6506–6525 (2018).
15. Reed I.S., Solomon G.: Polynomial codes over certain finite fields. SIAM J. Appl. Math. **8**(2), 300–304 (1960).
16. Roman S.: Coding and Information Theory, vol. 134. Springer Science & Business Media, New York (1992).
17. Sasidharan B., Agarwal G.K., Kumar P.V.: A high-rate MSR code with polynomial sub-packetization level. In: 2015 IEEE International Symposium on Information Theory, pp. 2051–2055 (2015).
18. Sasidharan B., Vajha M., Kumar P.V.: An explicit, coupled-layer construction of a high-rate MSR code with low sub-packetization level, small field size and $d < (n-1)$. In: 2017 IEEE International Symposium on Information Theory, pp. 2048–2052 (2017).
19. Shan Y., Chen K., Gong T., Zhou L., Zhou T., Wu Y.: Geometric partitioning: explore the boundary of optimal erasure code repair. In: Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles, pp. 457–471 (2021).
20. Shvachko K., Kuang H., Radia S., Chansler R.: The Hadoop Distributed File System. In: 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies, pp. 1–10 (2010).
21. Suh C., Ramchandran K.: Exact-repair MDS codes for distributed storage using interference alignment. In: 2010 IEEE International Symposium on Information Theory, pp. 161–165 (2010).
22. Tamo I., Wang Z., Bruck J.: Zigzag codes: MDS array codes with optimal rebuilding. IEEE Trans. Inf. Theory **59**(3), 1597–1616 (2012).
23. Tamo I., Wang Z., Bruck J.: Access versus bandwidth in codes for storage. IEEE Trans. Inf. Theory **60**(4), 2028–2037 (2014).
24. Vajha M., Ramkumar V., Puranik B., Kini G., Lobo E., Sasidharan B., Kumar P.V., Barg A., Ye M., Narayanamurthy S., et al.: Clay codes: moulding MDS codes to yield an MSR code. In: 16th USENIX Conference on File and Storage Technologies, pp. 139–154 (2018).
25. Vajha M., Balaji S., Kumar P.V.: Small-d MSR codes with optimal access, optimal sub-packetization and linear field size. IEEE Trans. Inf. Theory (2023). https://doi.org/10.1109/TIT.2023.3250458.
26. Wang N., Li G., Hu S., Ye M.: Constructing MSR codes with subpacketization 2 n/3 for k+ 1 helper nodes. IEEE Trans. Inf. Theory (2023). https://doi.org/10.1109/TIT.2023.3238759.
27. Wu Y., Dimakis A.G.: Reducing repair traffic for erasure coding-based storage via interference alignment. In: 2009 IEEE International Symposium on Information Theory, pp. 2276–2280 (2009). IEEE.
28. Wu T.-Y., Han Y.S., Li Z., Bai B., Zhang G., Zhang X., Wu X.: Achievable lower bound on the optimal access bandwidth of (k+ 2, k, 2)-MDS array code with degraded read friendly. In: 2021 IEEE Information Theory Workshop, pp. 1–5 (2021).
29. Wu Y.: Existence and construction of capacity-achieving network codes for distributed storage. IEEE J. Sel. Areas Commun. **28**(2), 277–288 (2010).

30. Ye M., Barg A.: Explicit constructions of high-rate MDS array codes with optimal repair bandwidth. IEEE Trans. Inf. Theory **63**(4), 2001–2014 (2017).
31. Ye M., Barg A.: Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization. IEEE Trans. Inf. Theory **63**(10), 6307–6317 (2017).