Update Bandwidth for Distributed Storage

Zhengrui Li, Sian-Jheng Lin, *Member, IEEE,* Po-Ning Chen, *Senior Member, IEEE,* Yunghsiang S. Han, *Fellow, IEEE,* and Hanxu Hou, *Member, IEEE*

***Abstract*—In this paper, we consider the update bandwidth in distributed storage systems (DSSs). The update bandwidth, which measures the transmission efficiency of the update process in DSSs, is defined as the average amount of data symbols transferred in the network when the data symbols stored in a node are updated. This paper contains the following contributions. First, we establish the closed-form expression of the minimum update bandwidth attainable by irregular array codes. Second, after defining a class of irregular array codes, called Minimum Update Bandwidth (MUB) codes, which achieve the minimum update bandwidth of irregular array codes, we determine the smallest code redundancy attainable by MUB codes. Third, the code parameters, with which the minimum code redundancy of irregular array codes and the smallest code redundancy of MUB codes can be equal, are identified, which allows us to define MR-MUB codes as a class of irregular array codes that simultaneously achieve the minimum code redundancy and the minimum update bandwidth. Fourth, we introduce explicit code constructions of MR-MUB codes and MUB codes with the smallest code redundancy. Fifth, we establish a lower bound of the update complexity of MR-MUB codes, which can be used to prove that the minimum update complexity of irregular array codes may not be achieved by MR-MUB codes. Last, we construct a class of $(n = k + 2, k)$ vertical maximum-distance separable (MDS) array codes that can achieve all of the minimum code redundancy, the minimum update bandwidth and the optimal repair bandwidth of irregular array codes.**

I. Introduction

Some distributed storage systems (DSSs) adopt replication policy to improve reliability. However, the replication policy requires a high level of storage overhead. To reduce this overhead while maintaining reliability, the erasure coding has been used in DSSs, such as Google File System [1] and Microsoft Azure Storage [2]. A main issue of erasure codes in DSSs is the required bandwidth to repair failure node(s). To tackle this issue, many linear block codes, such as regenerating codes [3], [4] and locally repairable codes (LRCs) [5], [6], were proposed in recent years. When the original data symbols change, the coded symbols stored in a DSS must be updated accordingly. Since performing updates consumes both bandwidth and energy, a better update efficiency is favorable for erasure codes in scenarios where updates are frequent. Different update scenarios have been studied [7]–[11]. When updating a stale node that has missed an update message from those nodes that has the message, [7] considered the minimization of the communications cost, of which tight bounds were developed. As a follow-up, [8] extended the setting to the regenerating-codes-based DSSs. A similar scenario was considered in [9], where the update scheme with the minimum communication cost for arbitrary linear functions had been investigated. Unlike the previous works, [10] considered the scenario that different versions of data files might coexist in a storage system after many update activities and exploited the differences across updates to reduce the I/O access. Recently, in an asynchronous update scenario, [11] studied the necessary storage cost to ensure data consistency. In this paper, a scenario different from all mentioned above is considered. Assuming no communication miss and no mismatch in data file versions, we focus on the typical update process that a node sends update information to other nodes for the maintenance of parity symbols among nodes.

This update process has two important phases, which are symbol transmission among nodes and symbol updating (i.e., reading-out and writing-in) in each node. Thus, the update efficiency should include the transmission efficiency and the I/O efficiency. Two definitions of I/O efficiency under the name of *update complexities* have been introduced in the literature [12]–[25]. By defining the *update complexity* as the *average* number of coded symbols (i.e., parity symbols) that must be updated when any single data symbol is changed, [12]–[14] studied the I/O efficiency of the update process of MDS array codes in the context of RAID storage systems. By following this definition, MDS array codes with the minimum update complexity were later constructed in [15], [16]. Then, the same definition of update complexity was adopted in the investigation of the update performance of regenerating codes [17], [18] and LRCs [19], [20]. In contrast, [21] defined the *update complexity* as the *maximum* number of coded symbols updated when a data symbol is altered. As a result, in [21], a code is said to be update-efficient if its maximum update complexity is sub-linear with respect to the code length. Based on this definition, many update-efficient schemes were subsequently introduced in [22]–[25]. In this work, we consider the *average* update complexity rather than the *maximal* one. There are two considerations for our selection. First, our paper focuses on MDS array codes, which are basically designed to tolerate a linear number of erasures; hence, a sub-linear maximal update complexity may not be a suitable criterion for their I/O efficiency. Second, the

This work was partially presented at the 2019 IEEE International Symposium on Information Theory.

This work was partially funded by the National Key R&D Program of China (No. 2020YFA0712300), and the National Natural Science Foundation of China (No. 62071446, 62071121)

Z. Li and S.-J. Lin are with the School of Information Science and Technology, University of Science and Technology of China (USTC), Hefei, Anhui, China, email: dd622089@mail.ustc.edu.cn, sjlin@ustc.edu.cn.

P.-N. Chen is with the Department of Electrical and Computer Engineering, National Chiao Tung University, email:poning@faculty.nctu.edu.tw.

Y. S. Han is with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China and College of Electrical Engineering and Computer Science, National Taipei University, email: yunghsiangh@gmail.com.

H. Hou is with the School of Electrical Engineering & Intelligentization, Dongguan University of Technology, email: houhanxu@163.com.

average update complexity can characterize the long-term I/O efficiency of an update process, which is a main concern of most storage systems. For simplicity, as in [12]–[20], we will use *update complexity* to refer to *average update complexity* in the rest of the paper.

One may regard the transmission efficiency as a function of the I/O efficiency and treat the latter as a key indicator for an update process's efficiency. A supporting observation is that when updating a single data symbol, as the number of symbols transmitted between nodes is at most one, the fewer nodes affected by an update of a symbol, the better the transmission efficiency. However, when updating many or even all symbols in a node simultaneously, the two efficiencies are no longer in perfect agreement, and an update process that minimizes the transmission efficiency may not achieve the minimum I/O efficiency (cf. Theorem 10). To our best knowledge, there is no work discussing the transmission efficiency in the update process of a DSS. Note that [7]–[9] also studied the transmission efficiency, however, of a different updating scenario. They considered a setting that an offline node comes back and has to download information from other nodes without the knowledge of which data symbols have been modified during the offline period and what their original values are. Instead, we consider the transmission cost that after a node is updated (as a usual frequent activity in a storage system), it must send update information to other nodes to maintain the integrity of the parity symbols among nodes.

In this paper, we introduce a new metric, called the *update bandwidth*, to measure the transmission efficiency in the update process of erasure codes applied in DSSs. It is defined as the average amount of symbols that must be transmitted among nodes when the data symbols stored in a node are updated. As the storage capacity of a node is very large nowadays, an erasure-codes-based DSS may contain a large amount of codewords, each of which, structured as an irregular array, places a column of symbols at a node. Let the symbols in a node corresponding to the same codeword be a coded block. As such, a node divides its storage capacity into several blocks, and each block stores all symbols from a column of a specific codeword. Since every codeword follows its own encoding and decoding operations and there is no exchange of updating information between codewords, without loss of generality, we could adopt the simplest setting that the DSS system only contains a single codeword in our analysis of updating bandwidth. For simplicity, we consider the updating operation of a block as a whole in this work.[1] Thus, when any data symbol in a block is required to be updated, all data symbols in the block are involved in this single updating operation.

The update model that we consider is described as follows.

[1]Although it is possible to consider the update bandwidth that allows a partial update of symbols within a block, exchanging the information of which symbols in a block are updated and which not with all the other nodes requires extra maintenance bandwidth. Since the number of cases corresponding to which symbols in a block are updated grows exponentially fast as the block size increases, the overhead to handle partial updating may increase considerably. Therefore, a more delicate tradeoff between the extra maintenance bandwidth/overhead and the save of the update bandwidth is necessary and is deferred as a future work.

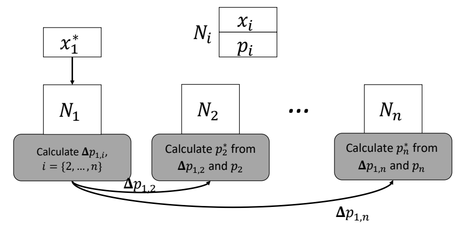

Fig. 1. An instance of the considered update model, where the data symbols stored in N_1 are updated, and N_i has the data vector $\mathbf{x}_i$ and the parity vector $\mathbf{p}_i$. When updating the data symbols stored in N_1, N_1 sends intermediate symbols $\{\Delta\mathbf{p}_{1,i}\}_{i=2,\ldots,n}$ respectively to all other nodes such that they can calculate the new parity vectors.

Assume that there are n nodes $\{N_i\}_{i=1}^n$ in the network. Node N_i stores data vector $\mathbf{x}_i$ and parity vector $\mathbf{p}_i$, where the former consists of data symbols, while the parity symbols are placed in the latter. Fig. 1 demonstrates the update procedure when the data vector $\mathbf{x}_1$ is updated to $\mathbf{x}_1^*$. In the update procedure, N_1 first calculates $n-1$ intermediate vectors $\{\Delta\mathbf{p}_{1,i}\}_{i=2}^n$, and then sends $\Delta\mathbf{p}_{1,i}$ to N_i respectively for $i = 2, 3, \ldots, n$. After receiving $\Delta\mathbf{p}_{1,i}$, N_i computes the updated parity vector $\mathbf{p}_i^*$ from $\Delta\mathbf{p}_{1,i}$ and the old parity vector $\mathbf{p}_i$. This completes the update procedure. Notably, this update model is similar to the one adopted in [26], in which partial-updating schemes for erasure-coded storage are considered. Our general update model will be given formally in Section II-C.

It is worthy mentioning that the codes with the minimum update complexity (i.e., I/O efficiency) may not achieve the minimum update bandwidth, and vice versa. To show that, Fig. 2 presents two $(n = 4, k = 2)$ maximum distance separable (MDS) array codes, where the elements in the i-th column are the symbols stored in node N_i and the number of symbols in each node is $\alpha = 4$. In Fig. 2(a), the first row and the third row form an instance of a 2×4 P-code [16], and the second row and the fourth row form another instance of a 2×4 P-code. Thus, Fig. 2(a) is an instance of a 4×4 P-code. Furthermore, Fig. 2(b) is an instance of our codes proposed in Section V. In Figs. 2(a) and 2(b), the data symbols $\{x_{i,j}\}_{i=1,\ldots,4,j=1,2}$ are arranged in the first two gray rows, and the last two rows are occupied by parity symbols. It can be verified that the data symbols can be recovered by accessing any two columns of the codes in Figs. 2(a) and 2(b), and hence $k = 2$. It is known that P-codes [16] achieve the minimum update complexity when $n - k = 2$. Hence, when updating a data symbol in Fig. 2(a), we must update at least $n-k = 2$ parity symbols. For example, when updating $x_{1,1}$, the third symbol in the second column $x_{1,1}+x_{3,1}$ and the third symbol in the fourth column $x_{1,1}+x_{2,1}$ need to be updated. However, in Fig. 2(b), when updating a data symbol, two or three parity symbols need to be updated, i.e., the corresponding update complexity is larger than 2. For example, when updating $x_{1,1}$, the third symbol in the second column $x_{1,1}+x_{4,2}$ and the fourth symbol in the fourth column $x_{1,1}+x_{1,2}+x_{2,2}$ need to be updated. Yet, the updating of $x_{1,2}$ requires the modification of both parity symbols in the third column (i.e, $x_{2,1}+x_{1,2}$ and $x_{4,1}+x_{4,2}+x_{1,2}$) and the fourth symbol in the fourth column (i.e., $x_{1,1}+x_{1,2}+x_{2,2}$).

Next, we consider the update bandwidth. Suppose that the

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2021.3105439

$x_{1,1}$	$x_{2,1}$	$x_{3,1}$	$x_{4,1}$
$x_{1,2}$	$x_{2,2}$	$x_{3,2}$	$x_{4,2}$
$x_{3,1}+x_{4,1}$	$x_{1,1}+x_{3,1}$	$x_{2,1}+x_{4,1}$	$x_{1,1}+x_{2,1}$
$x_{3,2}+x_{4,2}$	$x_{1,2}+x_{3,2}$	$x_{2,2}+x_{4,2}$	$x_{1,2}+x_{2,2}$

(a)

$x_{1,1}$	$\underline{x_{2,1}}$	$x_{3,1}$	$\underline{x_{4,1}}$
$x_{1,2}$	$\underline{x_{2,2}}$	$\underline{x_{3,2}}$	$x_{4,2}$
$x_{4,1}+x_{3,2}$	$x_{1,1}+x_{4,2}$	$\underline{x_{2,1}+x_{1,2}}$	$x_{3,1}+x_{2,2}$
$x_{2,1}+x_{2,2}+x_{3,2}$	$x_{3,1}+x_{3,2}+x_{4,2}$	$x_{4,1}+x_{4,2}+x_{1,2}$	$\underline{x_{1,1}+x_{1,2}+x_{2,2}}$

(b)

Fig. 2. (a) presents an instance, where gray rows contain the data symbols, of a 4×4 P-code with optimal update complexity 2 and update bandwidth 4; (b) presents an instance, where gray rows contain the data symbols, of a proposed $(n = 4, k = 2)$ code, which has update complexity larger than 2 and minimum update bandwidth 3. Note that the instance presented in (b) also has the optimal repair bandwidth.

two data symbols in the first node in Fig. 2(a) are updated, i.e., $x_{1,j}$ are updated to $x^*_{1,j}$, $j = 1, 2$. The first node should send two symbols $\Delta x_{1,1}$ and $\Delta x_{1,2}$ to both nodes 2 and 4, where $\Delta x_{i,j} = x^*_{i,j} - x_{i,j}$. Thus, the required bandwidth is four. It is easy to check that the required bandwidth of updating two data symbols of any other node is also four. Therefore, the update bandwidth of the 4×4 P-code is four. Next we show that the update bandwidth of the code in Fig. 2(b) is three. When two data symbols in node 1 in Fig. 2(b) are updated, we only need to send $\Delta x_{1,1}$ to node 2, $\Delta x_{1,2}$ to node 3, and $(\Delta x_{1,1} + \Delta x_{1,2})$ to node 4. Therefore, the required update bandwidth is three. We can verify that the required update bandwidth when updating any other node in Fig. 2(b) is also three. Consequently, the update bandwidth of the code in Fig. 2(b) is better than that of the 4×4 P-code in Fig. 2(a). We will show in Section IV that the code in Fig. 2(b) achieves the minimum update bandwidth among all $(4, 2)$ irregular array codes with two data symbols per node.

Other than update complexity and update bandwidth, the *repair bandwidth*, defined as the amount of symbols downloaded from the surviving nodes to repair the failed node, is also an important consideration in DSSs. The repair problem was first brought into the spotlight by Dimakis et al. [3]. It can be anticipated that a well-designed code with both minimum update bandwidth and optimal repair bandwidth is attractive for DSSs. Surprisingly, the code in Fig. 2(b) also achieves the optimal repair bandwidth among all $(4, 2)$ MDS array codes. One can check that we can repair the four symbols stored in node 1 by downloading the six underlined symbols in Fig. 2(b), i.e., $\{x_{2,1}, x_{2,2}\}$ from node 2, $\{x_{3,2}, x_{2,1} + x_{1,2}\}$ from node 3, and $\{x_{4,1}, x_{1,1} + x_{1,2} + x_{2,2}\}$ from node 4. Thus, the repair bandwidth of node 1 is six, which is optimal for the parameters of $n = 4$, $k = 2$ and $\alpha = 4$ [3]. We can verify that the repair bandwidth of any other node in Fig. 2(b) is also six. Therefore, the repair bandwidth of the code in Fig. 2(b) is optimal.

The contributions of this paper are as follows.

- We introduce a new metric, i.e., update bandwidth, and emphasize its importance in scenarios where storage updates are frequent.
- We consider irregular array codes with a given level of protection against block erasures [27], and establish the closed-form expression of the minimum update bandwidth attainable for such codes.
- Referring to the class of irregular array codes that achieve the minimum update bandwidth as MUB codes, we next derive the smallest code redundancy attainable by MUB codes.
- Comparing the smallest code redundancy of MUB codes with the minimum code redundancy of irregular array codes derived in [27], we identify a class of MUB codes, called MR-MUB codes, that can achieve simultaneously the minimum code redundancy of irregular array codes and the minimum update bandwidth of irregular array codes.
- Systematic code constructions for MR-MUB codes and for MUB codes with the smallest code redundancy are both provided.
- We establish a lower bound of the update complexity of MR-MUB codes, by which we confirm that the update complexity of irregular array codes may not be achieved by MR-MUB codes.
- We construct an $(n, k = n - 2)$ MR-MUB code with the optimal repair bandwidth for all nodes via the transformation in [28], confirming the existence of the irregular array codes that can simultaneously achieve the minimum code redundancy, the minimum update bandwidth and the optimal repair bandwidth for all codes.

The rest of this paper is organized as follows. Section II introduces the notations used in this paper and the proposed update model. Section III establishes the necessary condition for the existence of an irregular array code. In Section IV, via the form of integer programs, we determine the minimum update bandwidth of irregular array codes and the smallest code redundancy of MUB codes. Section V presents the explicit constructions of MR-MUB codes and MUB codes. Section VI derives a lower bound of the update complexity of MR-MUB codes. Section VII devises a class of $(n = k+2, k)$ MR-MUB codes with the optimal repair bandwidth for all nodes. Section VIII concludes this work.

II. Preliminary

A. Definition

We first introduce the notations used in this paper. Let $[n] \triangleq \{1, \ldots, n\}$ for a positive integer n, and by convention, set $[0] \triangleq \emptyset$. $(a_i)_{i\in[n]}$ denotes an index set $(a_1, a_2, \ldots, a_n)$. Let $[x_{i,j}]_{i\in[m],j\in[n]}$ denote an $m \times n$ matrix whose entry in row i and column j is $x_{i,j}$. $\mathsf{wt}(\mathbf{v})$ denotes the weight of vector $\mathbf{v}$, i.e., the number of nonzero elements in vector $\mathbf{v}$. $\mathbf{M}^\mathsf{T}$ represents the transpose of matrix $\mathbf{M}$. $\mathsf{row}(\mathbf{M})$, $\mathsf{col}(\mathbf{M})$ and $\mathsf{rank}(\mathbf{M})$ represent the number of rows of $\mathbf{M}$, the number of columns of $\mathbf{M}$ and the rank of $\mathbf{M}$, respectively. $\mathbf{M}^{-1}$ denotes the inverse matrix of $\mathbf{M}$, provided $\mathbf{M}$ is invertible. $|S|$ denotes the cardinality of a set S. $\mathbb{F}_q$ denotes the finite field of size q, where q is a power of a prime. For two discrete random variables X and Y, their joint probability distribution is denoted as $P_{XY}(x, y)$. $H_q(X)$ denotes the q-ary entropy of X, and $I_q(X; Y)$ denotes the q-ary mutual information between X and Y, where q is the base of the logarithm. We consider linear codes throughout the paper and the main notations used in this paper are listed in Table I.

B. Irregular array code

As its name reveals, an irregular array code [27], [29] consists of codewords that are irregular arrays, where the size of each column can be arbitrary as shown in Fig. 3. Formally, given a positive integer n and two column vectors $\mathbf{m} = [m_1 \ \ldots \ m_n]^\mathsf{T}$ and $\mathbf{p} = [p_1 \ \ldots \ p_n]^\mathsf{T}$, where $m_i, p_i \geq 0$ for $i \in [n]$, the codeword of an irregular array code $\mathcal{C}$ over $\mathbb{F}_q$ is denoted as

$$\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n), \tag{1}$$

where the column vector $\mathbf{c}_i$ contains m_i data symbols and p_i parity symbols. Specifically, we denote

$$\mathbf{c}_i = \begin{bmatrix} \mathbf{x}_i \\ \mathbf{p}_i \end{bmatrix}, \quad i \in [n], \tag{2}$$

where $\mathbf{x}_i$ is the i-th data vector that contains m_i data symbols and $\mathbf{p}_i$ is the i-th parity vector that contains p_i parity symbols. Since $\mathbf{x}_i$ contains data symbols, we can naturally consider that $\mathbf{x}_i$ is uniformly distributed over $\mathbb{F}_q^{m_i}$, and $\mathbf{x}_i$ and $\mathbf{x}_j$ are independent for $i \neq j \in [n]$. As such, we have

$$H_q(\mathbf{x}_i) = m_i \qquad \forall i \in [n], \tag{3}$$

$$I_q(\mathbf{x}_i; \mathbf{x}_j) = 0 \qquad \forall i, j \in [n], i \neq j. \tag{4}$$

As all symbols in $\mathbf{p}_i \in \mathbb{F}_q^{p_i}$ may not be independent, we can only obtain

$$H_q(\mathbf{p}_i) \leq p_i \qquad \forall i \in [n]. \tag{5}$$

The storage redundancy (i.e., code redundancy) of $\mathcal{C}$ is the total number of parity symbols, i.e., $R = \sum_{i=1}^n p_i$. An example of irregular array codes is illustrated in Fig. 3, where the gray cells contain the data symbols. In this example, we have $\mathbf{m} = [4 \ 2 \ 2 \ 0]^\mathsf{T}$, $\mathbf{p} = [2 \ 3 \ 3 \ 3]^\mathsf{T}$ and $R = 11$. In addition, the first column of the irregular array code in Fig. 3 stores four data symbols $\mathbf{x}_1 = [x_{1,1} \ x_{1,2} \ x_{1,3} \ x_{1,4}]^\mathsf{T}$ and two parity symbols $\mathbf{p}_1 = [x_{2,1} + x_{2,2} \ x_{3,2}]^\mathsf{T}$.

$x_{1,1}$	$x_{2,1}$	$x_{3,1}$	$x_{1,1}+x_{1,3}+x_{3,1}$
$x_{1,2}$	$x_{2,2}$	$x_{3,2}$	$x_{1,2}+x_{1,4}+x_{3,1}$
$x_{1,3}$	$x_{1,1}$	$x_{1,3}$	$x_{2,2}+x_{3,1}$
$x_{1,4}$	$x_{1,2}$	$x_{1,4}$	
$x_{2,1}+x_{2,2}$	$x_{3,1}+x_{3,2}$	$x_{2,1}$	
$x_{3,2}$			

Fig. 3. This figure, where the gray cells contain the data symbols, shows a $(4, 2, \mathbf{m})$ irregular MDS array code with $\mathbf{m} = [4\ 2\ 2\ 0]^\mathsf{T}$ and $\mathbf{p} = [2\ 3\ 3\ 3]^\mathsf{T}$.

When $m_i + p_i = m_j + p_j$ for any $i \neq j \in [n]$, the irregular array code $\mathcal{C}$ is reduced to a regular array code. When $m_i = m_j$ and $p_i = p_j$ for all $i \neq j \in [n]$, $\mathcal{C}$ is called a vertical array code. As an example, both codes in Fig. 2 are vertical array codes. When a regular array code satisfies $p_i = 0$ for $i \in [k]$ and $m_j = 0$ for $k < j \leq n$, it is called a horizontal array code. If we can retrieve all the data symbols by accessing any k columns, and there is a set of $k-1$ columns which we can not retrieve all the data symbols from, then the code $\mathcal{C}$ is parameterized as an $(n, k, \mathbf{m})$ irregular array code.

We will demonstrate in Section V-C that the code in Fig. 3 can only be reconstructed by accessing at least any two columns, and hence it is a $(4, 2, \mathbf{m})$ irregular array code. In fact, the code in Fig. 3 is also an MUB code with the smallest code redundancy (cf. Section IV-C). In the subsection that follows, we will introduce the update model and the update bandwidth of $(n, k, \mathbf{m})$ irregular array codes.

C. Update model and update bandwidth

In an $(n, k, \mathbf{m})$ irregular array code $\mathcal{C}$, each parity symbol can be generated as a linear combination of all data symbols. Thus, the parity symbols in each column can be obtained from

$$\mathbf{p}_j = \sum_{i=1}^n \mathbf{M}_{i,j}\mathbf{x}_i \quad j \in [n], \tag{6}$$

where $\mathbf{M}_{i,j}$ is a $p_j \times m_i$ matrix, called construction matrix. Apparently, when the data vector in node i is updated from $\mathbf{x}_i$ to $\mathbf{x}_i^*$, node j with $j \in [n]\backslash\{i\}$ needs to update its parity vector via $\mathbf{p}_j^* = \mathbf{p}_j + \mathbf{M}_{i,j}\Delta\mathbf{x}_i$, where $\Delta\mathbf{x}_i = \mathbf{x}_i^* - \mathbf{x}_i$. Such an update process can be divided into two steps, which are performed by node i and node j, respectively. In the first step, with a pre-specified linear transformation $\mathbf{A}_{i,j}$, node i calculates the intermediate vector $\mathbf{A}_{i,j}\Delta\mathbf{x}_i$, and sends the symbols in this vector to node j. In the second step, node j also pre-specifies a linear transformation $\mathbf{B}_{i,j}$ and calculates $\Delta\mathbf{p}_j = \mathbf{p}_j^* - \mathbf{p}_j$ from the intermediate vector $\mathbf{A}_{i,j}\Delta\mathbf{x}_i$ just received via $\Delta\mathbf{p}_j = \mathbf{B}_{i,j}\mathbf{A}_{i,j}\Delta\mathbf{x}_i$. As a result of (6), the two matrices $\mathbf{A}_{i,j}$ and $\mathbf{B}_{i,j}$ corresponding to the linear transformations respectively performed by node i and node j must satisfy $\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}$. Based on the above update model, the number of symbols sent from node i to node j is $\mathsf{row}(\mathbf{A}_{i,j})$.

Denote

$$\gamma_{i,j} \triangleq \min_{\mathbf{A}_{i,j},\mathbf{B}_{i,j}} \{\mathsf{row}(\mathbf{A}_{i,j}) | \mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}\}, \text{ for } i \neq j. \tag{7}$$

as the minimum amount of symbols sent from node i to node j when updating the data symbols stored in node i. The

TABLE I
MAIN NOTATIONS USED IN THIS PAPER

Notation	Description
n	The number of nodes
m_i	The number of data symbols in node i
p_i	The number of parity symbols in node i
$\mathbf{m}$	$\mathbf{m} = [m_1 \dots m_n]^\mathsf{T}$
$\mathbf{p}$	$\mathbf{p} = [p_1 \dots p_n]^\mathsf{T}$
$\mathbf{x}_i$	The i-th data vector
$\mathbf{p}_i$	The i-th parity vector
$\mathbf{c}_i$	$\mathbf{c}_i = [\mathbf{x}_i^\mathsf{T} \quad \mathbf{p}_i^\mathsf{T}]^\mathsf{T}$, the i-th column vector
$\mathbf{C}$	$\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n)$, the codeword of an irregular array code
$\mathbf{M}_{i,j}$	The construction matrix
$\mathbf{A}_{i,j}$, $\mathbf{B}_{i,j}$	A decomposition of $\mathbf{M}_{i,j}$, i.e., $\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}$
$\mathcal{E}$	A subset of $[n]$ with $\lvert\mathcal{E}\rvert = n-k$, where the elements in it are denoted as e_i with $i \in [n-k]$
$\bar{\mathcal{E}}$	$\bar{\mathcal{E}} = [n] \setminus \mathcal{E}$, where the elements in it are denoted as $\bar{e}_i$ with $i \in [k]$
$\mathbf{C}_\mathcal{E}$, $\mathbf{X}_\mathcal{E}$, $\mathbf{P}_\mathcal{E}$	$\mathbf{C}_\mathcal{E} = [\mathbf{c}_{e_1}^\mathsf{T} \dots \mathbf{c}_{e_{n-k}}^\mathsf{T}]^\mathsf{T}$, $\mathbf{X}_\mathcal{E} = [\mathbf{x}_{e_1}^\mathsf{T} \dots \mathbf{x}_{e_{n-k}}^\mathsf{T}]^\mathsf{T}$, $\mathbf{P}_\mathcal{E} = [\mathbf{p}_{e_1}^\mathsf{T} \dots \mathbf{p}_{e_{n-k}}^\mathsf{T}]^\mathsf{T}$
B	The number of data symbols
R	The number of parity symbols, i.e., code redundancy
$\gamma_{i,j}$	The minimum number of symbols sent from node i to node j when updating the data symbols in node i
γ	The average required bandwidth when updating a node, i.e., update bandwidth
$\gamma_{\min}$	The minimum update bandwidth among all irregular array codes
$R_{\min}$	The minimum code redundancy among all irregular array codes
R_{sma}	The smallest code redundancy for irregular array codes with update bandwidth equal to $\gamma_{\min}$
θ	The average number of parity symbols affected by a change of a single data symbol, i.e., update complexity

following theorem then shows that in order to achieve $\gamma_{i,j}$, $\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}$ must correspond to a full rank decomposition of $\mathbf{M}_{i,j}$.

Theorem 1. $\mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j}$ *if, and only if,* $\mathsf{rank}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{B}_{i,j}) = \mathsf{row}(\mathbf{A}_{i,j})$*, and both* $\mathbf{A}_{i,j}$ *and* $\mathbf{B}_{i,j}$ *are full rank. Furthermore,* $\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j})$.

Proof. For better readability, the proof is relegated to Appendix A. □

Theorem 1 indicates that $\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j})$ is the minimum amount of symbols required to be sent from node i to node j when updating the data symbols stored in node i. We thus define the update bandwidth γ for a code $\mathcal{C}$ as the average required bandwidth. Formally,

Definition 1. *Given an* $(n, k, \mathbf{m})$ *irregular array code* $\mathcal{C}$ *with construction matrices* $\{\mathbf{M}_{i,j}\}_{i,j\in[n]}$*, the update bandwidth* γ *of code* $\mathcal{C}$ *is defined as*

$$\gamma \triangleq \frac{1}{n}\sum_{i=1}^{n}\sum_{j\in[n]\setminus\{i\}} \gamma_{i,j}, \tag{8}$$

where $\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j})$.

By Theorem 1, the update bandwidth γ can be achieved by adopting two full-rank matrices that fulfill $\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}$, where $\mathbf{B}_{i,j}$ is a $p_j \times \gamma_{i,j}$ matrix and $\mathbf{A}_{i,j}$ is a $\gamma_{i,j} \times m_i$ matrix. In the rest of the paper, the full-rank matrices $\mathbf{B}_{i,j}$ and $\mathbf{A}_{i,j}$ used in our update model are fixed as the ones with rank $\gamma_{i,j}$.

D. Encoding aspect of the update model

The update model in the previous subsection can also be equivalently characterized via an encoding aspect from (7). Specifically, we can first calculate

$$\mathbf{p}_{i,j} = \mathbf{A}_{i,j}\mathbf{x}_i \qquad \forall i, j \in [n], i \neq j. \tag{9}$$

Similar to (5), since symbols in $\mathbf{p}_{i,j} \in \mathbb{F}_q^{\gamma_{i,j}}$ are possibly dependent, we can only obtain

$$H_q(\mathbf{p}_{i,j}) \leq \gamma_{i,j} \qquad \forall i, j \in [n], i \neq j. \tag{10}$$

Then, as $\mathbf{p}_j = \sum_{i=1}^{n} \mathbf{M}_{i,j}\mathbf{x}_i$ for any $j \in [n]$ (cf. (6)), we obtain

$$\mathbf{p}_j = \sum_{i=1, i\neq j}^{n} \mathbf{B}_{i,j}\mathbf{p}_{i,j} \qquad \forall j \in [n]. \tag{11}$$

As a result, the parity symbols are the coded symbols from two sets of encoding matrices $\{\mathbf{A}_{i,j}\}_{i,j\in[n]}$ and $\{\mathbf{B}_{i,j}\}_{i,j\in[n]}$. This encoding aspect of the update model will be adopted in later sections. Since the number of symbols passed from (9) to (11) is $\sum_{i=1}^{n}\sum_{j\in[n]\setminus\{i\}} \gamma_{i,j} = n\gamma$, the average number of symbols transmitted among all nodes during the encoding process is equal to the update bandwidth γ.

III. NECESSARY CONDITION FOR THE EXISTENCE OF AN IRREGULAR ARRAY CODE

In this section, we provide a necessary condition for the parameters $\{p_j\}_{j\in[n]}$ and $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ such that an $(n, k, \mathbf{m})$ irregular array code $\mathcal{C}$, where retrieval of data symbols can only be guaranteed by any other k columns but not by any other $k-1$ columns, exists (cf. Theorem 2 and Corollary 1). For simplicity, we use $H(\cdot)$ and $I(\cdot\,;\,\cdot)$ to represent $H_q(\cdot)$ and $I_q(\cdot\,;\,\cdot)$ in this section.

Some notations used in the proofs below are first introduced (cf. Table I). For a subset $\mathcal{E} \subset [n]$ with $\lvert\mathcal{E}\rvert = n-k$, the elements in $\mathcal{E}$ are denoted as e_i with $i \in [n-k]$. Similarly, denote the elements in $\bar{\mathcal{E}} \triangleq [n] \setminus \mathcal{E}$ as $\bar{e}_i$ with $i \in [k]$. Let $\mathbf{X}_\mathcal{E} \triangleq [\mathbf{x}_{e_1}^\mathsf{T} \dots \mathbf{x}_{e_{n-k}}^\mathsf{T}]^\mathsf{T}$ and $\mathbf{X}_{\bar{\mathcal{E}}} \triangleq [\mathbf{x}_{\bar{e}_1}^\mathsf{T} \dots \mathbf{x}_{\bar{e}_k}^\mathsf{T}]^\mathsf{T}$, and $\mathbf{C}_\mathcal{E}$,

$\mathbf{C}_{\bar{\mathcal{E}}}$, $\mathbf{P}_{\mathcal{E}}$ and $\mathbf{P}_{\bar{\mathcal{E}}}$ are similarly defined. Equation (6) can then be rewritten using these notations as

$$\mathbf{p}_j = \sum_{i\in[n-k]} \mathbf{M}_{e_i,j}\mathbf{x}_{e_i} + \sum_{i\in[k]} \mathbf{M}_{\bar{e}_i,j}\mathbf{x}_{\bar{e}_i}. \tag{12}$$

Thus, we can write

$$\mathbf{P}_{\bar{\mathcal{E}}} = \begin{bmatrix}\mathbf{p}_{\bar{e}_1}\\ \mathbf{p}_{\bar{e}_2}\\ \vdots \\ \mathbf{p}_{\bar{e}_k}\end{bmatrix} = \begin{bmatrix}\mathbf{M}_{e_1,\bar{e}_1} & \dots & \mathbf{M}_{e_{n-k},\bar{e}_1}\\ \mathbf{M}_{e_1,\bar{e}_2} & \dots & \mathbf{M}_{e_{n-k},\bar{e}_2}\\ \vdots & \ddots & \vdots\\ \mathbf{M}_{e_1,\bar{e}_k} & \dots & \mathbf{M}_{e_{n-k},\bar{e}_k}\end{bmatrix}\mathbf{X}_{\mathcal{E}} + \begin{bmatrix}\mathbf{M}_{\bar{e}_1,\bar{e}_1} & \dots & \mathbf{M}_{\bar{e}_k,\bar{e}_1}\\ \mathbf{M}_{\bar{e}_1,\bar{e}_2} & \dots & \mathbf{M}_{\bar{e}_k,\bar{e}_2}\\ \vdots & \ddots & \vdots\\ \mathbf{M}_{\bar{e}_1,\bar{e}_k} & \dots & \mathbf{M}_{\bar{e}_k,\bar{e}_k}\end{bmatrix}\mathbf{X}_{\bar{\mathcal{E}}}. \tag{13}$$

Let

$$\mathbf{M}_{\mathcal{E}} \triangleq \begin{bmatrix}\mathbf{M}_{e_1,\bar{e}_1} & \dots & \mathbf{M}_{e_{n-k},\bar{e}_1}\\ \mathbf{M}_{e_1,\bar{e}_2} & \dots & \mathbf{M}_{e_{n-k},\bar{e}_2}\\ \vdots & \ddots & \vdots\\ \mathbf{M}_{e_1,\bar{e}_k} & \dots & \mathbf{M}_{e_{n-k},\bar{e}_k}\end{bmatrix}, \quad \mathbf{M}_{\bar{\mathcal{E}}} \triangleq \begin{bmatrix}\mathbf{M}_{\bar{e}_1,\bar{e}_1} & \dots & \mathbf{M}_{\bar{e}_k,\bar{e}_1}\\ \mathbf{M}_{\bar{e}_1,\bar{e}_2} & \dots & \mathbf{M}_{\bar{e}_k,\bar{e}_2}\\ \vdots & \ddots & \vdots\\ \mathbf{M}_{\bar{e}_1,\bar{e}_k} & \dots & \mathbf{M}_{\bar{e}_k,\bar{e}_k}\end{bmatrix}. \tag{14}$$

Then, from (13) and (14), we establish that

$$\mathbf{P}_{\bar{\mathcal{E}}} = \mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}} + \mathbf{M}_{\bar{\mathcal{E}}}\mathbf{X}_{\bar{\mathcal{E}}}. \tag{15}$$

In the following, we provide four lemmas that will be useful in characterizing a necessary condition for the existence of an $(n, k, \mathbf{m})$ irregular array code in Theorem 2.

Lemma 1. *Given a matrix $\mathbf{A} \in \mathbb{F}_q^{a\times b}$ and a random column vector $\mathbf{b} \in \mathbb{F}_q^b$, we have $H(\mathbf{A}\mathbf{b}) \le \mathsf{rank}(\mathbf{A})$.*

Proof. The lemma trivially holds when $\mathsf{rank}(\mathbf{A}) = 0$ or $\mathsf{rank}(\mathbf{A}) = \mathsf{row}(\mathbf{A})$. Here, we provide the proof subject to $\mathsf{row}(\mathbf{A}) > \mathsf{rank}(\mathbf{A}) > 0$. Given a matrix $\mathbf{A} \in \mathbb{F}_q^{a\times b}$, there is an invertible matrix $\mathbf{R}$ such that $\mathbf{RA} = \left[\begin{smallmatrix}\mathbf{A}'\\ [\mathbf{0}]\end{smallmatrix}\right]$, where $[\mathbf{0}]$ is a $(\mathsf{row}(\mathbf{A}) - \mathsf{rank}(\mathbf{A})) \times \mathsf{col}(\mathbf{A})$ zero matrix. Then, given $\mathbf{A}'\mathbf{b}$, we can determine $\mathbf{A}\mathbf{b}$ via $\mathbf{A}\mathbf{b} = \mathbf{R}^{-1}\left[\begin{smallmatrix}\mathbf{A}'\mathbf{b}\\ [\mathbf{0}]\end{smallmatrix}\right]$, and vice versa. Thus, we have $H(\mathbf{A}\mathbf{b}|\mathbf{A}'\mathbf{b}) = H(\mathbf{A}'\mathbf{b}|\mathbf{A}\mathbf{b}) = 0$. As $I(\mathbf{A}'\mathbf{b}; \mathbf{A}\mathbf{b}) = H(\mathbf{A}'\mathbf{b}) - H(\mathbf{A}'\mathbf{b}|\mathbf{A}\mathbf{b}) = H(\mathbf{A}\mathbf{b}) - H(\mathbf{A}\mathbf{b}|\mathbf{A}'\mathbf{b})$, we conclude $H(\mathbf{A}\mathbf{b}) = H(\mathbf{A}'\mathbf{b}) \le \mathsf{row}(\mathbf{A}'\mathbf{b}) = \mathsf{rank}(\mathbf{A})$. This completes the proof. □

The next three lemmas associate $\mathbf{m}$, $\mathbf{p}$ and $\{\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j})\}_{i,j\in[n]}$ through $\mathsf{rank}(\mathbf{M}_{\mathcal{E}})$.

Lemma 2. *Given any $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n - k$, if each codeword $\mathbf{C} \in \mathcal{C}$ can be determined uniquely by $\mathbf{C}_{\bar{\mathcal{E}}}$, then we have $\sum_{i\in\mathcal{E}} m_i \le \mathsf{rank}(\mathbf{M}_{\mathcal{E}})$.*

Proof. If the knowledge of $\mathbf{C}_{\bar{\mathcal{E}}}$ can reconstruct the entire $\mathcal{C}$, then $H(\mathbf{X}_{\mathcal{E}}|\mathbf{C}_{\bar{\mathcal{E}}}) = 0$. Thus, we have

$$I(\mathbf{C}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}) = H(\mathbf{X}_{\mathcal{E}}) - H(\mathbf{X}_{\mathcal{E}}|\mathbf{C}_{\bar{\mathcal{E}}}) = H(\mathbf{X}_{\mathcal{E}}) = \sum_{i\in\mathcal{E}} m_i. \tag{16}$$

Since $I(\mathbf{X}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}) = 0$ as indicated by (4), we get

$$\begin{aligned} I(\mathbf{C}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}) &= I(\mathbf{X}_{\bar{\mathcal{E}}}, \mathbf{P}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}})\\ &= I(\mathbf{X}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}) + I(\mathbf{P}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}|\mathbf{X}_{\bar{\mathcal{E}}})\\ &= I(\mathbf{P}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}|\mathbf{X}_{\bar{\mathcal{E}}}). \end{aligned} \tag{17}$$

We then obtain from (15) and (17) that

$$\begin{aligned} I(\mathbf{P}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}|\mathbf{X}_{\bar{\mathcal{E}}}) =& I(\mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}} + \mathbf{M}_{\bar{\mathcal{E}}}\mathbf{X}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}|\mathbf{X}_{\bar{\mathcal{E}}})\\ =& H(\mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}} + \mathbf{M}_{\bar{\mathcal{E}}}\mathbf{X}_{\bar{\mathcal{E}}}|\mathbf{X}_{\bar{\mathcal{E}}})\\ &- H(\mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}} + \mathbf{M}_{\bar{\mathcal{E}}}\mathbf{X}_{\bar{\mathcal{E}}}|\mathbf{X}_{\bar{\mathcal{E}}},\mathbf{X}_{\mathcal{E}})\\ =& H(\mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}}|\mathbf{X}_{\bar{\mathcal{E}}})\\ =& H(\mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}}), \end{aligned}$$

which, together with (16), (17) and Lemma 1, implies

$$\begin{aligned} \sum_{i\in\mathcal{E}} m_i = I(\mathbf{C}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}) &= I(\mathbf{P}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}|\mathbf{X}_{\bar{\mathcal{E}}})\\ &= H(\mathbf{M}_{\mathcal{E}}\mathbf{X}_{\mathcal{E}}) \le \mathsf{rank}(\mathbf{M}_{\mathcal{E}}). \end{aligned}$$

□

Lemma 3. $\mathsf{rank}(\mathbf{M}_{\mathcal{E}}) \le \sum_{j\in\bar{\mathcal{E}}} \min\{p_j, \sum_{i\in\mathcal{E}} \gamma_{i,j}\}$.

Proof. We first note from (14) that

$$\mathsf{rank}(\mathbf{M}_{\mathcal{E}}) \le \sum_{j\in[k]} \mathsf{rank}([\mathbf{M}_{e_1,\bar{e}_j} \dots \mathbf{M}_{e_{n-k},\bar{e}_j}]). \tag{18}$$

Using $\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j})$ from Theorem 1, we obtain

$$\begin{aligned} \mathsf{rank}([\mathbf{M}_{e_1,\bar{e}_j} \dots \mathbf{M}_{e_{n-k},\bar{e}_j}]) &\le \sum_{i\in[n-k]} \mathsf{rank}(\mathbf{M}_{e_i,\bar{e}_j})\\ &= \sum_{i\in[n-k]} \gamma_{e_i,\bar{e}_j}. \end{aligned} \tag{19}$$

Next, we note that

$$\begin{aligned} \mathsf{rank}([\mathbf{M}_{e_1,\bar{e}_j} \dots \mathbf{M}_{e_{n-k},\bar{e}_j}]) &\le \mathsf{row}([\mathbf{M}_{e_1,\bar{e}_j} \dots \mathbf{M}_{e_{n-k},\bar{e}_j}])\\ &= p_{\bar{e}_j}. \end{aligned} \tag{20}$$

Combining (19) and (20) yields

$$\mathsf{rank}([\mathbf{M}_{e_1,\bar{e}_j} \dots \mathbf{M}_{e_{n-k},\bar{e}_j}]) \le \min\left\{p_{\bar{e}_j}, \sum_{i\in[n-k]} \gamma_{e_i,\bar{e}_j}\right\}. \tag{21}$$

The validity of the lemma can thus be confirmed by (18) and (21). □

Lemma 4. $\mathsf{rank}(\mathbf{M}_{\mathcal{E}}) \le \sum_{i\in\mathcal{E}} \min\{m_i, \sum_{j\in\bar{\mathcal{E}}} \gamma_{i,j}\}$.

Proof. The proof of this lemma is similar to that of Lemma 3. First from (14), we establish

$$\mathsf{rank}(\mathbf{M}_{\mathcal{E}}) = \mathsf{rank}(\mathbf{M}_{\mathcal{E}}^{\mathsf{T}}) \le \sum_{i\in[n-k]} \mathsf{rank}([\mathbf{M}_{e_i,\bar{e}_1}^{\mathsf{T}} \dots \mathbf{M}_{e_i,\bar{e}_k}^{\mathsf{T}}]). \tag{22}$$

In parallel to (19) and (20), we next derive $\mathsf{rank}([\mathbf{M}_{e_i,\bar{e}_1}^{\mathsf{T}} \dots \mathbf{M}_{e_i,\bar{e}_k}^{\mathsf{T}}]) \le \sum_{j\in[k]} \gamma_{e_i,\bar{e}_j}$ and $\mathsf{rank}([\mathbf{M}_{e_i,\bar{e}_1}^{\mathsf{T}} \dots \mathbf{M}_{e_i,\bar{e}_k}^{\mathsf{T}}]) \le \mathsf{row}([\mathbf{M}_{e_i,\bar{e}_1}^{\mathsf{T}} \dots \mathbf{M}_{e_i,\bar{e}_k}^{\mathsf{T}}]) = m_{e_i}$, which immediately gives

$$\mathsf{rank}([\mathbf{M}_{e_i,\bar{e}_1}^{\mathsf{T}} \dots \mathbf{M}_{e_i,\bar{e}_k}^{\mathsf{T}}]) \le \min\left\{m_{e_i}, \sum_{j\in[k]} \gamma_{e_i,\bar{e}_j}\right\}. \tag{23}$$

The lemma then follows from (22) and (23). □

After establishing the above four lemmas, we are now ready to prove the main result in this section.

Theorem 2. *Given any $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n - k$, if each codeword $\mathbf{C} \in \mathcal{C}$ can be determined uniquely by $\mathbf{C}_{\bar{\mathcal{E}}}$, then the following inequalities must hold:*

$$\sum_{j\in\bar{\mathcal{E}}} \gamma_{i,j} \geq m_i \qquad \forall i \in \mathcal{E}, \tag{24}$$

$$\sum_{i\in\mathcal{E}} m_i \leq \sum_{j\in\bar{\mathcal{E}}} \min\left\{p_j, \sum_{i\in\mathcal{E}} \gamma_{i,j}\right\}. \tag{25}$$

Proof. Inequality (25) is an immediate consequence of Lemmas 2 and 3.

The inequality in (24) can be proved by contradiction. Suppose $\sum_{j\in\bar{\mathcal{E}}} \gamma_{u,j} < m_u$ for some $u \in \mathcal{E}$. Then, we can infer from Lemma 4 that

$$\begin{aligned} \mathsf{rank}(\mathbf{M}_{\mathcal{E}}) &\leq \sum_{i\in\mathcal{E}\setminus\{u\}} \min\left\{m_i, \sum_{j\in\bar{\mathcal{E}}} \gamma_{i,j}\right\} + \sum_{j\in\bar{\mathcal{E}}} \gamma_{u,j} \\ &< \sum_{i\in\mathcal{E}\setminus\{u\}} m_i + m_u = \sum_{i\in\mathcal{E}} m_i, \end{aligned} \tag{26}$$

which contradicts to Lemma 2. Consequently, inequality (24) must hold for every $i \in \mathcal{E}$. □

For completeness, we conclude the section by reiterating the result in Theorem 2 in the following corollary.

Corollary 1. *An $(n, k, \mathbf{m})$ irregular array code $\mathcal{C}$ with construction matrices $\{\mathbf{M}_{i,j}\}_{i,j\in[n]}$ and numbers of parity symbols specified in $\mathbf{p}$ must fulfill* (24) *and* (25) *for every $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n - k$.*

IV. Lower bounds for code redundancy and update bandwidth

In this section, three lower bounds will be established, which are lower bounds respectively for code redundancy and update bandwidth, and a lower bound for code redundancy subject to the minimum update bandwidth. Their achievability by explicit constructions of irregular array codes under $k \mid m_i$ for all $i \in [n]$ will be shown in Section V. Without loss of generality, we assume in this section that

$$m_1 \geq m_2 \geq \cdots \geq m_n \geq 0. \tag{27}$$

A. Minimization of code redundancy

Theorem 2 indicates that a lower bound for the code redundancy of an $(n, k, \mathbf{m})$ irregular array code can be obtained by solving the integer program problem below.[2]

[2]In order to keep the constraints close to the respective theorems that follow, we will repeat (24) and (25) in Integer Programings 1-6 whenever they are applied. Specifically, (24) will reappear in (28a), (37a) and (39), and (25) will be repeated in (28b), (37b), (46b) and (54b).

Integer Program 1. *To minimize $R = \sum_{i=1}^{n} p_i$ subject to*

$$\begin{cases} \sum_{j\in\bar{\mathcal{E}}} \gamma_{i,j} \geq m_i \qquad \forall i \in \mathcal{E} & \text{(28a)} \\ \sum_{i\in\mathcal{E}} m_i \leq \sum_{j\in\bar{\mathcal{E}}} \min\left\{p_j, \sum_{i\in\mathcal{E}} \gamma_{i,j}\right\} & \text{(28b)} \end{cases}$$

among all $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n - k$.

Since the object function of Integer Program 1 is only a function of $\mathbf{p}$, a code redundancy R is attainable due to a choice of $\mathbf{p}$, if there exists a set of corresponding $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ that can validate both (28a) and (28b). A valid selection of such $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ for a given $\mathbf{p}$ is to persistently increase $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ until both (28a) and

$$p_j \leq \sum_{i\in\mathcal{E}} \gamma_{i,j} \qquad \forall j \in \bar{\mathcal{E}} \tag{29}$$

are satisfied for arbitrary choice of $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n - k$. As a result, we can disregard (28a) and reduce (28b) to

$$\sum_{i\in\mathcal{E}} m_i \leq \sum_{j\in\bar{\mathcal{E}}} p_j, \tag{30}$$

leading to a new integer program setup as follows.

Integer Program 2. *To minimize $R = \sum_{i=1}^{n} p_i$ subject to*

$$\sum_{i\in\mathcal{E}} m_i \leq \sum_{j\in\bar{\mathcal{E}}} p_j \tag{31}$$

among all $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n - k$.

Lemma 5. *Integer Program 1 is equivalent to Integer Program 2.*

Proof. It is obvious that all minimizers of Integer Program 1 satisfy the constraint in Integer Program 2. On the contrary, given a minimizer $\mathbf{p}$ of Integer Program 2, we can assign $\gamma_{i,j} = \max\{m_i, p_j\}$ to satisfy the constraints in Integer Program 1. Thus, Integer Program 1 and Integer Program 2 are equivalent. □

Remark 1. *Note that Integer Program 2, which was first given in [27], is not related to the update bandwidth γ of an irregular array code, while the proposed setup in Integer Program 1 is. Thus, the latter setup can be used to determine an irregular array code of update-bandwidth efficiency by replacing the object function R with update bandwidth γ. However, for the minimization of code redundancy, the two integer program settings are equivalent as confirmed in Lemma 5.*

To solve Integer Program 2, Tosato and Sandell [27] introduced a water level parameter μ, defined as

$$\mu = \max\left\{m_{n-k}, \left\lceil \frac{B}{k} \right\rceil\right\}, \tag{32}$$

where $B \triangleq \sum_{i\in[n]} m_i$ is the total number of data symbols, and by following the assumption in (27), m_{n-k} is the $(n-k)$-

th largest element in vector $\mathbf{m}$. It was shown in [27] that the minimum code redundancy equals

$$R_{\min} = \sum_{i=1}^{n-k}([\mu - m_i]_+ + m_i), \tag{33}$$

which can only be achieved by those $\mathbf{p}$'s satisfying

$$\begin{cases} p_i = [\mu - m_i]_+ & \text{for } 1 \le i \le n-k, \\ p_i \le [\mu - m_i]_+ & \text{for } n-k < i \le n, \\ \sum_{i=n-k+1}^{n} p_i = \sum_{i=1}^{n-k} m_i, \end{cases} \tag{34}$$

where $[x]_+ \triangleq \max\{0, x\}$. The class of $(n, k, \mathbf{m})$ irregular array codes conforming to (34) is called irregular MDS array codes [27]. In particular, when $k \mid B$ and $m_1 \le B/k$, (33) and (34) can be respectively reduced to

$$R_{\min} = \frac{(n-k)}{k} B, \tag{35}$$

and

$$p_i = \frac{B}{k} - m_i \qquad \forall i \in [n]. \tag{36}$$

B. Minimization of update bandwidth

We now turn to the determination of the minimum update bandwidth. As similar to Integer Program 1, a lower bound for the update bandwidth of an $(n, k, \mathbf{m})$ irregular array code can be obtained using the integer program below.

Integer Program 3. *To minimize* $\gamma = \frac{1}{n}\sum_{i=1}^{n}\sum_{j\in[n]\setminus\{i\}} \gamma_{i,j}$ *subject to*

$$\begin{cases} \sum_{j\in\bar{\mathcal{E}}} \gamma_{i,j} \ge m_i \qquad \forall i \in \mathcal{E} & (37a) \\ \sum_{i\in\mathcal{E}} m_i \le \sum_{j\in\bar{\mathcal{E}}} \min\left\{p_j, \sum_{i\in\mathcal{E}} \gamma_{i,j}\right\} & (37b) \end{cases}$$

among all $\mathcal{E} \subset [n]$ *with* $|\mathcal{E}| = n - k$.

Since $\mathbf{p}$ is not used in the above object function, a choice of $\{\gamma_{i,j}\}_{i\ne j\in[n]}$ is feasible for the minimization of γ as long as there is a corresponding $\mathbf{p}$ that validates both (37a) and (37b). A valid selection of such $\mathbf{p}$ for given $\{\gamma_{i,j}\}_{i\ne j\in[n]}$ is to set $p_j = \sum_{i\in[n]} \gamma_{i,j}$, which reduces (37b) to a consequence of (37a), i.e.,

$$\sum_{i\in\mathcal{E}} m_i \le \sum_{j\in\bar{\mathcal{E}}}\sum_{i\in\mathcal{E}} \gamma_{i,j}. \tag{38}$$

As a result, by following an analogous proof to that used in Lemma 5, Integer Program 3 can also be solved through the following equivalent setup.

Integer Program 4. *To minimize* $\gamma = \frac{1}{n}\sum_{i=1}^{n}\sum_{j\in[n]\setminus\{i\}} \gamma_{i,j}$ *subject to*

$$\sum_{j\in\bar{\mathcal{E}}} \gamma_{i,j} \ge m_i \qquad \forall i \in \mathcal{E} \tag{39}$$

among all $\mathcal{E} \subset [n]$ *with* $|\mathcal{E}| = n - k$.

The solution of Integer Program 4 is then given in the following theorem.

Theorem 3. *(Minimum update bandwidth) The minimum update bandwidth determined through Integer Program 4 is given by*

$$\gamma_{\min} = \frac{B}{n} + \frac{(n-k-1)}{n}\sum_{i\in[n]}\left\lceil \frac{m_i}{k} \right\rceil. \tag{40}$$

Under $k < n-1$*, the minimum update bandwidth can only be achieved by the assignment that satisfies for every* $i \in [n]$*,*

$$\begin{cases} \sum_{u\in[w_i]} \gamma_{i,j_u(i)} = w_i \left\lfloor \frac{m_i}{k} \right\rfloor, \\ \gamma_{i,j_u(i)} = \left\lceil \frac{m_i}{k} \right\rceil \qquad \forall u \in [n-1] \setminus [w_i], \end{cases} \tag{41}$$

where $w_i \triangleq k\left\lceil \frac{m_i}{k} \right\rceil - m_i < k$,[3] *and for notational convenience, we let the indices* $j_1(i), j_2(i), \ldots, j_{n-1}(i)$ *be a permutation of* $[n] \setminus \{i\}$ *such that*

$$0 \le \gamma_{i,j_1(i)} \le \cdots \le \gamma_{i,j_{n-1}(i)} \quad \text{for } i \in [n]. \tag{42}$$

When $k = n-1$*, any* $\{\gamma_{i,j}\}_{i\ne j\in[n]}$ *that achieves* $\gamma_{\min}$ *must satisfy*

$$\sum_{j\in[n]\setminus\{i\}} \gamma_{i,j} = m_i \quad \forall i \in [n]. \tag{43}$$

Proof. For better readability, the proof is relegated to Appendix B. □

Note that under $k = n-1$, the assignment that achieves $\gamma_{\min}$ is no longer restricted to (41) but must be generalized to (43). An example is provided below.

Example 1. *For an* $(3, 2, \mathbf{m} = [5\ 5\ 5]^{\mathsf{T}})$ *irregular array code, the assignment of* (41) *gives* $\gamma_{i,j_1(i)} = 2$ *and* $\gamma_{i,j_2(i)} = 3$ *for* $i \in [3]$*, but*

$$\gamma_{i,j} = \begin{cases} 5, & (i,j) \in \{(1,3),(2,3),(3,2)\} \\ 0, & (i,j) \in \{(1,2),(2,1),(3,1)\} \end{cases} \tag{44}$$

can also achieve $\gamma_{\min} = 5$*. This justifies our separate consideration of the case of* $k = n-1$*.*

C. Determination of the smallest code redundancy subject to $\gamma = \gamma_{\min}$

In Theorem 3, the class of optimal $\{\gamma_{i,j}\}_{i\ne j\in[n]}$ that achieve $\gamma_{\min}$ is also determined. In particular, when $k < n-1$ and $k \mid m_i$ for every i, we have $w_i = 0$ and $[w_i] = \emptyset$ for all $i \in [n]$, which together with (41) indicates that

$$\gamma_{i,j} = \frac{m_i}{k} \quad \forall i \ne j \in [n] \tag{45}$$

uniquely achieves $\gamma_{\min}$. This facilitates our finding the smallest code redundancy attainable subject to $\gamma = \gamma_{\min}$ as formulated in Integer Program 5 below.

[3] A particular situation that will be considered in the next subsection is $k \mid m_i$, in which case we have $w_i = 0$ and $[w_i] = \emptyset$, and (41) is reduced to $\gamma_{i,j_u(i)} = \frac{m_i}{k}$ for $u \in [n-1]$.

Integer Program 5. *To minimize $R=\sum_{i=1}^n p_i$ subject to*

$$\begin{cases}\gamma_{i,j}=\dfrac{m_i}{k} & \forall i\neq j\in[n] \qquad (46a)\\ \displaystyle\sum_{i\in\mathcal{E}} m_i\leq\sum_{j\in\bar{\mathcal{E}}}\min\left\{p_j,\sum_{i\in\mathcal{E}}\gamma_{i,j}\right\} & \qquad (46b)\end{cases}$$

among all $\mathcal{E}\subset[n]$ with $|\mathcal{E}|=n-k$, provided $1\leq k<n-1$ and $k\mid m_i$ for all $i\in[n]$.

Theorem 4. *The solution of Integer Program 5 is given by*

$$R_{\texttt{sma}}\triangleq\frac{(n-1)}{k}\sum_{i=1}^{n-k}m_i+\frac{(n-k)}{k}m_{n-k+1}, \qquad (47)$$

where by following the assumption in (27), m_i is the i-th largest element in vector $\mathbf{m}$. The smallest code redundancy subject to $\gamma=\gamma_{\min}$ is uniquely achieved by

$$p_j=\begin{cases}\frac{1}{k}\sum_{i\in[n-k+1]\setminus\{j\}}m_i & \textit{for } 1\leq j\leq n-k,\\ \frac{1}{k}\sum_{i=1}^{n-k}m_i & \textit{for } n-k<j\leq n.\end{cases} \qquad (48)$$

Proof. We first prove by contradiction that

$$p_j\geq\sum_{i\in\mathcal{E}}\gamma_{i,j}=\frac{1}{k}\sum_{i\in\mathcal{E}}m_i \qquad \forall\mathcal{E}\text{ and }\forall j\in\bar{\mathcal{E}}. \qquad (49)$$

Suppose there are $\mathcal{E}\subset[n]$ with $|\mathcal{E}|=n-k$ and $j'\in\bar{\mathcal{E}}$ such that

$$p_{j'}<\sum_{i\in\mathcal{E}}\gamma_{i,j'}=\frac{1}{k}\sum_{i\in\mathcal{E}}m_i. \qquad (50)$$

Then, (46b) results in a contradiction as follows:

$$\begin{aligned}\sum_{i\in\mathcal{E}}m_i&\leq\min\left\{p_{j'},\sum_{i\in\mathcal{E}}\gamma_{i,j'}\right\}+\sum_{j\in\bar{\mathcal{E}}\setminus\{j'\}}\min\left\{p_j,\sum_{i\in\mathcal{E}}\gamma_{i,j}\right\}\\ &<\sum_{j\in\bar{\mathcal{E}}}\sum_{i\in\mathcal{E}}\gamma_{i,j}\\ &=\sum_{j\in\bar{\mathcal{E}}}\sum_{i\in\mathcal{E}}\frac{m_i}{k}=\sum_{i\in\mathcal{E}}m_i, \qquad (51)\end{aligned}$$

where (51) follows from (46a). Thus, (49) holds for arbitrary $\mathcal{E}\subset[n]\setminus\{j\}$. As a result, we have

$$\begin{aligned}p_j&\geq\max_{\mathcal{E}\subset[n]\setminus\{j\}:|\mathcal{E}|=n-k}\frac{1}{k}\sum_{i\in\mathcal{E}}m_i\\ &=\begin{cases}\frac{1}{k}\sum_{i\in[n-k+1]\setminus\{j\}}m_i & \text{for } 1\leq j\leq n-k,\\ \frac{1}{k}\sum_{i=1}^{n-k}m_i & \text{for } n-k<j\leq n,\end{cases}\end{aligned} \qquad (52)$$

which implies

$$R=\sum_{j=1}^n p_j\geq\frac{(n-1)}{k}\sum_{i=1}^{n-k}m_i+\frac{(n-k)}{k}m_{n-k+1}=R_{\texttt{sma}}. \qquad (53)$$

Since any $\{p_j\}_{j\in[n]}$ that satisfies (52) with strict inequality for some $j\in[n]$ cannot achieve $R_{\texttt{sma}}$, the smallest code redundancy subject to $\gamma=\gamma_{\min}$ is uniquely achieved by the one that fulfills (52) with equality. □

The contradiction proof in (51) requires $\sum_{j\in\bar{\mathcal{E}}}\gamma_{i,j}=m_i$, which is guaranteed by (41) when $k\mid m_i$ for all $i\in[n]$. However, without $k\mid m_i$ for all $i\in[n]$, the $\sum_{j\in\bar{\mathcal{E}}}\gamma_{i,j}$ in (41) may not achieve m_i but generally lies between m_i and $k\left\lceil\frac{m_i}{k}\right\rceil$. Our preliminary study indicates that the general formula of $R_{\texttt{sma}}$ for arbitrary $k<n-1$ and arbitrary $\mathbf{m}$ does not seem to have a simple expression but depends on the pattern of $\mathbf{w}=[w_1\ w_2\ \cdots w_n]^\mathsf{T}$. Theorem 5 only deals with $\mathbf{w}=[0\ 0\ \cdots\ 0]^\mathsf{T}$. The establishment of the smallest code redundancy for cases that allow $k\nmid m_i$ is left as a future research.

Surprisingly, in the particular case of $k=n-1$, we found $R_{\texttt{sma}}=R_{\min}$ due to the fact that $\sum_{j\in\bar{\mathcal{E}}}\gamma_{i,j}=m_i$ is guaranteed by (43).

Integer Program 6. *To minimize $R=\sum_{i=1}^n p_i$ subject to*

$$\begin{cases}\displaystyle\sum_{j\in[n]\setminus\{i\}}\gamma_{i,j}=m_i \quad \forall i\in[n] & \qquad (54a)\\ \displaystyle\sum_{i\in\mathcal{E}}m_i\leq\sum_{j\in\bar{\mathcal{E}}}\min\left\{p_j,\sum_{i\in\mathcal{E}}\gamma_{i,j}\right\} & \qquad (54b)\end{cases}$$

among all $\mathcal{E}\subset[n]$ with $|\mathcal{E}|=n-k$, provided $k=n-1$.

Theorem 5. *The solution of Integer Program 6 is given by the $R_{\min}$ in (33), which can only be achieved by those $\mathbf{p}$'s satisfying (34).*

Proof. It suffices to prove that Integer Program 2 and Integer Program 6 are equivalent under $n-k=1$. We first note that under $n-k=1$, all feasible $\mathbf{p}$ and $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ satisfying (54a) and (54b), i.e.,

$$m_i=\sum_{j\in[n]\setminus\{i\}}\gamma_{i,j}\leq\sum_{j\in[n]\setminus\{i\}}\min\{p_j,\gamma_{i,j}\} \quad \forall i\in[n], \qquad (55)$$

must validate (30), i.e.,

$$m_i\leq\sum_{j\in[n]\setminus\{i\}}p_j \quad \forall i\in[n]. \qquad (56)$$

On the contrary, for every $\mathbf{p}$ that fulfills (56), we can always construct $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ with $\gamma_{i,j}\leq p_j$ such that (55) holds. Thus, Integer Program 2 is equivalent to Integer Program 6. □

V. Explicit constructions of MUB and MR-MUB codes

A. MR-MUB and MUB codes

Based on the previous section, we can now define two particular classes of irregular array codes.

Definition 2. *A Minimal Update Bandwidth (MUB) code is an $(n,k,\mathbf{m})$ irregular array code with update bandwidth equal to $\gamma_{\min}$.*

Definition 3. *A Minimum Redundancy and Minimum Update Bandwidth (MR-MUB) code is an $(n,k,\mathbf{m})$ irregular array code, of which the code redundancy and the update bandwidth are equal to $R_{\min}$ and $\gamma_{\min}$, respectively.*

Note that the existence of MR-MUB codes for certain parameters n, k and $\mathbf{m}$ is not guaranteed. In certain cases, we can only have $R_{\mathtt{sma}} > R_{\min}$, i.e., the smallest code redundancy subject to $\gamma = \gamma_{\min}$ is strictly larger than the minimum code redundancy among all irregular array codes. An example is given in Fig. 3, where we can obtain from (47) that the smallest code redundancy of $(4, 2, \mathbf{m} = [4\ 2\ 2\ 0]^{\mathsf{T}})$ irregular array codes is equal to

$$R_{\mathtt{sma}} = \frac{3}{2}\sum_{i=1}^{2} m_i + \frac{2}{2}m_3 = \frac{3}{2}(4+2)+2 = 11, \quad (57)$$

while the minimum code redundancy in (33) is given by

$$R_{\min} = ([4-4]_+ + 4) + ([4-2]_+ + 2) = 8. \quad (58)$$

It can be verified that the code redundancy of the irregular array code in Fig. 3 achieves $p_1+p_2+p_3+p_4 = 11 = R_{\mathtt{sma}}$.

To confirm that the code in Fig. 3 is an MUB code, we note that the update of the first node has to send $\Delta x_{1,1}$ and $\Delta x_{1,2}$ to node 2, $\Delta x_{1,3}$ and $\Delta x_{1,4}$ to node 3, $(\Delta x_{1,1} + \Delta x_{1,3})$ and $(\Delta x_{1,2} + \Delta x_{1,4})$ to node 4, respectively. Thus, the required update bandwidth for node 1 is 6. Similarly, we can verify that the required bandwidths of the second, the third and the fourth nodes are 3, 3 and 0, respectively. As a result, $\gamma = \frac{1}{4}(6+3+3+0) = 3$, which equals $\gamma_{\min}$ in (40).

Two particular situations, which guarantee the existence of MR-MUB codes, are $k = 1$ and $k = n-1$. In the former situation, we can obtain from (47) and (33) that

$$R_{\mathtt{sma}} = R_{\min} = \frac{(n-k)}{k}B = \frac{(n-k)}{k}\sum_{i\in[n]} m_i, \quad (59)$$

while the latter has been proven in Theorem 5. For $1 < k < n-1$, however, it is interesting to find that an MR-MUB code exists only when $\mathbf{m}$ is either an extremely balanced all-equal vector or an extremely unbalanced all-zero-but-one vector, which is proven in the next theorem under $k \mid m_i$ for all $i \in [n]$.

Theorem 6. *Under $1 < k < n-1$ and $k \mid m_i$ for all $i \in [n]$, $(n, k, \mathbf{m})$ MR-MUB codes exist if, and only if, one of the two situations occurs:*

$$\begin{cases} m_i = \frac{B}{n}\ \forall i \in [n], \\ p_j = \frac{(n-k)}{nk}B\ \forall j \in [n]. \end{cases} \quad (60)$$

and

$$\begin{cases} m_1 = B,\ \text{and}\ m_i = 0\ \text{for}\ 2 \le i \le n, \\ p_1 = 0,\ \text{and}\ p_j = \frac{B}{k}\ \text{for}\ 2 \le j \le n. \end{cases} \quad (61)$$

In either situation, $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ follows from (46a).

Proof. The theorem can be proved by simply equating the two p_1's that respectively achieve $R_{\min}$ and $R_{\mathtt{sma}}$. Specifically, (34) indicates that $R_{\min}$ is achieved by $p_1 = [\mu - m_1]_+$, where μ is given in (32). From (48), $R_{\mathtt{sma}}$ is reached when

$$p_1 = \frac{1}{k}\left(\sum_{i=1}^{n-k+1} m_i - m_1\right) = \frac{1}{k}\sum_{i=2}^{n-k+1} m_i. \quad (62)$$

We thus have

$$p_1 = [\mu - m_1]_+ = \frac{1}{k}\sum_{i=2}^{n-k+1} m_i. \quad (63)$$

We then distinguish between two cases: $p_1 = 0$ and $p_1 > 0$.

Consider $p_1 = \frac{1}{k}\sum_{i=2}^{n-k+1} m_i = 0$, which from (27), immediately leads to $m_1 = B$ and $m_2 = m_3 = \cdots = m_n = 0$. Thus, we obtain from (32) and (34) that $\mu = \frac{B}{k}$ and $p_j = \frac{B}{k}$ for $2 \le j \le n$. As anticipated, this $\mathbf{p}$ also satisfies (48) and validates $R_{\min} = R_{\mathtt{sma}}$.

Next, we consider $p_1 = [\mu - m_1]_+ > 0$, which leads to $\mu > m_1$. As $m_{n-k} \le m_1$ from (32), we have $\mu = \frac{B}{k} > m_1$. Thus, (63) becomes

$$\frac{B}{k} - m_1 = \frac{1}{k}\sum_{i=2}^{n-k+1} m_i, \quad (64)$$

which implies

$$(k-1)m_1 = \underbrace{m_{n-k+2} + \cdots + m_n}_{k-1}. \quad (65)$$

We can then conclude from (27) that $m_1 = m_2 = \cdots = m_n$. The verification of $R_{\min} = R_{\mathtt{sma}}$ straightforwardly follows. □

In practice, it may be unusual to place all data symbols in one node. Thus, we will focus on the construction of MR-MUB codes that follows (60) in the next subsection. In other words, the $(n, k, \mathbf{m} = [m\ m\ \cdots\ m]^{\mathsf{T}})$ MR-MUB codes considered in the rest of the paper are (n, k) vertical MDS array codes with each node containing m data symbols and $p = \frac{(n-k)}{k}m$ parity symbols subject to $k \mid m$.

Note that Theorem 6 seems limited in its applicability since (60) simply shows vertical MDS codes can achieve both $R_{\min}$ and $\gamma_{\min}$ under a particular case of $k \mid m$. However, without the condition of $k \mid m$, vertical MDS array codes may not form a sub-class of MR-MUB codes. This can be justified by two observations. First, it can be verified from (34) that the fulfillment of both $p_i = [\mu - m_i]_+$ for $1 \le i \le n-k$ and $\sum_{i=n-k+1}^{n} p_i = \sum_{i=1}^{n-k} m_i$ under each $p_i = p$ and each $m_i = m$ requires $k \mid nm$. Thus, under $k \nmid nm$, vertical MDS array codes cannot achieve the minimum code redundancy, and hence cannot be MR-MUB codes. Second, when $k \mid nm$ but $k \nmid m$, examples and counterexamples for vertical MDS array codes being able to achieve simultaneously $R_{\min}$ and $\gamma_{\min}$ can both be constructed as follows. Hence, we conjecture that $k \mid m$ is also a necessary condition for vertical MDS array codes being MUB codes, provided $k \nmid n$.

Example 2. *A supporting example of our conjecture follows when $n = 6$, $k = 3$ and $m = 4$, where setting $p_i = p = 4$ and*

$$\gamma_{i,j} = \begin{cases} 1, & j \in \{(i\, mod\, 6)+1, [(i+1)\, mod\, 6]+1\} \\ 2, & otherwise \end{cases} \quad (66)$$

for $i \neq j \in [n]$ fulfills both (24) *and* (25)*, and achieves simultaneously $R_{\min}$ and $\gamma_{\min}$.*

Example 3. *A counterexample to our conjecture exists when $n = 9$, $k = 6$ and $m = 2$. From* (34)*, we know $R_{\min}$ can only be achieved by adopting $p_j = 1$ for $j \in [n]$. By* (41)*, the achievability of $\gamma_{\min}$ requires $\gamma_{i,j_1(i)} = \gamma_{i,j_2(i)} = \gamma_{i,j_3(i)} = \gamma_{i,j_4(i)} = 0$ and $\gamma_{i,j_5(i)} = \gamma_{i,j_6(i)} = \gamma_{i,j_7(i)} = \gamma_{i,j_8(i)} = 1$ for every $i \in [n]$. Then, the pigeon hole principle implies*

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2021.3105439

that there is j' such that $\{\gamma_{i,j'}\}_{i\in[n]\setminus\{j'\}}$ contains at least four 0's. Let $\gamma_{i_1,j'}=\gamma_{i_2,j'}=\gamma_{i_3,j'}=\gamma_{i_4,j'}=0$ and $\mathcal{E}=\{i_1,i_2,i_3\}$, where $j'\notin\{i_1,i_2,i_3,i_4\}$. A violation to (25) *can thus be obtained as follows: $\sum_{i\in\mathcal{E}} m_i=|\mathcal{E}|m=6$, and $\sum_{j\in\bar{\mathcal{E}}}\min\{p_j,\sum_{i\in\mathcal{E}}\gamma_{i,j}\}=\min\{p_{j'},\underbrace{\sum_{i\in\mathcal{E}}\gamma_{i,j'}}_{=0}\}+\sum_{j\in\bar{\mathcal{E}}\setminus\{j'\}}\min\{p_j,\sum_{i\in\mathcal{E}}\gamma_{i,j}\}\le 5$. Consequently,* $(9,6)$ *vertical MDS array codes with each node having $m=2$ data symbols cannot be MR-MUB codes.*

Theorem 6 only deals with the situation of $1<k<n-1$. For completeness, the next corollary incorporates also the two particular cases of $k=1$ and $k=n-1$.

Corollary 2. *Under $1\le k<n$ and $k\mid m$, an $(n,k,m\mathbf{1})$ MR-MUB code must parameterize with*

$$p_j=\frac{(n-k)}{k}m\triangleq p\quad\forall j\in[n],\tag{67}$$

$$\gamma_{i,j}=\frac{m}{k}\quad\forall i\neq j\in[n],\tag{68}$$

where $\mathbf{1}\triangleq[1\ 1\ \cdots\ 1]^{\mathsf{T}}$ is the all-one vector.

Proof. We only substantiate the corollary for $k=1$ and $k=n-1$ since the situation of $1<k<n-1$ has been proved in Theorem 6. The validity of (67) under $k=1$ and $k=n-1$ can be confirmed by (36). We can also obtain from (46a) that (68) holds under $k=1$. It remains to verify (68) under $k=n-1$ by contradiction.

Fix $k=n-1$. Suppose there is a $j'\in[n]\setminus\{i\}$ such that $\gamma_{i,j'}<\frac{m}{k}=p_{j'}$. A contradiction can be established from (25) as follows:

$$\begin{aligned}m=m_i&\le\sum_{j\in[n]\setminus\{i\}}\min\{p_j,\gamma_{i,j}\}\\&\le\sum_{j\in[n]\setminus\{i,j'\}}p_j+\gamma_{i,j'}<\sum_{j\in[n]\setminus\{i\}}p_j=m.\end{aligned}\tag{69}$$

Accordingly, $\gamma_{i,j}\ge\frac{m}{k}$ for all $i\neq j\in[n]$, which implies

$$\sum_{j\in[n]\setminus\{i\}}\gamma_{i,j}\ge\frac{m}{k}(n-1)=m.\tag{70}$$

By noting from (43) that the inequality in (70) must be replaced by an equality, (68) holds under $k=n-1$. □

As horizontal array codes are widely implemented in DSSs, we close this subsection by checking whether an $(n,k,m\mathbf{1}_k)$ horizontal array code can achieve the minimum update bandwidth $\gamma_{\min}$, where $\mathbf{1}_k\triangleq[1\ \cdots\ 1\ 0\ \cdots\ 0]^{\mathsf{T}}$ is the vector, the first k components of which equal one and the remaining of which are zero.[4] As a result, $\mathbf{M}_{i,j}$ is the zero matrix for either $i\in[n]\setminus[k]$ or $j\in[k]$, implying from Theorem 1 that the corresponding $\gamma_{i,j}=\mathsf{rank}(\mathbf{M}_{i,j})=0$. We then derive from (24) that for $i\in[k]$, $j'\in[n]\setminus[k]$ and $\bar{\mathcal{E}}=\{j'\}\cup[k]\setminus\{i\}$,

$$\sum_{j\in\bar{\mathcal{E}}}\gamma_{i,j}=\gamma_{i,j'}\ge m_i=m,\tag{71}$$

[4]From its definition given in Section II-B, an (n,k) horizontal array code should parameterize with $m_1=m_2=\cdots=m_k=p_{k+1}=p_{k+2}=\cdots=p_n=m$ and $p_1=p_2=\cdots=p_k=m_{k+1}=m_{k+2}=\cdots=m_n=0$.

and hence the update bandwidth of an $(n,k,m\mathbf{1}_k)$ horizontal array code must satisfy

$$\gamma=\frac{1}{n}\sum_{i\in[n]}\sum_{j\in[n]\setminus[k]}\gamma_{i,j}=\frac{1}{n}\sum_{i\in[k]}\sum_{j\in[n]\setminus[k]}\gamma_{i,j}\tag{72}$$

$$\ge\frac{(n-k)km}{n}.\tag{73}$$

Since $\gamma_{i,j}\ge m$ for $i\in[k]$ and $j\in[n]\setminus[k]$, it is obvious that the lower bound in (73) can only be achieved by $\gamma_{i,j}=m$ for $i\in[k]$ and $j\in[n]\setminus[k]$, and $\gamma_{i,j}=0$, otherwise. Comparing (73) with the minimum update bandwidth $\gamma_{\min}$ for general irregular array codes in Theorem 3, we found that other than the two trivial cases of $k=1$ and $k=n-1$, an $(n,k,m\mathbf{1}_k)$ horizontal array code cannot achieve $\gamma_{\min}$ except when $m=1$, and therefore cannot be an MUB code. This finding complements the result in Theorem 6 as it holds without the condition $k\mid m_i$ for all $i\in[n]$.

Theorem 7. *Under $1<k<n-1$, the update bandwidth of an $(n,k,m\mathbf{1}_k)$ horizontal array code cannot achieve $\gamma_{\min}$ except when $m=1$.*

Proof. It suffices to prove that for $1<k<n-1$ and $m>1$, the lower bound in (73) is greater than $\gamma_{\min}$, i.e.,

$$\begin{aligned}\frac{(n-k)km}{n}>\gamma_{\min}&=\frac{mk+(n-k-1)k\left\lceil\frac{m}{k}\right\rceil}{n}\\&=\frac{mk}{n}\left(1+\frac{(n-k-1)\left\lceil\frac{m}{k}\right\rceil}{m}\right),\end{aligned}\tag{74}$$

which can be verified straightforwardly via $m>\left\lceil\frac{m}{k}\right\rceil$. The proof is completed by noting that under $m=1$, (74) becomes an equality, and the previously mentioned setting to achieve (73) coincides with (41). □

B. Construction of MR-MUB codes

For the construction of an $(n,k,m\mathbf{1})$ MR-MUB code, denoted as $\mathcal{C}_{\text{O}}$ for convenience, we require $\mathbf{x}_i\in\mathbb{F}_q^m$ and $\mathbf{p}_j\in\mathbb{F}_q^p$ with $p=\frac{(n-k)}{k}m$ for $i,j\in[n]$. The construction of $\{\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$ and $\{\mathbf{B}_{i,j}\}_{i\neq j\in[n]}$ associated with $\mathcal{C}_{\text{O}}$ are then addressed as follows.

First, we construct $\{\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$ of dimension $\frac{m}{k}\times m$. We begin by choosing an $(n-1,k)$ MDS array code $\mathcal{M}$ over $\mathbb{F}_q$ with encoding function $\mathcal{F}:\mathbb{F}_q^m\longrightarrow\mathbb{F}_q^{\frac{m}{k}\times(n-1)}$, where $\frac{m}{k}$ is the number of rows of the MDS array code, and m is the number of all data symbols of $\mathcal{M}$. For example, subject to $q\ge n-1$, we can let $\mathcal{M}$ be an MDS array code, where each row of the code array is a codeword of the $(n-1,k)$ Reed-Solomon (RS) code. Denote $\mathbf{F}_i\triangleq\mathcal{F}(\mathbf{x}_i)$. Then, $\{\mathbf{p}_{i,j}\}_{i\neq j\in[n]}$ defined in (9), as well as $\{\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$, can be characterized via

$$\mathbf{p}_{i,[(i+j-1)\bmod n]+1}=\mathbf{A}_{i,[(i+j-1)\bmod n]+1}\mathbf{x}_i=(\mathbf{F}_i)_j,\tag{75}$$

for any $j\in[n]$, where $(\mathbf{F}_i)_j$ is the j-th column of the matrix $\mathbf{F}_i$. This indicates that

$$\mathbf{F}_i=[\mathbf{p}_{i,i+1}\ \ldots\ \mathbf{p}_{i,n}\ \mathbf{p}_{i,1}\ \ldots\ \mathbf{p}_{i,i-1}]\quad\forall i\in[n].\tag{76}$$

Next, we construct $\{\mathbf{B}_{i,j}\}_{i\neq j\in[n]}$ of dimension $p\times\frac{m}{k}$. Choose a $p\times\frac{(n-1)m}{k}$ matrix $\mathbf{V}$ over $\mathbb{F}_q$ such that arbitrary selection of p columns of $\mathbf{V}$ form an invertible matrix. For example, $\mathbf{V}$ can be a Vandermonde matrix subject to $q\geq\frac{(n-1)}{k}m$. We then let

$$\mathbf{B}_{[(i+j-1)\bmod n]+1,j}=[(\mathbf{V})_{(i-1)\frac{m}{k}+1}\ (\mathbf{V})_{(i-1)\frac{m}{k}+2}\cdots(\mathbf{V})_{i\frac{m}{k}}], \quad (77)$$

for any $i\in[n-1]$ and $j\in[n]$, which implies that

$$\mathbf{V}=[\mathbf{B}_{\underline{j}_n+1,j}\ \mathbf{B}_{\underline{j+1}_n+1,j}\cdots\mathbf{B}_{\underline{j+(n-2)}_n+1,j}], \quad (78)$$

where $\underline{j}_n\triangleq(j\bmod n)$ for any integer j. Note that the right-hand-side of (78) remains constant regardless of $j\in[n]$. Thus, we can obtain from (11) that

$$\mathbf{p}_j=\mathbf{V}\left[\mathbf{p}^{\mathsf{T}}_{\underline{j}_n+1,j}\quad\mathbf{p}^{\mathsf{T}}_{\underline{j+1}_n+1,j}\quad\cdots\quad\mathbf{p}^{\mathsf{T}}_{\underline{j+(n-2)}_n+1,j}\right]^{\mathsf{T}}. \quad (79)$$

We now prove the code so constructed is an MR-MUB code.

Theorem 8. *$\mathcal{C}_O$ is an $(n,k,m\mathbf{1})$ MR-MUB code.*

Proof. The proof requires verifying two properties, which are *i*) $\mathcal{C}_{\mathrm{O}}$ being an $(n,k,m\mathbf{1})$ array code, and *ii*) $\mathcal{C}_{\mathrm{O}}$ achieving $R_{\min}$ and $\gamma_{\min}$.

First, we justify *i*), i.e., $\mathcal{C}_{\mathrm{O}}$ satisfying that given any set $\mathcal{E}\subset[n]$ with $|\mathcal{E}|=n-k$, the codeword $\mathbf{C}$ of $\mathcal{C}_{\mathrm{O}}$ can be reconstructed from $\mathbf{C}_{\bar{\mathcal{E}}}$. When $\mathbf{C}_{\bar{\mathcal{E}}}$ is given, both $\mathbf{X}_{\bar{\mathcal{E}}}$ and $\mathbf{P}_{\bar{\mathcal{E}}}$ are known, and so are $\{\mathbf{p}_{\bar{e}_i,\bar{e}_j}\}_{i\neq j\in[k]}$ according to (9). We can then establish from (11) that

$$\begin{aligned}\mathbf{p}_{\bar{e}_j}-\sum_{i=1,i\neq j}^{k}\mathbf{B}_{\bar{e}_i,\bar{e}_j}\mathbf{p}_{\bar{e}_i,\bar{e}_j}&=\sum_{i=1}^{n-k}\mathbf{B}_{e_i,\bar{e}_j}\mathbf{p}_{e_i,\bar{e}_j}\\&=\left[\mathbf{B}_{e_1,\bar{e}_j}\quad\cdots\quad\mathbf{B}_{e_{n-k},\bar{e}_j}\right]\begin{bmatrix}\mathbf{p}_{e_1,\bar{e}_j}\\\vdots\\\mathbf{p}_{e_{n-k},\bar{e}_j}\end{bmatrix}\quad\forall j\in[k],\end{aligned} \quad (80)$$

Since $\mathbf{p}_{\bar{e}_j}-\sum_{i=1,i\neq j}^{k}\mathbf{B}_{\bar{e}_i,\bar{e}_j}\mathbf{p}_{\bar{e}_i,\bar{e}_j}$ is known and any p columns of $\mathbf{V}$, as defined in (78), forms an invertible matrix, we can obtain $\{\mathbf{p}_{e_i,\bar{e}_j}\}_{i\in[n-k],j\in[k]}$ by left-multiplying (80) by $[\mathbf{B}_{e_1,\bar{e}_j}\ \ldots\ \mathbf{B}_{e_{n-k},\bar{e}_j}]^{-1}$. With the knowledge of k columns $\{\mathbf{p}_{e_i,\bar{e}_j}\}_{j\in[k]}$ of $\mathbf{F}_{e_i}$ in (76), we can recover $\mathbf{x}_{e_i}$ via the decoding algorithm of the $(n-1,k)$ MDS array code $\mathcal{M}$. By this procedure, $\{\mathbf{x}_i\}_{i\in[n]}$ can all be recovered.

Next, we verify *ii*). From (76), we have $\mathbf{p}_{i,j}\in\mathbb{F}_q^{\frac{m}{k}}$ and hence $\gamma_{i,j}=\frac{m}{k}$, which leads to $\gamma=\gamma_{\min}$ as pointed out in (68). In addition, (79) shows $p_j=\frac{(n-k)}{k}m$ for $j\in[n]$, and hence $R_{\min}$ is achieved as addressed in (67). The justification of the two required properties of $\mathcal{C}_{\mathrm{O}}$ is thus completed. □

The $(4,2,2\,\mathbf{1})$ MR-MUB code in Fig. 2(b) can be constructed via the proposed procedure. First, with $\mathbf{x}_i=[x_{i,1}\ x_{i,2}]^{\mathsf{T}}$, $\mathcal{M}$ is chosen as a $(3,2)$ parity-check code over $\mathbb{F}_q$, which gives

$$\mathbf{F}_i=\left[x_{i,1}\quad x_{i,2}\quad x_{i,1}+x_{i,2}\right]\quad\forall i\in[n]. \quad (81)$$

Thus, from (76), we have $\mathbf{p}_{1,2}=x_{1,1}$, $\mathbf{p}_{1,3}=x_{1,2}$, $\mathbf{p}_{1,4}=x_{1,1}+x_{1,2}$. The remaining $\mathbf{p}_{i,j}$ can be similarly obtained and are listed in Table II.

Next, we specify

$$\mathbf{V}=\begin{bmatrix}0&1&1\\1&1&0\end{bmatrix}, \quad (82)$$

which satisfies that the selection of any two columns forms an invertible matrix. By (79), we have

$$\begin{aligned}\mathbf{p}_1&=\mathbf{V}\,[\mathbf{p}_{2,1}^{\mathsf{T}}\ \mathbf{p}_{3,1}^{\mathsf{T}}\ \mathbf{p}_{4,1}^{\mathsf{T}}]^{\mathsf{T}}\\&=\begin{bmatrix}0&1&1\\1&1&0\end{bmatrix}\begin{bmatrix}x_{2,1}+x_{2,2}\\x_{3,2}\\x_{4,1}\end{bmatrix}=\begin{bmatrix}x_{3,2}+x_{4,1}\\x_{2,1}+x_{2,2}+x_{3,2}\end{bmatrix}.\end{aligned} \quad (83)$$

$\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$ can be similarly obtained and can be found in Fig. 2(b).

We now demonstrate via this example how erased nodes can be systematically recovered based on the chosen $\mathcal{M}$ and $\mathbf{V}$. Suppose nodes 1 and 2 are erased. As knowing from (79) that

$$\mathbf{p}_3=\begin{bmatrix}0&1&1\\1&1&0\end{bmatrix}\begin{bmatrix}\mathbf{p}_{4,3}\\\mathbf{p}_{1,3}\\\mathbf{p}_{2,3}\end{bmatrix}, \quad (84)$$

we perform (80) to obtain

$$\mathbf{p}_3-\begin{bmatrix}0\\1\end{bmatrix}\mathbf{p}_{4,3}=\begin{bmatrix}1&1\\1&0\end{bmatrix}\begin{bmatrix}\mathbf{p}_{1,3}\\\mathbf{p}_{2,3}\end{bmatrix}. \quad (85)$$

Since $\mathbf{p}_3$ is known and $\mathbf{p}_{4,3}$ can be obtained from $\mathbf{x}_4$ via $\mathbf{p}_{4,3}=\mathbf{A}_{4,3}\mathbf{x}_4$, we can recover $\mathbf{p}_{1,3}$ and $\mathbf{p}_{2,3}$ via

$$\begin{bmatrix}\mathbf{p}_{1,3}\\\mathbf{p}_{2,3}\end{bmatrix}=\begin{bmatrix}1&1\\1&0\end{bmatrix}^{-1}\left(\mathbf{p}_3-\begin{bmatrix}0\\1\end{bmatrix}\mathbf{p}_{4,3}\right). \quad (86)$$

The recovery of $\mathbf{p}_{1,4}$ and $\mathbf{p}_{2,4}$ can be similarly done via

$$\begin{bmatrix}\mathbf{p}_{1,4}\\\mathbf{p}_{2,4}\end{bmatrix}=\begin{bmatrix}0&1\\1&1\end{bmatrix}^{-1}\left(\mathbf{p}_4-\begin{bmatrix}1\\0\end{bmatrix}\mathbf{p}_{3,4}\right). \quad (87)$$

We then note from (76) that $\mathbf{F}_1=\mathcal{F}(\mathbf{x}_1)=[\mathbf{p}_{1,2}\ \mathbf{p}_{1,3}\ \mathbf{p}_{1,4}]$ is a codeword of $\mathcal{M}$, corresponding to $\mathbf{x}_1$, and its second and third columns are just recovered via (86) and (87). By equating the second and the third columns of $\mathbf{F}_1$ with (81), the recovery of $\mathbf{x}_1$ is done. We can similarly recover $\mathbf{x}_2$ by using the recovered $\mathbf{p}_{2,3}$ and $\mathbf{p}_{2,4}$ in (86) and (87). The recovery of the two erased nodes is thus completed.

C. Construction of MUB codes with the smallest code redundancy

We continue to propose a construction of $(n,k,\mathbf{m})$ MUB codes with the smallest code redundancy, and denote the code to be constructed as $\mathcal{C}_{\mathrm{U}}$ for notational convenience. This can be considered a generalization of the code construction in the previous subsection.

TABLE II
$\{\mathbf{p}_{i,j}\}_{i\neq j\in[n]}$ OF THE MR-MUB CODE PRESENTED IN FIG. 2(B), WHERE THE ELEMENT IN THE i-TH ROW AND THE j-TH COLUMN IS $\mathbf{p}_{i,j}$.

null	$x_{1,1}$	$x_{1,2}$	$x_{1,1}+x_{1,2}$
$x_{2,1}+x_{2,2}$	null	$x_{2,1}$	$x_{2,2}$
$x_{3,2}$	$x_{3,1}+x_{3,2}$	null	$x_{3,1}$
$x_{4,1}$	$x_{4,2}$	$x_{4,1}+x_{4,2}$	null

For the construction of $\mathcal{C}_{\mathrm{U}}$, we require $\mathbf{x}_i \in \mathbb{F}_q^{m_i}$ and $\mathbf{p}_j \in \mathbb{F}_q^{p_j}$ with $\{p_j\}_{j\in[n]}$ specified in (48) for $i, j \in [n]$. The construction of $\{\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$ and $\{\mathbf{B}_{i,j}\}_{i\neq j\in[n]}$ associated with $\mathcal{C}_{\mathrm{U}}$ are then addressed as follows.

First, for $i \neq j \in [n]$, we construct $\mathbf{A}_{i,j}$ of dimension $\frac{m_i}{k} \times m_i$. For each $i \in [n]$, choose an $(n-1, k)$ MDS array code $\mathcal{M}_i$ over $\mathbb{F}_q$ with encoding function $\mathcal{F}_i : \mathbb{F}_q^{m_i} \longrightarrow \mathbb{F}_q^{\frac{m_i}{k}\times(n-1)}$, where $\frac{m_i}{k}$ is the number of rows of the MDS array code, and m_i is the number of all data symbols of $\mathcal{M}_i$. Denote $\mathbf{F}_i \triangleq \mathcal{F}_i(\mathbf{x}_i)$. Then, $\{\mathbf{p}_{i,j}\}_{i\neq j\in[n]}$ defined in (9), as well as $\{\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$, can be characterized via

$$\mathbf{p}_{i,[(i+j-1) \bmod n]+1} = \mathbf{A}_{i,[(i+j-1) \bmod n]+1}\mathbf{x}_i = (\mathbf{F}_i)_j \quad (88)$$

for any $j \in [n]$. This indicates that

$$\mathbf{F}_i = [\mathbf{p}_{i,i+1} \ \ldots \ \mathbf{p}_{i,n} \ \mathbf{p}_{i,1} \ \ldots \ \mathbf{p}_{i,i-1}] \quad \forall i \in [n]. \quad (89)$$

Next, for $i \neq j \in [n]$, we construct $\mathbf{B}_{i,j}$ of dimension $p_j \times \frac{m_i}{k}$. Choose a $p_j \times \sum_{i\in[n]\setminus\{j\}} \frac{m_i}{k}$ matrix $\mathbf{V}_j$ over $\mathbb{F}_q$ such that arbitrary selection of p_j columns of $\mathbf{V}_j$ form an invertible matrix. We then get $\{\mathbf{B}_{i,j}\}_{i\neq j\in[n]}$ from

$$\mathbf{V}_j = \begin{bmatrix}\mathbf{B}_{j+1,j} & \ldots & \mathbf{B}_{n,j} & \mathbf{B}_{1,j} & \ldots & \mathbf{B}_{j-1,j}\end{bmatrix} \quad \forall j \in [n]. \quad (90)$$

Thus, we can obtain from (11) that

$$\mathbf{p}_j = \mathbf{V}_j \begin{bmatrix}\mathbf{p}_{j+1,j}^{\mathsf{T}} & \ldots & \mathbf{p}_{n,j}^{\mathsf{T}} & \mathbf{p}_{1,j}^{\mathsf{T}} & \ldots & \mathbf{p}_{j-1,j}^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}}. \quad (91)$$

We now prove the code so constructed is an MUB code with the smallest code redundancy.

Theorem 9. *$\mathcal{C}_U$ is an $(n, k, \mathbf{m})$ MUB code with the smallest code redundancy.*

Proof. For better readability, the proof is relegated to Appendix C. □

We demonstrate that the $(4, 2, \mathbf{m} = [4\ 2\ 2\ 0]^{\mathsf{T}})$ MUB code in Fig. 3 can be constructed via the proposed procedure. First, with $\mathbf{x}_1 = [x_{1,1} \ldots x_{1,4}]^{\mathsf{T}}$, $\mathcal{M}_1$ is chosen as a $(3, 2)$ MDS array code, which encodes $\mathbf{x}_1$ into

$$\mathbf{F}_1 = \begin{bmatrix} x_{1,1} & x_{1,3} & x_{1,1}+x_{1,3} \\ x_{1,2} & x_{1,4} & x_{1,2}+x_{1,4}\end{bmatrix} = \begin{bmatrix}\mathbf{p}_{1,2} & \mathbf{p}_{1,3} & \mathbf{p}_{1,4}\end{bmatrix}. \quad (92)$$

For $i = 2$ and 3, $\mathcal{M}_i$ is chosen to be a $(3, 2)$ parity check code over $\mathbb{F}_q$, as the one in (81). Since $m_4 = 0$, $\{\mathbf{p}_{4,j}\}_{j\in[4]\setminus\{4\}}$ are null vectors. The resulting $\{\mathbf{p}_{i,j}\}_{i\neq j\in[n]}$ are listed in Table III.

TABLE III
$\{\mathbf{p}_{i,j}\}_{i\neq j\in[n]}$ OF THE MUB CODE PRESENTED IN FIG. 3, WHERE THE ELEMENT IN THE i-TH ROW AND THE j-TH COLUMN IS $\mathbf{p}_{i,j}$.

null	$\begin{bmatrix} x_{1,1} \\ x_{1,2}\end{bmatrix}$	$\begin{bmatrix} x_{1,3} \\ x_{1,4}\end{bmatrix}$	$\begin{bmatrix} x_{1,1}+x_{1,3} \\ x_{1,2}+x_{1,4}\end{bmatrix}$
$x_{2,1}+x_{2,2}$	null	$x_{2,1}$	$x_{2,2}$
$x_{3,2}$	$x_{3,1}+x_{3,2}$	null	$x_{3,1}$
null	null	null	null

Next, we obtain from (48) that $p_1 = 2$ and $p_2 = p_3 = p_4 = 3$, and specify

$$\mathbf{V}_1 = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\ \mathbf{V}_2 = \mathbf{V}_3 = \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}, \quad \text{and} \quad \mathbf{V}_4 = \begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix}, \quad (93)$$

where the selection of any p_i columns from $\mathbf{V}_i$ forms an invertible matrix. By (91) and Table III, we have

$$\mathbf{p}_1 = \begin{bmatrix}x_{2,1}+x_{2,2}\\ x_{3,2}\end{bmatrix}, \quad \mathbf{p}_2 = \begin{bmatrix}x_{1,1}\\ x_{1,2}\\ x_{3,1}+x_{3,2}\end{bmatrix}, \quad \mathbf{p}_3 = \begin{bmatrix}x_{1,3}\\ x_{1,4}\\ x_{2,1}\end{bmatrix}, \quad \text{and} \quad \mathbf{p}_4 = \begin{bmatrix}x_{1,1}+x_{1,3}+x_{3,1}\\ x_{1,2}+x_{1,4}+x_{3,1}\\ x_{2,2}+x_{3,1}\end{bmatrix}, \quad (94)$$

as presented in Fig. 3.

Based on this example, the systematic recovery of erased nodes can be demonstrated as follows. Suppose nodes 1 and 2 are erased. Then, through (91) and (93), we have

$$\mathbf{p}_3 = \begin{bmatrix}\mathbf{p}_{1,3}\\ \mathbf{p}_{2,3}\end{bmatrix}, \text{ and } \mathbf{p}_4 - \begin{bmatrix}1\\1\\1\end{bmatrix} \mathbf{p}_{3,4} = \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} \begin{bmatrix}\mathbf{p}_{1,4}\\ \mathbf{p}_{2,4}\end{bmatrix}. \quad (95)$$

We can thus obtain $\mathbf{p}_{1,3}$, $\mathbf{p}_{1,4}$, $\mathbf{p}_{2,3}$ and $\mathbf{p}_{2,4}$. By noting $\mathbf{p}_{1,3} = [x_{1,3} \quad x_{1,4}]^{\mathsf{T}}$ and $\mathbf{p}_{1,4} = [x_{1,1}+x_{1,3} \quad x_{1,2}+x_{1,4}]^{\mathsf{T}}$, the recovery of $\mathbf{x}_1$ is done via the erasure correcting of $\mathcal{M}_1$. We can similarly recover $\mathbf{x}_2$ from $\mathbf{p}_{2,3}$ and $\mathbf{p}_{2,4}$. The recovery of the two erased nodes is therefore completed.

VI. UPDATE COMPLEXITY OF MR-MUB CODES

The update complexity of an array code, denoted as θ, is defined as the average number of parity symbols affected by updating a single data symbol [13]. For an $(n, k, \mathbf{m})$ irregular array code, a definition-implied lower bound for update complexity is $\theta \geq n - k$. This lower bound can be easily justified by contradiction. If $\theta < n - k$, then at most $(n - k - 1)$ nodes are affected when updating a data symbol, which leads to a contradiction that this data symbol cannot be reconstructed by the remaining k unaffected nodes.

Previous results on update complexity indicate that the lower bound $n-k$ is not attainable by (n, k) binary horizontal MDS array codes with $1 < k < n-1$ [13], while existence of $(n, k = n-2)$ binary vertical MDS array codes that achieve the lower bound $\theta = n-k = 2$ has been confirmed [15], [30]. Then, we consider whether or not the update complexity of MR-MUB codes can reach the definition-implied lower bound. Unfortunately, we found the answer is negative under $k > 1$, and will show in Theorem 10 that the update complexity of MR-MUB codes is lower-bounded by $n - k + \frac{k-1}{k}$.

In order to facilitate the presentation of the result in Theorem 10, five lemmas are addressed first. The first lemma indicates it suffices to consider the MR-MUB codes with $\{\mathbf{M}_{i,i}\}_{i\in[n]}$ being zero matrices; hence, we do not need to consider $\{\mathbf{M}_{i,i}\}_{i\in[n]}$ in the calculation of update complexity (cf. Lemma 6). The second lemma shows that for the determination of a lower bound of update complexity, we can focus on the decomposition of $\mathbf{M}_{i,j} = \mathbf{B}'_{i,j}\mathbf{A}'_{i,j}$ with $\mathbf{A}'_{i,j}$ containing an $\gamma_{i,j}\times\gamma_{i,j}$ identity submatrix. As a result, $\mathbf{B}'_{i,j}$ is a submatrix of $\mathbf{M}_{i,j}$ and the column weights of $\mathbf{M}_{i,j}$ are lower-bounded by the column weights of $\mathbf{B}'_{i,j}$ (cf. Lemma 7). The next two lemmas then study the column weights of general $\mathbf{B}_{i,j}$ that is not necessarily a submatrix of $\mathbf{M}_{i,j}$ (cf. Lemmas 8 and 9). The last lemma accounts for the number of non-zero columns in $\mathbf{M}_i^{(\ell)} \triangleq [(\mathbf{M}_{i,1})_\ell \ \ldots (\mathbf{M}_{i,i-1})_\ell \quad (\mathbf{M}_{i,i+1})_\ell \cdots \ (\mathbf{M}_{i,n})_\ell]$, where $(\mathbf{M}_{i,j})_\ell$ denotes the ℓ-th column of matrix $\mathbf{M}_{i,j}$.

Lemma 6. *For any $(n,k,\mathbf{m})$ irregular array code $\mathcal{C}$ with construction matrices $\{\mathbf{M}_{i,j}\}_{i,j\in[n]}$, we can construct another $(n,k,\mathbf{m})$ irregular array code $\mathcal{C}'$ with $\{\mathbf{M}'_{i,i}=[\mathbf{0}]\}_{i\in[n]}$ such that both codes have the same code redundancy and update bandwidth.*

Proof. Let the construction matrices of $\mathcal{C}'$ be defined as

$$\mathbf{M}'_{i,j} = \begin{cases}\mathbf{M}_{i,j} & i\neq j;\\ [\mathbf{0}] & i=j.\end{cases} \tag{96}$$

Then, there exists an invertible mapping between codewords of $\mathcal{C}'$ and $\mathcal{C}$, i.e.,

$$\begin{aligned}\mathbf{c}'_j &= \begin{bmatrix}\mathbf{x}_j\\ \mathbf{p}'_j\end{bmatrix} = \begin{bmatrix}\mathbf{x}_j\\ \mathbf{p}_j-\mathbf{M}_{j,j}\mathbf{x}_j\end{bmatrix} = \begin{bmatrix}\mathbf{I} & [\mathbf{0}]\\ -\mathbf{M}_{j,j} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{x}_j\\ \mathbf{p}_j\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{I} & [\mathbf{0}]\\ -\mathbf{M}_{j,j} & \mathbf{I}\end{bmatrix}\mathbf{c}_j \quad \text{for } j\in[n],\end{aligned} \tag{97}$$

where $\mathbf{I}$ denotes an identity matrix of proper size. A consequence of (97) is that all data symbols can be retrieved by accessing any k columns of the corresponding codeword of $\mathcal{C}$ if, and only if, the same can be done by accessing any k columns of the corresponding codeword of $\mathcal{C}'$. As $\mathcal{C}$ is an $(n,k,\mathbf{m})$ irregular array code, we confirm that $\mathcal{C}'$ is also an $(n,k,\mathbf{m})$ irregular array code. Since $\gamma'_{i,j} = \mathsf{rank}(\mathbf{M}'_{i,j}) = \mathsf{rank}(\mathbf{M}_{i,j}) = \gamma_{i,j}$ with $i\neq j\in[n]$, the update bandwidth of $\mathcal{C}'$ remains the same as that of $\mathcal{C}$ according to (8). The relation of $\mathbf{p}'_j = \mathbf{p}_j - \mathbf{M}_{j,j}\mathbf{x}_j$ indicates $p'_j = \mathsf{row}(\mathbf{p}'_j) = \mathsf{row}(\mathbf{p}_j) = p_j$ for $j\in[n]$, confirming $\mathcal{C}'$ and $\mathcal{C}$ have the same code redundancy. The lemma is therefore substantiated. □

For an $(n,k,\mathbf{m})$ irregular array codes, the number of symbols affected by the update of the ℓ-th symbol in $\mathbf{x}_i$ is

$$\theta_i^{(\ell)} = \sum_{j\in[n]\setminus i} \mathsf{wt}((\mathbf{M}_{i,j})_\ell), \tag{98}$$

and we can now omit $\mathbf{M}_{i,i}$ due to Lemma 6. The update complexity θ of an irregular array code is therefore given by

$$\theta = \frac{1}{B}\sum_{i\in[n]}\sum_{\ell\in[m_i]}\theta_i^{(\ell)}. \tag{99}$$

Since the update complexity is only related to the column weights of construction matrices, the next lemma provides a structure to be considered in the calculation of θ in (99).

Lemma 7. *There exists a full rank decomposition of construction matrix $\mathbf{M}_{i,j} = \mathbf{B}'_{i,j}\mathbf{A}'_{i,j}$ such that $\mathbf{A}'_{i,j}$ contains a $\gamma_{i,j}\times\gamma_{i,j}$ identity submatrix.*

Proof. The existence of a full rank decomposition $\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}$ has been confirmed in Section II-C. As $\mathbf{A}_{i,j}$ is with full row rank, there exists an invertible matrix $\mathbf{R}_{i,j}$ such that $\mathbf{A}'_{i,j} = \mathbf{R}_{i,j}\mathbf{A}_{i,j}$, where $\mathbf{A}'_{i,j}$ contains a $\gamma_{i,j}\times\gamma_{i,j}$ identity submatrix. We can then obtain a new full rank decomposition $\mathbf{M}_{i,j} = \mathbf{B}'_{i,j}\mathbf{A}'_{i,j}$ with $\mathbf{B}'_{i,j} = \mathbf{B}_{i,j}\mathbf{R}_{i,j}^{-1}$. □

When $\mathbf{A}'_{i,j}$ contains a $\gamma_{i,j}\times\gamma_{i,j}$ identity submatrix, $\mathbf{B}'_{i,j}$ must be a submatrix of $\mathbf{M}_{i,j}$. Thus, the column weights of $\mathbf{M}_{i,j}$ are lower-bounded by the column weights of $\mathbf{B}'_{i,j}$.

This brings up the study of the next two lemmas, which hold not just for a submatrix $\mathbf{B}'_{i,j}$ of $\mathbf{M}_{i,j}$ but for general full-rank decomposition $\mathbf{B}_{i,j}$.

Lemma 8. *Given an $(n,k,m\mathbf{1})$ MR-MUB code with construction matrices $\{\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$, $\mathbf{B}_{\mathcal{E},j} \triangleq [\mathbf{B}_{e_1,j}\ \ldots\ \mathbf{B}_{e_{n-k},j}]$ is an invertible matrix for every $\mathcal{E}\subset[n]$ with $|\mathcal{E}| = n-k$ and for every $j\notin\mathcal{E}$.*

Proof. First, we note respectively from (68) and (67) that $\gamma_{i,j} = \frac{m}{k}$ and $p_j = \frac{(n-k)m}{k}$. Hence, $\mathbf{B}_{e_i,j}$ is an $\frac{(n-k)m}{k}\times\frac{m}{k}$ matrix, implying $\mathbf{B}_{\mathcal{E},j}$ is an $\frac{(n-k)m}{k}\times\frac{(n-k)m}{k}$ square matrix. We then prove the lemma by contradiction.

Suppose $\mathbf{B}_{\mathcal{E},j}$ is not invertible for some $\mathcal{E}$ with $|\mathcal{E}| = n-k$ and some $j\notin\mathcal{E}$. Then, $\mathsf{rank}(\mathbf{B}_{\mathcal{E},j}) < \frac{(n-k)m}{k}$. According to (9) and (11), we have

$$\begin{aligned}\mathbf{p}_j &= \sum_{\ell\in\bar{\mathcal{E}}}\mathbf{B}_{\ell,j}\mathbf{p}_{\ell,j} + \sum_{i\in[n-k]}\mathbf{B}_{e_i,j}\mathbf{p}_{e_i,j} && (100)\\ &= \sum_{\ell\in\bar{\mathcal{E}}}\mathbf{B}_{\ell,j}\mathbf{A}_{\ell,j}\mathbf{x}_\ell + \sum_{i\in[n-k]}\mathbf{B}_{e_i,j}\mathbf{p}_{e_i,j} && (101)\\ &= [\mathbf{B}_{\bar{e}_1,j}\mathbf{A}_{\bar{e}_1,j}\ \cdots\ \mathbf{B}_{\bar{e}_k,j}\mathbf{A}_{\bar{e}_k,j}]\mathbf{X}_{\bar{\mathcal{E}}} + \mathbf{B}_{\mathcal{E},j}\mathbf{p}_{\mathcal{E},j}, && (102)\end{aligned}$$

where $\mathbf{p}_{\mathcal{E},j} \triangleq [\mathbf{p}_{e_1,j}^\mathsf{T}\ \ldots\ \mathbf{p}_{e_{n-k},j}^\mathsf{T}]^\mathsf{T}$. This implies

$$\begin{aligned}H(\mathbf{p}_j\mid\mathbf{X}_{\bar{\mathcal{E}}}) &= H(\mathbf{B}_{\mathcal{E},j}\mathbf{p}_{\mathcal{E},j}\mid\mathbf{X}_{\bar{\mathcal{E}}})\\ &\leq H(\mathbf{B}_{\mathcal{E},j}\mathbf{p}_{\mathcal{E},j}) \leq \mathsf{rank}(\mathbf{B}_{\mathcal{E},j}),\end{aligned} \tag{103}$$

where the last inequality follows from Lemma 1. We then derive based on (17) that

$$\begin{aligned}I(\mathbf{C}_{\bar{\mathcal{E}}};\mathbf{X}_{\mathcal{E}}) &= I(\mathbf{X}_{\mathcal{E}};\mathbf{P}_{\bar{\mathcal{E}}}\mid\mathbf{X}_{\bar{\mathcal{E}}}) = H(\mathbf{P}_{\bar{\mathcal{E}}}\mid\mathbf{X}_{\bar{\mathcal{E}}}) && (104)\\ &\leq \sum_{\ell\in\bar{\mathcal{E}}\setminus\{j\}} H(\mathbf{p}_\ell\mid\mathbf{X}_{\bar{\mathcal{E}}}) + H(\mathbf{p}_j\mid\mathbf{X}_{\bar{\mathcal{E}}}) && (105)\\ &\leq \sum_{\ell\in\bar{\mathcal{E}}\setminus\{j\}} H(\mathbf{p}_\ell) + H(\mathbf{p}_j\mid\mathbf{X}_{\bar{\mathcal{E}}}) && (106)\\ &\leq \sum_{\ell\in\bar{\mathcal{E}}\setminus\{j\}} p_\ell + \mathsf{rank}(\mathbf{B}_{\mathcal{E},j}) && (107)\\ &< (n-k)m = H(\mathbf{X}_{\mathcal{E}}), && (108)\end{aligned}$$

where the last strict inequality holds due to $\mathsf{rank}(\mathbf{B}_{\mathcal{E},j}) < \frac{(n-k)m}{k}$. The derivation in (108) indicates that $\mathbf{X}_{\mathcal{E}}$ cannot be reconstructed from $\mathcal{C}_{\bar{\mathcal{E}}}$, leading to a contradiction to the definition of $(n,k,m\mathbf{1})$ array codes. □

Lemma 9. *For an $(n,k,m\mathbf{1})$ MR-MUB code with construction matrices $\{\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{A}_{i,j}\}_{i\neq j\in[n]}$,*

$$\mathbf{B}_j \triangleq [\mathbf{B}_{1,j}\ \ldots\mathbf{B}_{j-1,j}\ \mathbf{B}_{j+1,j}\ \ldots\mathbf{B}_{n,j}] \tag{109}$$

must contain at least $\frac{(k-1)m}{k}$ columns whose weight is no less than 2.

Proof. Lemma 8 shows $\mathbf{B}_{\mathcal{E},j}$ is invertible for arbitrary $\mathcal{E}$ with $|\mathcal{E}| = n-k$; hence, $\mathbf{B}_j$ in (109) contains no zero column, and also has no identical columns. As each column of $\mathbf{B}_j$ consists of $\frac{(n-k)m}{k}$ components, the number of weight-one columns of $\mathbf{B}_j$ must be at most $\frac{(n-k)m}{k}$. We thus conclude that there are at least

$$\mathsf{col}(\mathbf{B}_j) - \frac{(n-k)m}{k} = \frac{(n-1)m}{k} - \frac{(n-k)m}{k} = \frac{(k-1)m}{k} \tag{110}$$

columns of $\mathbf{B}_j$ with weights no less than 2. This completes the proof. □

As previously mentioned, since the above two lemmas hold for the full-rank submatrix $\mathbf{B}'_{i,j}$ of $\mathbf{M}_{i,j}$, a lower bound of update complexity can thus be established.

Corollary 3. *The construction matrices $\{\mathbf{M}_{i,j}\}_{i\neq j\in[n]}$ of an $(n, k, m\mathbf{1})$ MR-MUB code must have at least $\frac{(k-1)mn}{k}$ columns with weights no less than 2.*

Proof. Lemma 9 holds for those full-rank submatrices $\{\mathbf{B}'_{i,j}\}_{i\neq j\in[n]}$ of $\{\mathbf{M}_{i,j}\}_{i\neq j\in[n]}$. Thus, there are at least $\frac{(k-1)m}{k}$ columns in $[\mathbf{M}_{1,j} \ \ldots \mathbf{M}_{j-1,j} \ \ \mathbf{M}_{j+1,j} \ \ldots \mathbf{M}_{n,j}]$, which have weights larger than 1. Consequently, the number of columns with weights no less than 2 in $\{\mathbf{M}_{i,j}\}_{i\neq j\in[n]}$ is at least $\frac{(k-1)mn}{k}$. □

Lemma 10. *Fix an $(n, k, m\mathbf{1})$ MR-MUB code. For every $i \in [n]$ and every $\ell \in [m]$, there are at least $n-k$ columns with non-zero weights in $\mathbf{M}_i^{(\ell)} \triangleq [(\mathbf{M}_{i,1})_\ell \ \ldots (\mathbf{M}_{i,i-1})_\ell \ \ (\mathbf{M}_{i,i+1})_\ell \ldots \ (\mathbf{M}_{i,n})_\ell]$.*

Proof. Denote the data symbol in the ℓ-th row of $\mathbf{x}_i$ as $x_{i,\ell}$. If there were k zero columns in $\mathbf{M}_i^{(\ell)}$, then we can find k parity vectors that are functionally independent of $x_{i,\ell}$ according to (6), which implies we can find k nodes that cannot be used to reconstruct $x_{i,\ell}$. A contradiction to the definition of $(n, k, m\mathbf{1})$ MR MUB codes is obtained. □

Considering Lemma 10 holds for every $i \in [n]$ and $\ell \in [m]$, an immediate consequence is summarized in the next corollary.

Corollary 4. *For an $(n, k, m\mathbf{1})$ MR-MUB code, there are at least $(n-k)mn$ columns with nonzero weights in all construction matrices $\{\mathbf{M}_{i,j}\}_{i\neq j\in[n]}$.*

Corollaries 3 and 4 then lead to the main result in this section.

Theorem 10. *The update complexity θ of $(n, k, m\mathbf{1})$ MR-MUB codes is lower-bounded by $n-k+\frac{k-1}{k}$.*

Proof. Denote by $\theta(\ell)$ the number of columns exactly with weight ℓ in all construction matrices $\{\mathbf{M}_{i,j}\}_{i\neq j\in[n]}$. Since $\mathsf{row}(\mathbf{M}_{i,j}) = \frac{(n-k)m}{k}$, it is obvious that $\theta(\ell) = 0$ for $\ell > \frac{(n-k)m}{k}$. Let $\Theta(\ell) \triangleq \sum_{i=\ell}^{\frac{(n-k)m}{k}} \theta(i)$. We then derive from (99) that

$$\begin{aligned}\theta &= \frac{1}{nm}\sum_{\ell\in[\frac{(n-k)m}{k}]} \ell\cdot\theta(\ell) = \frac{1}{nm}\sum_{\ell\in[\frac{(n-k)m}{k}]}\Theta(\ell) \\ &\geq \frac{\Theta(1)+\Theta(2)}{nm}.\end{aligned} \tag{111}$$

As Corollaries 3 and 4 imply $\Theta(2) \geq \frac{(k-1)mn}{k}$ and $\Theta(1) \geq (n-k)mn$, respectively, (111) indicates that $\theta \geq n-k+\frac{k-1}{k}$. □

VII. A class of MR-MUB codes with the optimal repair bandwidth

A. Generic transformation for code construction

Consider (n, k) MDS regular array codes with each node having exactly the same number of symbols, denoted as α. Hence, $m_i + p_i = \alpha$ for every $i \in [n]$. Let β_i be the amount of symbols that needs to be downloaded from all other $n-1$ nodes when repairing node i. Then, it is known [3] that for all (n, k) MDS regular array code designs, $\beta_i \geq \frac{(n-1)\alpha}{(n-k)}$ for every $i \in [n]$. As a consequence of this universal lower bound for every β_i, an (n, k) MDS regular array code is said to be with the optimal repair bandwidth for all nodes if $\beta_i = \frac{(n-1)\alpha}{(n-k)}$ for every $i \in [n]$.

In 2018, Li et al. [28] proposed a generic transformation that converts a nonbinary (n, k) MDS regular array code with node size α into another (n, k) MDS regular array code with node size $\alpha' = (n-k)\alpha$ over the same field $\mathbb{F}_q$ such that 1) some chosen $(n-k)$ nodes have the optimal repair bandwidth $\frac{(n-1)\alpha'}{(n-k)} = (n-1)\alpha$, and 2) the normalized repair bandwidth[5] of the remaining k nodes are preserved. Additionally, after applying the transformation $\lceil\frac{n}{n-k}\rceil$ times, a nonbinary (n, k) MDS regular array code can be converted into an (n, k) MDS regular array code with all nodes achieving the optimal repair bandwidth.

In this section, using the transformation in [28], an $(n, n-2, 2^{\lceil\frac{n}{n-k}\rceil}m\mathbf{1})$ regular array code that achieves the optimal repair bandwidth for all nodes is constructed from an $(n, n-2, m\mathbf{1})$ MR-MUB code under $k \mid m$. We will then prove in Theorem 12 that the transformed $(n, n-2, 2^{\lceil\frac{n}{n-k}\rceil}m\mathbf{1})$ regular array code also have the minimum code redundancy and the minimum update bandwidth and hence is an MR-MUB code.

For completeness, we restate the generic transform in [28] in the form that is necessary in this paper in the following theorem. Similar to [28], the symbols of the codes we construct are over $\mathbb{F}_q$ with $q > 2$, where the elements of $\mathbb{F}_q$ are denoted as $\{0, 1, \mathrm{g}, \ldots, \mathrm{g}^{q-2}\}$ and g is a primitive element of $\mathbb{F}_q$.

Theorem 11. *(Generic transform for $(n = k+2, k)$ regular array codes [28]) Let $\mathbf{C}^{(0)} \triangleq [\mathbf{c}_1^{(0)} \ldots \mathbf{c}_n^{(0)}]$ and $\mathbf{C}^{(1)} \triangleq [\mathbf{c}_1^{(1)} \ldots \mathbf{c}_n^{(1)}]$ be codewords of a nonbinary $(n = k+2, k)$*

[5]The normalized repair bandwidth for a node is defined as

$$\frac{\text{the number of symbols downloaded for repairing this node}}{\text{the number of symbols repaired}}. \tag{112}$$

MDS regular array code $\mathcal{C}$ over $\mathbb{F}_q$ with node size α, where the data symbols used to generate $\mathbf{C}^{(0)}$ and $\mathbf{C}^{(1)}$ can be different. Denote by β_i the repair bandwidth of $\mathcal{C}$ for node i. Then,

$$\mathbf{C}' = \begin{bmatrix} \mathbf{c}_1^{(0)} & \cdots & \mathbf{c}_k^{(0)} & \mathbf{c}_{k+1}^{(0)} & \mathbf{c}_{k+2}^{(0)} + \mathbf{c}_{k+2}^{(1)} \\ \mathbf{c}_1^{(1)} & \cdots & \mathbf{c}_k^{(1)} & \mathbf{c}_{k+2}^{(0)} + \mathrm{g}\,\mathbf{c}_{k+2}^{(1)} & \mathbf{c}_{k+1}^{(1)} \end{bmatrix} \in \mathbb{F}_q^{2\alpha\times(k+2)}, \tag{113}$$

are codewords of an $(n = k+2, k)$ MDS regular array code $\mathcal{C}'$ with node size $\alpha' = 2\alpha$, and its repair bandwidth for node i satisfies

$$\beta'_i = \begin{cases} 2\beta_i, & \text{for } i \in [k] \\ \frac{(n-1)\alpha'}{n-k} = (n-1)\alpha, & \text{for } k < i \le n = k+2. \end{cases} \tag{114}$$

It is worth noting that the last two nodes of the transformed code $\mathcal{C}'$ have achieved the universal lower bound and therefore is with the optimal repair bandwidth. Furthermore, it can be inferred from (114) that if the code $\mathcal{C}$ before transformation is already with the optimal repair bandwidth for every node, then the repair bandwidths of $\mathcal{C}'$ are also optimal for all nodes.

B. MR-MUB code construction with the optimal repair bandwidth

According to Theorem 11, given an $(n, k = n-2, m\mathbf{1})$ MR-MUB code $\mathcal{C}$ under $k \mid m$, we can construct an $(n, n-2, 2m\mathbf{1})$ regular array code $\mathcal{C}'$ that satisfies 1) the last two nodes are with the optimal repair bandwidth, and 2) the remaining k nodes preserve the same normalized repair bandwidths as their corresponding nodes of $\mathcal{C}$. In order to distinguish between the codewords before and after transformation, we will use

$$\left\{\mathbf{y}_i = \begin{bmatrix} \mathbf{y}_i^{(0)} \\ \mathbf{y}_i^{(1)} \end{bmatrix}\right\}_{i\in[n]} \quad \text{and} \quad \left\{\mathbf{q}_i = \begin{bmatrix} \mathbf{q}_i^{(0)} \\ \mathbf{q}_i^{(1)} \end{bmatrix}\right\}_{i\in[n]} \tag{115}$$

to denote the data vectors and the parity vectors of the transformed code $\mathcal{C}'$, respectively. Data vectors and parity vectors of the base code $\mathcal{C}$ are respectively denoted as $\{\mathbf{x}_i^{(\ell)}\}_{i\in[n],\ell\in\{0,1\}}$ and $\{\mathbf{p}_i^{(\ell)}\}_{i\in[n],\ell\in\{0,1\}}$. We then have the following correspondence between $\{\mathbf{y}_i^{(\ell)}, \mathbf{q}_i^{(\ell)}\}_{i\in[n],\ell\in\{0,1\}}$ and $\{\mathbf{x}_i^{(\ell)}, \mathbf{p}_i^{(\ell)}\}_{i\in[n],\ell\in\{0,1\}}$:

$$\mathbf{C}' = \begin{bmatrix} \mathbf{y}_1^{(0)} & \cdots & \mathbf{y}_k^{(0)} & \mathbf{y}_{k+1}^{(0)} & \mathbf{y}_{k+2}^{(0)} \\ \mathbf{q}_1^{(0)} & \cdots & \mathbf{q}_k^{(0)} & \mathbf{q}_{k+1}^{(0)} & \mathbf{q}_{k+2}^{(0)} \\ \mathbf{y}_1^{(1)} & \cdots & \mathbf{y}_k^{(1)} & \mathbf{y}_{k+1}^{(1)} & \mathbf{y}_{k+2}^{(1)} \\ \mathbf{q}_1^{(1)} & \cdots & \mathbf{q}_k^{(1)} & \mathbf{q}_{k+1}^{(1)} & \mathbf{q}_{k+2}^{(1)} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^{(0)} & \cdots & \mathbf{x}_k^{(0)} & \mathbf{x}_{k+1}^{(0)} & \mathbf{x}_{k+2}^{(0)} + \mathbf{x}_{k+2}^{(1)} \\ \mathbf{p}_1^{(0)} & \cdots & \mathbf{p}_k^{(0)} & \mathbf{p}_{k+1}^{(0)} & \mathbf{p}_{k+2}^{(0)} + \mathbf{p}_{k+2}^{(1)} \\ \mathbf{x}_1^{(1)} & \cdots & \mathbf{x}_k^{(1)} & \mathbf{x}_{k+2}^{(0)} + \mathrm{g}\,\mathbf{x}_{k+2}^{(1)} & \mathbf{x}_{k+1}^{(1)} \\ \mathbf{p}_1^{(1)} & \cdots & \mathbf{p}_k^{(1)} & \mathbf{p}_{k+2}^{(0)} + \mathrm{g}\,\mathbf{p}_{k+2}^{(1)} & \mathbf{p}_{k+1}^{(1)} \end{bmatrix}, \tag{116}$$

which implies

$$\begin{cases} \mathbf{x}_{k+1}^{(1)} = \mathbf{y}_{k+2}^{(1)} \\ \mathbf{x}_{k+1}^{(0)} = \mathbf{y}_{k+1}^{(0)} \end{cases} \text{and} \quad \begin{cases} \mathbf{x}_{k+2}^{(1)} = (\mathrm{g}-1)^{-1}\big(\mathbf{y}_{k+1}^{(1)} - \mathbf{y}_{k+2}^{(0)}\big) \\ \mathbf{x}_{k+2}^{(0)} = \mathbf{y}_{k+2}^{(0)} - (\mathrm{g}-1)^{-1}\big(\mathbf{y}_{k+1}^{(1)} - \mathbf{y}_{k+2}^{(0)}\big) \end{cases} \tag{117}$$

We then present the main theorem in this section.

Theorem 12. *$\mathcal{C}'$ (whose codewords are defined in (116)) is an $(n, k = n-2, 2m\mathbf{1})$ MR-MUB code over $\mathbb{F}_q$.*

Proof. For better readability, the proof is relegated to Appendix D. □

By Theorems 11 and 12, we can optimize the repair bandwidth of two selected nodes at a time, and reapply the transformation $\lceil \frac{n}{2} \rceil$ times to obtain an $(n, n-2, 2^{\lceil n/2 \rceil} m\mathbf{1})$ MR-MUB code with optimal repair bandwidth for all nodes as long as $k \mid m$.

Although the transformation in [28] holds for general k, a further generation of Theorem 12 to general k satisfying, e.g., $k < n-2$, cannot be done by following a similar procedure to the current proof, and the transformed code may not be an MR-MUB code. Hence, what we have proven in Theorem 12 is a particular case that guarantees the transformed code is an MR-MUB code if the code before transformation is an MR-MUB code.

VIII. Conclusion

In this paper, we introduced a new metric, called the *update bandwidth*, which measures the transmission efficiency in the update process of $(n, k, \mathbf{m})$ irregular array codes in DSSs. It is an essential measure in scenarios where updates are frequent. The closed-form expression of the minimum update bandwidth $\gamma_{\min}$ was established (cf. Theorem 3), and the code parameters, using which the minimum update bandwidth (MUB) can be achieved, were identified. These code parameters then constitute the class of MUB codes. As code redundancy is also an important consideration in DSSs, we next investigated the smallest code redundancy attainable by MUB codes (cf. Theorems 4 and 5).

We then seek to construct a class of irregular array codes that achieves both the minimum code redundancy and the minimum update bandwidth, named MR-MUB codes. The code parameters for MR-MUB codes are therefore determined (cf. Theorem 6). An interesting result is that under $1 < k < n-1$ and $k \mid m_i$ for $i \in [n]$, MR-MUB codes can only be vertical MDS codes unless $\mathbf{m} = [m_1 \ \cdots \ m_n]$ containing only a single non-zero component. The explicit construction of MR-MUB codes was thus focused on (n, k) vertical MDS codes, i.e., $(n, k, m\mathbf{1})$ MR-MUB codes (cf. Section V-B). A further generalization of the MR-MUB code construction was subsequently proposed for a class of MUB codes with the smallest code redundancy (cf. Section V-C).

At last, we studied the update complexity and repair bandwidth of MR-MUB codes. Through the establishment of a lower bound for the update complexity of MR-MUB codes (cf. Theorem 10), we found MR-MUB codes may not simultaneously achieve the minimum update complexity. However, an $(n, k = n-2, m\mathbf{1})$ MR-MUB code with the optimal repair bandwidth for all nodes can be constructed via the transformation in [28] (cf. Theorem 12).

There are some challenging issues remain unsolved.

1) Determine the smallest update bandwidth attainable by irregular MDS array codes [27], defined as the irregular array codes with the minimum code redundancy.
2) Determine the smallest code redundancy attainable by MUB codes when the condition of $k \mid m_i$ for $i \in [n]$ is violated.
3) Examine whether $k \mid m$ is also a necessary condition for vertical MDS array codes being MUB codes, provided $k \nmid n$.
4) Check whether the lower bound for the update complexity of $(n, k, m\mathbf{1})$ MR-MUB codes in Theorem 10 can be improved or achieved.
5) Study the optimal repair bandwidth of MR-MUB codes under $n - k \geq 3$.
6) Investigate whether the exponentially growing node size of MR-MUB codes with optimal repair bandwidth in Section VII-B can be reduced, e.g., by considering near optimal repair bandwidth [31], [32].

Appendix A
The proof of Theorem 1

We first prove that $\mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j}$ implies $\mathsf{rank}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{B}_{i,j}) = \mathsf{row}(\mathbf{A}_{i,j})$, and both $\mathbf{A}_{i,j}$ and $\mathbf{B}_{i,j}$ have full rank, which in turns validates $\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j})$.

Assuming that $\mathsf{row}(\mathbf{A}_{i,j})$ is with the minimum value, i.e., $\mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j}$, we show by contradiction that $\mathsf{row}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{A}_{i,j})$. Suppose $\mathsf{rank}(\mathbf{A}_{i,j}) < \mathsf{row}(\mathbf{A}_{i,j})$. Then, there is an invertible matrix $\mathbf{R}_{i,j}$ satisfying $\mathbf{R}_{i,j}\mathbf{A}_{i,j} = \begin{bmatrix} \mathbf{A}'_{i,j} \\ [\mathbf{0}] \end{bmatrix}$, where $[\mathbf{0}]$ is a $(\mathsf{row}(\mathbf{A}_{i,j}) - \mathsf{rank}(\mathbf{A}_{i,j})) \times m_i$ zero matrix, and $\mathsf{rank}(\mathbf{A}'_{i,j}) = \mathsf{row}(\mathbf{A}'_{i,j}) = \mathsf{rank}(\mathbf{A}_{i,j})$. We thus have $\mathbf{M}_{i,j} = \mathbf{B}_{i,j}\mathbf{R}_{i,j}^{-1}\mathbf{R}_{i,j}\mathbf{A}_{i,j} = \mathbf{B}_{i,j}\mathbf{R}_{i,j}^{-1}\begin{bmatrix} \mathbf{A}'_{i,j} \\ [\mathbf{0}] \end{bmatrix}$, which implies $\mathbf{M}_{i,j} = \mathbf{B}'_{i,j}\mathbf{A}'_{i,j}$ with $\mathbf{B}'_{i,j}$ being the first $\mathsf{rank}(\mathbf{A}_{i,j})$ columns of $\mathbf{B}_{i,j}\mathbf{R}_{i,j}^{-1}$. However, $\mathsf{rank}(\mathbf{A}'_{i,j}) = \mathsf{rank}(\mathbf{A}_{i,j}) < \gamma_{i,j}$ contradicts to the definition of $\gamma_{i,j}$. We therefore confirm that if $\mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j}$, then $\mathsf{row}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{A}_{i,j})$. Similarly, we can show that if $\mathsf{col}(\mathbf{B}_{i,j}) = \gamma_{i,j}$, then $\mathsf{col}(\mathbf{B}_{i,j}) = \mathsf{rank}(\mathbf{B}_{i,j})$. As $\mathsf{row}(\mathbf{A}_{i,j}) = \mathsf{col}(\mathbf{B}_{i,j})$, we conclude that $\mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j}$ implies $\mathsf{rank}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{B}_{i,j}) = \mathsf{row}(\mathbf{A}_{i,j}) = \mathsf{col}(\mathbf{B}_{i,j})$, and both $\mathbf{A}_{i,j}$ and $\mathbf{B}_{i,j}$ have full rank. An immediate consequence of the above proof is that this pair of $\mathbf{A}_{i,j}$ and $\mathbf{B}_{i,j}$ is a minimizer of (7). By Sylvester's rank inequality, we have

$$\mathsf{rank}(\mathbf{B}_{i,j}) + \mathsf{rank}(\mathbf{A}_{i,j}) - \mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j} \leq \mathsf{rank}(\mathbf{M}_{i,j}).$$

It can also be inferred that

$$\mathsf{rank}(\mathbf{M}_{i,j}) \leq \min\{\mathsf{rank}(\mathbf{B}_{i,j}), \mathsf{rank}(\mathbf{A}_{i,j})\} = \gamma_{i,j}.$$

Hence,

$$\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j}).$$

We next show the converse statement, i.e., if both $\mathbf{A}_{i,j}$ and $\mathbf{B}_{i,j}$ are full rank and $\mathsf{rank}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{B}_{i,j}) = \mathsf{row}(\mathbf{A}_{i,j})$, then $\mathsf{row}(\mathbf{A}_{i,j}) = \gamma_{i,j}$. Given $\mathsf{rank}(\mathbf{B}_{i,j}) = \mathsf{row}(\mathbf{A}_{i,j})$, we obtain by Sylvester's rank inequality that

$$\begin{aligned}&\mathsf{rank}(\mathbf{B}_{i,j}) + \mathsf{rank}(\mathbf{A}_{i,j}) - \mathsf{row}(\mathbf{A}_{i,j})\\ =&\mathsf{rank}(\mathbf{A}_{i,j}) \leq \mathsf{rank}(\mathbf{M}_{i,j}) = \gamma_{i,j},\end{aligned}$$

which, together with $\gamma_{i,j} = \mathsf{rank}(\mathbf{M}_{i,j}) \leq \mathsf{rank}(\mathbf{A}_{i,j})$, establishes $\mathsf{row}(\mathbf{A}_{i,j}) = \mathsf{rank}(\mathbf{A}_{i,j}) = \gamma_{i,j}$. This completes the proof.

Appendix B
The proof of Theorem 3

The proof is divided into four steps. First, we show all choices of $\{\gamma_{i,j}\}_{i \neq j \in [n]}$ satisfying (39) yield an update bandwidth no less than the $\gamma_{\min}$ given in (40). Second, we verify (41) can achieve $\gamma_{\min}$. Third, we prove that (41) is the only assignment that can achieve $\gamma_{\min}$ under $k < n - 1$. Last, we complete the proof by considering separately the situation of $k = n - 1$.

Step 1. Fix a set of $\{\gamma_{i,j}\}_{i \neq j \in [n]}$ satisfying (39). Since (39) holds for arbitrary $\mathcal{E}$, we can let $\bar{\mathcal{E}} = \{j_1(i), \ldots, j_k(i)\}$ and obtain

$$\sum_{u \in [k]} \gamma_{i, j_u(i)} \geq m_i. \tag{118}$$

Noting that $\{\gamma_{i,j_u(i)}\}_{u \in [n-1]}$ is in ascending order (cf. (42)), and that $\gamma_{i,j}$ is a non-negative integer, we obtain from (118) that

$$\gamma_{i, j_k(i)} \geq \left\lceil \frac{m_i}{k} \right\rceil. \tag{119}$$

We continue to derive

$$\begin{aligned}\sum_{j \in [n] \setminus \{i\}} \gamma_{i,j} &= \sum_{u \in [k]} \gamma_{i,j_u(i)} + \sum_{u \in [n-1] \setminus [k]} \gamma_{i,j_u(i)}\\ &\geq \sum_{u \in [k]} \gamma_{i,j_u(i)} + (n-k-1)\gamma_{i,j_k(i)}.\end{aligned} \tag{120}$$

Combining (118), (119) and (120) gives

$$\sum_{j \in [n] \setminus \{i\}} \gamma_{i,j} \geq m_i + (n-k-1)\left\lceil \frac{m_i}{k} \right\rceil, \tag{121}$$

which implies

$$\begin{aligned}\gamma =&\frac{1}{n}\sum_{i \in [n]}\sum_{j \in [n] \setminus \{i\}} \gamma_{i,j}\\ \geq&\frac{B}{n} + \frac{(n-k-1)}{n}\sum_{i \in [n]}\left\lceil \frac{m_i}{k} \right\rceil\\ =&\gamma_{\min}.\end{aligned}$$

Step 2. Next, we confirm (41) is a valid choice of $\{\gamma_{i,j}\}_{i \neq j \in [n]}$ that achieves $\gamma_{\min}$. The validity of (39) can be confirmed by

$$w_i = k\left\lceil \frac{m_i}{k} \right\rceil - m_i < k\left(\frac{m_i}{k} + 1\right) - m_i = k \tag{122}$$

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2021.3105439

and

$$\begin{aligned}\sum_{j\in\bar{\mathcal{E}}}\gamma_{i,j} \geq &\sum_{i\in[k]}\gamma_{i,j_u(i)}\\ =&w_i\left\lfloor\frac{m_i}{k}\right\rfloor+(k-w_i)\left\lceil\frac{m_i}{k}\right\rceil\\ =&k\left\lceil\frac{m_i}{k}\right\rceil-\left(k\left\lceil\frac{m_i}{k}\right\rceil-m_i\right)\left(\left\lceil\frac{m_i}{k}\right\rceil-\left\lfloor\frac{m_i}{k}\right\rfloor\right)\\ =&\begin{cases}k\left\lceil\frac{m_i}{k}\right\rceil-0, & k\mid m_i\\ k\left\lceil\frac{m_i}{k}\right\rceil-\left(k\left\lceil\frac{m_i}{k}\right\rceil-m_i\right), & k\nmid m_i\end{cases}\\ =&m_i. \end{aligned} \tag{123}$$

Hence, we can derive based on (42) and (123) that

$$\begin{aligned}\sum_{j\in[n]\setminus\{i\}}\gamma_{i,j}=&\sum_{u\in[k]}\gamma_{i,j_u(i)}+\sum_{u\in[n-1]\setminus[k]}\gamma_{i,j_u(i)}\\ =&m_i+(n-k-1)\left\lceil\frac{m_i}{k}\right\rceil,\end{aligned}$$

which immediately gives

$$\begin{aligned}\gamma=&\frac{1}{n}\sum_{i\in[n]}\sum_{j\in[n]\setminus\{i\}}\gamma_{i,j}\\ =&\frac{B}{n}+\frac{(n-k-1)}{n}\sum_{i\in[n]}\left\lceil\frac{m_i}{k}\right\rceil=\gamma_{\min}.\end{aligned}$$

Step 3. It remains to show by contradiction that no other assignment of $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ can achieve $\gamma_{\min}$. The task will be done under $k<n-1$ in this step. The situation of $k=n-1$ will be separately considered in the next step.

Suppose $\{\gamma'_{i,j}\}_{i\neq j\in[n]}$ also achieves $\gamma_{\min}$. We then differentiate among four cases.

Case 1: If there is $i'\in[n]$ such that $\gamma'_{i',j_{w_{i'}+1}(i')}>\left\lceil\frac{m_{i'}}{k}\right\rceil$, then we obtain from (42) and (122) that $\gamma'_{i',j_k(i')}>\left\lceil\frac{m_{i'}}{k}\right\rceil$, which together with (118) implies

$$\begin{aligned}&\sum_{j\in[n]\setminus\{i'\}}\gamma'_{i',j}\\ =&\sum_{u\in[k]}\gamma'_{i',j_u(i')}+\sum_{u\in[n-1]\setminus[k]}\gamma'_{i',j_u(i')}\\ >&m_{i'}+(n-k-1)\left\lceil\frac{m_{i'}}{k}\right\rceil.\end{aligned} \tag{124}$$

Because $\{\gamma'_{i,j}\}_{i\neq j\in[n]}$ fulfills (39) and hence validates (121), we have

$$\begin{aligned}\gamma=&\frac{1}{n}\sum_{i\in[n]}\sum_{j\in[n]\setminus\{i\}}\gamma'_{i,j}\\ =&\frac{1}{n}\Big(\sum_{j\in[n]\setminus\{i'\}}\gamma'_{i',j}+\sum_{i\in[n]\setminus\{i'\}}\sum_{j\in[n]\setminus\{i\}}\gamma'_{i,j}\Big)\\ >&\frac{1}{n}\left(\left(m_{i'}+(n-k-1)\left\lceil\frac{m_{i'}}{k}\right\rceil\right)\right.\\ &\left.+\sum_{i\in[n]\setminus\{i'\}}\left(m_i+(n-k-1)\left\lceil\frac{m_i}{k}\right\rceil\right)\right)\\ =&\gamma_{\min},\end{aligned} \tag{125}$$

contradicting to the assumption of $\{\gamma'_{i,j}\}_{i\neq j\in[n]}$ achieving $\gamma_{\min}$.

Case 2: If there is $i'\in[n]$ such that $\gamma'_{i',j_{w_{i'}+1}(i')}<\left\lceil\frac{m_{i'}}{k}\right\rceil$ and $w_{i'}+1=k$, then we can infer from (42) that

$$\gamma'_{i',j_{w_{i'}}(i')}\leq\left\lfloor\frac{m_{i'}}{k}\right\rfloor, \tag{126}$$

and hence

$$\begin{aligned}&\sum_{u\in[k]}\gamma'_{i',j_u(i')}\\ =&\sum_{u\in[w_{i'}]}\gamma'_{i',j_u(i')}+\gamma'_{i',j_{w_{i'}+1}(i')}\\ &+\underbrace{\sum_{u\in[k]\setminus[w_{i'}+1]}\gamma'_{i',j_u(i')}}_{=0}\\ <&w_{i'}\left\lfloor\frac{m_{i'}}{k}\right\rfloor+\left\lceil\frac{m_{i'}}{k}\right\rceil\\ &+\underbrace{(k-w_{i'}-1)}_{=0}\left\lceil\frac{m_{i'}}{k}\right\rceil=m_{i'}\end{aligned} \tag{127}$$

where the first strict inequality in (127) is due to $\gamma'_{i',j_{w_{i'}+1}(i')}<\left\lceil\frac{m_{i'}}{k}\right\rceil$, and the last equality follows from a similar derivation to (123). The inequality (127) then contradicts to (118).

Case 3: If there is $i'\in[n]$ such that $\gamma'_{i',j_{w_{i'}+1}(i')}<\left\lceil\frac{m_{i'}}{k}\right\rceil$ and $w_{i'}+1<k$, then we must have $\gamma'_{i',j_k(i')}>\left\lceil\frac{m_{i'}}{k}\right\rceil$. This is because in case $\gamma'_{i',j_k(i')}\leq\left\lceil\frac{m_{i'}}{k}\right\rceil$ under $\gamma'_{i',j_{w_{i'}+1}(i')}<\left\lceil\frac{m_{i'}}{k}\right\rceil$ and $w_{i'}+1<k$, we can obtain from (42) that

$$\gamma'_{i',j_{w_{i'}}(i')}\leq\left\lfloor\frac{m_{i'}}{k}\right\rfloor, \tag{128}$$

and hence a similar derivation to (127) gives

$$\begin{aligned}&\sum_{u\in[k]}\gamma'_{i',j_u(i')}\\ =&\sum_{u\in[w_{i'}]}\gamma'_{i',j_u(i')}+\gamma'_{i',j_{w_{i'}+1}(i')}\\ &+\sum_{u\in[k]\setminus[w_{i'}+1]}\gamma'_{i',j_u(i')}\\ <&w_{i'}\left\lfloor\frac{m_{i'}}{k}\right\rfloor+\left\lceil\frac{m_{i'}}{k}\right\rceil\\ &+(k-w_{i'}-1)\left\lceil\frac{m_{i'}}{k}\right\rceil=m_{i'}.\end{aligned} \tag{129}$$

The inequality (129) then contradicts to (118), and therefore $\gamma'_{i',j_k(i')}>\left\lceil\frac{m_{i'}}{k}\right\rceil$. We continue to derive based on (118) that

$$\begin{aligned}&\sum_{j\in[n]\setminus\{i'\}}\gamma'_{i',j}\\ =&\sum_{u\in[k]}\gamma'_{i',j_u(i')}+\sum_{u\in[n-1]\setminus[k]}\gamma'_{i',j_u(i')}\\ >&m_{i'}+(n-k-1)\left\lceil\frac{m_{i'}}{k}\right\rceil,\end{aligned} \tag{130}$$

based on which the same contradiction as (125) can be resulted.

Case 4: The previous three cases indicate that $\gamma'_{i',j_{w_{i'}+1}(i')} = \lceil \frac{m_{i'}}{k} \rceil$ for all $i' \in [n]$. Now if there is $i' \in [n]$ and $w_{i'} < u' \leq n-1$ such that $\gamma'_{i',j_{u'}(i')} < \gamma'_{i',j_{u'+1}(i')}$, then we again use (118) to obtain

$$\begin{aligned}&\sum_{j\in[n]\setminus\{i'\}} \gamma'_{i',j}\\ &= \sum_{u\in[k]} \gamma'_{i',j_u(i')} + \sum_{u\in[n-1]\setminus[k]} \gamma'_{i',j_u(i')}\\ &> m_{i'} + (n-k-1)\left\lceil \frac{m_{i'}}{k} \right\rceil,\end{aligned}$$

based on which the same contradiction as (125) can, again, be resulted.

The above four cases conclude that $\gamma'_{i,j_u(i)} = \lceil \frac{m_i}{k} \rceil$ for $u \in [n-1] \setminus [w_i]$ and $i \in [n]$. Finally, (121) implies

$$\begin{aligned}\sum_{j\in[n]\setminus\{i\}} \gamma'_{i,j} &= \sum_{u\in[w_i]} \gamma'_{i,j_u(i)} + (n-w_i-1)\left\lceil \frac{m_i}{k} \right\rceil\\ &\geq m_i + (n-k-1)\left\lceil \frac{m_i}{k} \right\rceil.\end{aligned} \tag{131}$$

Since the sum of the left-hand-side of (131) is equal to the sum of the right-hand-side of (131), which is exactly $\gamma_{\min}$, we must have

$$\begin{aligned}&\sum_{u\in[w_i]} \gamma'_{i,j_u(i)} + (n-w_i-1)\left\lceil \frac{m_i}{k} \right\rceil\\ =&m_i + (n-k-1)\left\lceil \frac{m_i}{k} \right\rceil,\end{aligned}$$

which in turn gives

$$\begin{aligned}\sum_{u\in[w_i]} \gamma'_{i,j_u(i)} &= m_i + (w_i - k)\left\lceil \frac{m_i}{k} \right\rceil\\ &= w_i \left\lfloor \frac{m_i}{k} \right\rfloor,\end{aligned}$$

where the last equality can be confirmed similarly as (123).

Step 4. Last, we prove (43). Note that the proofs in Steps 1 and 2 remain valid under $k = n-1$, but some derivations in Step 3, e.g., (124), may not be applied when $k = n-1$. In fact, when $k = n-1$, a larger class of assignments on $\{\gamma_{i,j}\}_{i\neq j\in[n]}$ can achieve $\gamma_{\min}$. We show (43) by contradiction. Suppose $\{\gamma'_{i,j}\}_{i\neq j\in[n]}$ achieves $\gamma_{\min}$ but satisfies $\sum_{j\in[n]\setminus\{i'\}} \gamma'_{i',j} > m_{i'}$ for some $i' \in [n]$. Then,

$$\begin{aligned}\gamma &= \frac{1}{n}\left(\sum_{i\in[n]\setminus\{i'\}} \sum_{j\in[n]\setminus\{i\}} \gamma'_{i,j} + \sum_{j\in[n]\setminus\{i'\}} \gamma'_{i',j} \right)\\ &> \frac{1}{n}\left(\sum_{i\in[n]\setminus\{i'\}} m_i + m_{i'} \right)\\ &= \frac{B}{n} = \gamma_{\min},\end{aligned}$$

which leads to a contradiction.

Appendix C
The Proof of Theorem 9

Similar to the proof of Theorem 8, the substantiation of this theorem requires verifying two properties: $i)$ $\mathcal{C}_{\rm U}$ is an $(n, k, \mathbf{m})$ array code, and $ii)$ $\mathcal{C}_{\rm U}$ achieves $R_{\tt sma}$ and $\gamma_{\min}$.

First, we justify $i)$, i.e., $\mathcal{C}_{\rm U}$ satisfying that given any set $\mathcal{E} \subset [n]$ with $|\mathcal{E}| = n-k$, the codeword $\mathbf{C}$ of $\mathcal{C}_{\rm U}$ can be reconstructed from $\mathbf{C}_{\bar{\mathcal{E}}}$. When $\mathbf{C}_{\bar{\mathcal{E}}}$ is given, both $\mathbf{X}_{\bar{\mathcal{E}}}$ and $\mathbf{P}_{\bar{\mathcal{E}}}$ are known, and so are $\{\mathbf{p}_{\bar{e}_i,\bar{e}_j}\}_{i\neq j\in[k]}$ according to (9). We can then establish from (11) that

$$\begin{aligned}&\mathbf{p}_{\bar{e}_j} - \sum_{i=1,i\neq j}^{k} \mathbf{B}_{\bar{e}_i,\bar{e}_j}\mathbf{p}_{\bar{e}_i,\bar{e}_j}\\ &= \begin{bmatrix}\mathbf{B}_{e_1,\bar{e}_j} & \ldots & \mathbf{B}_{e_{n-k},\bar{e}_j}\end{bmatrix} \begin{bmatrix}\mathbf{p}_{e_1,\bar{e}_j}\\ \vdots \\ \mathbf{p}_{e_{n-k},\bar{e}_j}\end{bmatrix} \quad \forall j \in [k].\end{aligned} \tag{132}$$

According to (48), we have

$$\begin{aligned}\mathsf{row}([\mathbf{B}_{e_1,\bar{e}_j} \ \ldots \ \mathbf{B}_{e_{n-k},\bar{e}_j}]) = p_j &\geq \sum_{i\in[n-k]} \frac{m_{e_i}}{k}\\ &= \sum_{i\in[n-k]} \mathsf{col}(\mathbf{B}_{e_i,\bar{e}_j}).\end{aligned} \tag{133}$$

Since any p_j columns of $\mathbf{V}_j$, as defined in (90), forms an invertible matrix, we obtain from (133) that $[\mathbf{B}_{e_1,\bar{e}_j} \ \ldots \ \mathbf{B}_{e_{n-k},\bar{e}_j}]$ is of full column rank, and hence $\{\mathbf{p}_{e_i,\bar{e}_j}\}_{i\in[n-k],j\in[k]}$ can be solved via (132). With the knowledge of k columns $\{\mathbf{p}_{e_i,\bar{e}_j}\}_{j\in[k]}$ of $\mathbf{F}_{e_i}$ in (89), we can recover $\mathbf{x}_{e_i}$ via the decoding algorithm of the $(n-1,k)$ MDS array code $\mathcal{M}_{e_i}$. By this procedure, $\{\mathbf{x}_i\}_{i\in[n]}$ can all be recovered.

Next, we verify $ii)$. From (89), we have $\mathbf{p}_{i,j} \in \mathbb{F}_q^{\frac{m_i}{k}}$ and hence $\gamma_{i,j} = \frac{m_i}{k}$, which leads to $\gamma = \gamma_{\min}$ as pointed out in (46a). In addition, (91) shows $\{p_j\}_{j\in[n]}$ follows (48), and hence $R_{\tt sma}$ is achieved as addressed in Theorem 4. The justification of the two required properties of $\mathcal{C}_{\rm U}$ is thus completed.

Appendix D
The Proof of Theorem 12

Recall from (67) that each node of $\mathcal{C}$ has $\alpha = m + p = m + \frac{(n-k)}{k}m = \frac{nm}{k}$ symbols. Thus, from (116), each node of $\mathcal{C}'$ contains $\alpha' = 2\alpha = \frac{2nm}{k}$ symbols.

Since each node of the transformed code $\mathcal{C}'$ has $p' = 2p$ parity symbols, its code redundancy achieves the minimum value given in (35). It remains to show $\mathcal{C}'$ also achieves the minimum update bandwidth.

Using the notations in Section II-C, where the encoding matrices of $\mathcal{C}$ are denoted as $\{\mathbf{A}_{i,j}\}_{i,j\in[n]}$ and $\{\mathbf{B}_{i,j}\}_{i,j\in[n]}$, we consider the update of node i of $\mathcal{C}'$ for $i \in [k]$. From (116), we need to compute

$$\Delta\mathbf{y}_i^{(\ell)} = \mathbf{y}_i^{(\ell)*} - \mathbf{y}_i^{(\ell)} \quad \text{for } \ell = 1, 2, \tag{134}$$

where we add a star in the superscript to denote the value of a vector after this updating. Then, we must renew $\mathbf{q}_j^{(\ell)}$ for $j \in [k]\setminus\{i\}$ based on $\Delta\mathbf{y}_i^{(\ell)}$ according to $\mathbf{q}_j^{(\ell)} + \mathbf{B}_{i,j}\mathbf{A}_{i,j}\Delta\mathbf{y}_i^{(\ell)}$, since the correspondence in (116) indicates $\mathbf{q}_j^{(\ell)} = \mathbf{p}_j^{(\ell)}$ and $\Delta\mathbf{y}_i^{(\ell)} = \Delta\mathbf{x}_i^{(\ell)}$, i.e.,

$$\begin{aligned}\mathbf{q}_j^{(\ell)*} &= \mathbf{p}_j^{(\ell)*} = \mathbf{p}_j^{(\ell)} + \mathbf{B}_{i,j}\mathbf{A}_{i,j}\Delta\mathbf{x}_i^{(\ell)} \\ &= \mathbf{q}_j^{(\ell)} + \mathbf{B}_{i,j}\mathbf{A}_{i,j}\Delta\mathbf{x}_i^{(\ell)}.\end{aligned}$$

Accordingly, node i shall send both $\mathbf{A}_{i,j}\Delta\mathbf{y}_i^{(0)} = \mathbf{A}_{i,j}\Delta\mathbf{x}_i^{(0)}$ and $\mathbf{A}_{i,j}\Delta\mathbf{y}_i^{(1)} = \mathbf{A}_{i,j}\Delta\mathbf{x}_i^{(1)}$ to node $j \in [k] \setminus \{i\}$, which implies $\gamma'_{i,j} = 2\gamma_{i,j}$ for $i \neq j \in [k]$. The renew of $\mathbf{q}_{k+1}$ requires sending $\mathbf{A}_{i,k+1}\Delta\mathbf{x}_i^{(0)}$ and $\mathbf{A}_{i,k+2}\big(\Delta\mathbf{x}_i^{(0)} + \mathrm{g}\,\Delta\mathbf{x}_i^{(1)}\big)$ to node $k+1$ since

$$\begin{aligned}\mathbf{q}_{k+1}^{(0)*} &= \mathbf{q}_{k+1}^{(0)} + \mathbf{B}_{i,k+1}\mathbf{A}_{i,k+1}\Delta\mathbf{y}_i^{(0)} \\ &= \mathbf{p}_{k+1}^{(0)} + \mathbf{B}_{i,k+1}\mathbf{A}_{i,k+1}\Delta\mathbf{x}_i^{(0)},\end{aligned}$$

and

$$\begin{aligned}\mathbf{q}_{k+1}^{(1)*} =& \mathbf{p}_{k+2}^{(0)*} + \mathrm{g}\,\mathbf{p}_{k+2}^{(1)*} \\ =& \big(\mathbf{p}_{k+2}^{(0)} + \mathbf{B}_{i,k+2}\mathbf{A}_{i,k+2}\Delta\mathbf{x}_i^{(0)}\big) \\ &+ \mathrm{g}\,\big(\mathbf{p}_{k+2}^{(1)} + \mathbf{B}_{i,k+2}\mathbf{A}_{i,k+2}\Delta\mathbf{x}_i^{(1)}\big) \\ =& \mathbf{p}_{k+2}^{(0)} + \mathrm{g}\,\mathbf{p}_{k+2}^{(1)} + \mathbf{B}_{i.k+2}\mathbf{A}_{i,k+2}\big(\Delta\mathbf{x}_i^{(0)} + \mathrm{g}\,\Delta\mathbf{x}_i^{(1)}\big) \\ =& \mathbf{q}_{k+1}^{(1)} + \mathbf{B}_{i.k+2}\mathbf{A}_{i,k+2}\big(\Delta\mathbf{x}_i^{(0)} + \mathrm{g}\,\Delta\mathbf{x}_i^{(1)}\big).\end{aligned}$$

Thus, $\gamma'_{i,k+1} = \gamma_{i,k+1} + \gamma_{i,k+2}$ for $i \in [k]$. We can similarly obtain $\gamma'_{i,k+2} = \gamma_{i,k+1} + \gamma_{i,k+2}$ for $i \in [k]$ when concerning the adjustment of $\mathbf{q}_{k+2}$ due to the update of $\mathbf{y}_i$.

We next consider the update of $\mathbf{y}_{k+1}$. Again, we compute $\Delta\mathbf{y}_{k+1}^{(\ell)} = \mathbf{y}_{k+1}^{(\ell)*} - \mathbf{y}_{k+1}^{(\ell)}$ for $\ell = 0, 1$. Note that all of $\mathbf{x}_{k+1}^{(0)}$, $\mathbf{x}_{k+2}^{(0)}$ and $\mathbf{x}_{k+2}^{(1)}$ are involved in this update. Since $\mathbf{y}_{k+2}^{(0)} = \mathbf{x}_{k+2}^{(0)} + \mathbf{x}_{k+2}^{(1)}$ remains unchanged, we have $\Delta\mathbf{y}_{k+2}^{(0)} = \Delta\mathbf{x}_{k+2}^{(0)} + \Delta\mathbf{x}_{k+2}^{(1)} = \mathbf{0}$, which together with (117) implies $\Delta\mathbf{x}_{k+1}^{(0)} = \Delta\mathbf{y}_{k+1}^{(0)}$ and $\Delta\mathbf{x}_{k+2}^{(1)} = (\mathrm{g}-1)^{-1}\Delta\mathbf{y}_{k+1}^{(1)}$. As a result, for $j \in [k]$, the new parity vectors $\mathbf{q}_j^*$ are renewed according to

$$\begin{aligned}\mathbf{q}_j^{(0)*} =& \mathbf{p}_j^{(0)*} \\ =& \mathbf{p}_j^{(0)} + \mathbf{B}_{k+1,j}\mathbf{A}_{k+1,j}\Delta\mathbf{x}_{k+1}^{(0)} + \mathbf{B}_{k+2,j}\mathbf{A}_{k+2,j}\Delta\mathbf{x}_{k+2}^{(0)} \\ =& \mathbf{q}_j^{(0)} + \mathbf{B}_{k+1,j}\mathbf{A}_{k+1,j}\Delta\mathbf{x}_{k+1}^{(0)} + \mathbf{B}_{k+2,j}\mathbf{A}_{k+2,j}\Delta\mathbf{x}_{k+2}^{(0)} \\ =& \mathbf{q}_j^{(0)} + \mathbf{B}_{k+1,j}\mathbf{A}_{k+1,j}\Delta\mathbf{y}_{k+1}^{(0)} \\ &- (\mathrm{g}-1)^{-1}\mathbf{B}_{k+2,j}\mathbf{A}_{k+2,j}\Delta\mathbf{y}_{k+1}^{(1)}\end{aligned}$$

and

$$\begin{aligned}\mathbf{q}_j^{(1)*} =& \mathbf{p}_j^{(1)*} \\ =& \mathbf{p}_j^{(1)} + \mathbf{B}_{k+2,j}\mathbf{A}_{k+2,j}\Delta\mathbf{x}_{k+2}^{(1)} \\ =& \mathbf{q}_j^{(1)} + \mathbf{B}_{k+2,j}\mathbf{A}_{k+2,j}\Delta\mathbf{x}_{k+2}^{(1)} \\ =& \mathbf{q}_j^{(1)} + (\mathrm{g}-1)^{-1}\mathbf{B}_{k+2,j}\mathbf{A}_{k+2,j}\Delta\mathbf{y}_{k+1}^{(1)},\end{aligned}$$

which indicates node $k+1$ should send $\mathbf{A}_{k+1,j}\Delta\mathbf{y}_{k+1}^{(0)}$ and $\mathbf{A}_{k+2,j}\Delta\mathbf{y}_{k+1}^{(1)}$ to node j to renew its parity vector; hence, $\gamma'_{k+1,j} = \gamma_{k+1,j} + \gamma_{k+2,j}$ for $j \in [k]$. Concerning the renew of $\mathbf{q}_{k+2}$, we derive

$$\begin{aligned}\mathbf{q}_{k+2}^{(0)*} =& \mathbf{p}_{k+2}^{(0)*} + \mathbf{p}_{k+2}^{(1)*} \\ =& \mathbf{p}_{k+2}^{(0)} + \mathbf{p}_{k+2}^{(1)} + \mathbf{B}_{k+1,k+2}\mathbf{A}_{k+1,k+2}\Delta\mathbf{x}_{k+1}^{(0)} \\ &+ \mathbf{B}_{k+2,k+2}\mathbf{A}_{k+2,k+2}\Delta\mathbf{x}_{k+2}^{(0)} \\ &+ \mathbf{B}_{k+2,k+2}\mathbf{A}_{k+2,k+2}\Delta\mathbf{x}_{k+2}^{(1)} \\ =& \mathbf{q}_{k+2}^{(0)} + \mathbf{B}_{k+1,k+2}\mathbf{A}_{k+1,k+2}\Delta\mathbf{x}_{k+1}^{(0)} \\ &+ \mathbf{B}_{k+2,k+2}\mathbf{A}_{k+2,k+2}\Delta\mathbf{x}_{k+2}^{(0)} \\ &+ \mathbf{B}_{k+2,k+2}\mathbf{A}_{k+2,k+2}\Delta\mathbf{x}_{k+2}^{(1)} \\ =& \mathbf{q}_{k+2}^{(0)} + \mathbf{B}_{k+1,k+2}\mathbf{A}_{k+1,k+2}\Delta\mathbf{y}_{k+1}^{(0)}\end{aligned}$$

and

$$\begin{aligned}\mathbf{q}_{k+2}^{(1)*} =& \mathbf{p}_{k+1}^{(1)*} \\ =& \mathbf{p}_{k+1}^{(1)} + \mathbf{B}_{k+2,k+1}\mathbf{A}_{k+2,k+1}\Delta\mathbf{x}_{k+2}^{(1)} \\ =& \mathbf{q}_{k+2}^{(1)} + \mathbf{B}_{k+2,k+1}\mathbf{A}_{k+2,k+1}\Delta\mathbf{x}_{k+2}^{(1)} \\ =& \mathbf{q}_{k+2}^{(1)} + (\mathrm{g}-1)^{-1}\mathbf{B}_{k+2,k+1}\mathbf{A}_{k+2,k+1}\Delta\mathbf{y}_{k+1}^{(1)},\end{aligned}$$

which indicates node $k+1$ should send $\mathbf{A}_{k+1,k+2}\Delta\mathbf{y}_{k+1}^{(0)}$ and $\mathbf{A}_{k+2,k+1}\Delta\mathbf{y}_{k+1}^{(1)}$ to node $k+2$; hence, $\gamma'_{k+1,k+2} = \gamma_{k+1,k+2} + \gamma_{k+2,k+1}$.

Last, we consider the update of $\mathbf{y}_{k+2}$, and can similarly obtain $\gamma'_{k+2,j} = \gamma_{k+1,j} + \gamma_{k+2,j}$ for $j \in [k]$ and $\gamma'_{k+2,k+1} = \gamma_{k+1,k+2} + \gamma_{k+2,k+1}$.

We summarize the matrix of $\gamma'_{i,j}$ for $i \neq j \in [n]$ in (135) at the top of next page. Since $\mathcal{C}$ is an $(n = k+2, k, m\mathbf{1})$ MR-MUB code, we know from (68) that $\gamma_{i,j} = \frac{m}{k}$ for $i \neq j \in [n]$. We then conclude from (135) that $\gamma'_{i,j} = \frac{2m}{k}$ for $i \neq j \in [n]$. Consequently, $\mathcal{C}'$ is an $(n, n-2, 2m\mathbf{1})$ MR-MUB code over $\mathbb{F}_q$, which can be confirmed by Theorem 3.

REFERENCES

[1] D. Ford, F. Labelle, F. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in globally distributed storage systems," in *Proc. of OSDI*, Jan. 2010, pp. 61–74.

[2] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. li, and S. Yekhanin, "Erasure coding in Windows Azure storage," in *Proc. USENIX Annual Technical Conference*, June 2012, pp. 15–26.

[3] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[4] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 5227–5239, Aug. 2011.

[5] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6925–6934, Nov. 2012.

[6] D. S. Papailiopoulos and A. G. Dimakis, "Locally repairable codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 10, pp. 5843–5855, Oct. 2014.

[7] P. Nakkiran, N. B. Shah, and K. V. Rashmi, "Fundamental limits on communication for oblivious updates in storage networks," in *2014 IEEE Global Communications Conference*, 2014, pp. 2363–2368.

[8] P. Nakkiran, N. B. Shah, K. V. Rashmi, A. Sahai, and K. Ramchandran, "Optimal oblivious updates in distributed storage networks," *aaa*, 2016. [Online]. Available: http://www.cs.cmu.edu/~rvinayak/papers/ObUp.pdf

[9] N. Prakash and M. Médard, "Communication cost for updating linear functions when message updates are sparse: Connections to maximally recoverable codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 12, pp. 7557–7576, 2018.

$$\begin{bmatrix} \gamma'_{1,2} & \cdots & \gamma'_{1,k} & \gamma'_{1,k+1} & \gamma'_{1,k+2} \\ \gamma'_{2,1} & \cdots & \gamma'_{2,k} & \gamma'_{2,k+1} & \gamma'_{2,k+2} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \gamma'_{k,1} & \cdots & \gamma'_{k,k-1} & \gamma'_{k,k+1} & \gamma'_{k,k+2} \\ \gamma'_{k+1,1} & \cdots & \cdots & \gamma'_{k+1,k} & \gamma'_{k+1,k+2} \\ \gamma'_{k+2,1} & \cdots & \cdots & \gamma'_{k+2,k} & \gamma'_{k+2,k+1} \end{bmatrix} = \begin{bmatrix} 2\gamma_{1,2} & \cdots & 2\gamma_{1,k} & \gamma_{1,k+1}+\gamma_{1,k+2} & \gamma_{1,k+1}+\gamma_{1,k+2} \\ 2\gamma_{2,1} & \cdots & 2\gamma_{2,k} & \gamma_{2,k+1}+\gamma_{2,k+2} & \gamma_{2,k+1}+\gamma_{2,k+2} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 2\gamma_{k,1} & \cdots & 2\gamma_{k,k-1} & \gamma_{k,k+1}+\gamma_{k,k+2} & \gamma_{k,k+1}+\gamma_{k,k+2} \\ \gamma_{k+1,1}+\gamma_{k+2,1} & \cdots & \cdots & \gamma_{k+1,k}+\gamma_{k+2,k} & \gamma_{k+1,k+2}+\gamma_{k+2,k+1} \\ \gamma_{k+1,1}+\gamma_{k+2,1} & \cdots & \cdots & \gamma_{k+1,k}+\gamma_{k+2,k} & \gamma_{k+1,k+2}+\gamma_{k+2,k+1} \end{bmatrix}. \tag{135}$$

[10] J. Harshan, F. Oggier, and A. Datta, "Sparsity exploiting erasure coding for resilient storage and efficient I/O access in delta based versioning systems," in *2015 IEEE 35th International Conference on Distributed Computing Systems*, 2015, pp. 798–799.

[11] R. E. Ali and V. R. Cadambe, "Harnessing correlations in distributed erasure-coded key-value stores," *IEEE Transactions on Communications*, vol. 67, no. 9, pp. 5907–5920, 2019.

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

[13] M. Blaum, J. Bruck, and A. Vardy, "MDS array codes with independent parity symbols," *IEEE Trans. Inf. Theory*, vol. 42, pp. 529 – 542, April 1996.

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

[15] 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.

[16] C. Jin, H. Jiang, D. Feng, and L. Tian, "P-code: A new RAID-6 code with optimal properties," in *Proc. of the ICS09, Yorktown Heights*, 2009.

[17] 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, 2013.

[18] J. Li, X. Tang, and U. Parampalli, "A framework of constructions of minimal storage regenerating codes with the optimal access/update property," *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 1920–1932, 2015.

[19] M. Mehrabi, M. Ardakani, and M. Khabbazian, "Minimizing the update complexity of facebook HDFS-RAID locally repairable code," in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, 2017, pp. 1–5.

[20] M. Mehrabi, M. Shahabinejad, M. Ardakani, and M. Khabbazian, "Improving the update complexity of locally repairable codes," *IEEE Transactions on Communications*, vol. 66, no. 9, pp. 3711–3720, 2018.

[21] N. P. Anthapadmanabhan, E. Soljanin, and S. Vishwanath, "Update-efficient codes for erasure correction," in *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sep. 2010, pp. 376–382.

[22] Ankit Singh Rawat, S. Vishwanath, A. Bhowmick, and E. Soljanin, "Update efficient codes for distributed storage," in *2011 IEEE International Symposium on Information Theory Proceedings*, 2011, pp. 1457–1461.

[23] A. Mazumdar, G. W. Wornell, and V. Chandar, "Update efficient codes for error correction," in *2012 IEEE International Symposium on Information Theory Proceedings*, July 2012, pp. 1558–1562.

[24] Y. S. Han, H. Pai, R. Zheng, and P. K. Varshney, "Update-efficient error-correcting product-matrix codes," *IEEE Transactions on Communications*, vol. 63, no. 6, pp. 1925–1938, June 2015.

[25] K. S. Esmaili, A. Chiniah, and A. Datta, "Efficient updates in cross-object erasure-coded storage systems," in *2013 IEEE International Conference on Big Data*, 2013, pp. 28–32.

[26] F. Zhang, J. Huang, and C. Xie, "Two efficient partial-updating schemes for erasure-coded storage clusters," in *2012 IEEE Seventh International Conference on Networking, Architecture, and Storage*, June 2012, pp. 21–30.

[27] F. Tosato and M. Sandell, "Irregular MDS array codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5304–5314, Sep. 2014.

[28] J. Li, X. Tang, and C. Tian, "A generic transformation to enable optimal repair in MDS codes for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 64, no. 9, pp. 6257–6267, Sep. 2018.

[29] C. Chen, S. Lin, and N. Yu, "Irregular MDS array codes with fewer parity symbols," *IEEE Communications Letters*, vol. 23, no. 11, pp. 1909–1912, Nov. 2019.

[30] G. Zaitsev, V. A. Zinov'ev, and N. Semakov, "Minimum-check density codes for correcting bytes of errors, erasures, or defects," *Problemy Peredachi Informatsii*, vol. 19, no. 3, pp. 29–37, 1983.

[31] A. S. Rawat, I. Tamo, V. Guruswami, and K. Efremenko, "MDS code constructions with small sub-packetization and near-optimal repair bandwidth," *IEEE Trans. Inf. Theory*, vol. 64, no. 10, pp. 6506–6525, Oct. 2018.

[32] J. Li, Y. Liu, and X. Tang, "A systematic construction of MDS codes with small sub-packetization level and near-optimal repair bandwidth," *IEEE Trans. Inf. Theory*, vol. 67, no. 4, pp. 2162–2180, Apr. 2021.

Zhengrui Li received the B.Eng. degree in Information Security and M.Phil. degree in Cyber Security from the University of Science and Technology of China, Hefei, Anhui, China, in 2018 and 2021, respectively. He is now a PhD candidate at the Hong Kong University of Science and Technology. His research interests include erasure coding and coding for distributed storage systems.

Sian-Jheng Lin (M'16) received the B.Sc., M.Sc., and Ph.D. degrees in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 2004, 2006, and 2010, respectively. From 2010 to 2014, he was a postdoc with the Research Center for Information Technology Innovation, Academia Sinica. From 2014 to 2016, he was a postdoc with the Electrial Engineering Department at King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia. He was a part-time lecturer at Yuanpei University from 2007 to 2008, and at Hsuan Chuang University From 2008 to 2010. He is currently a project researcher with the School of Information Science and Technology at University of Science and Technology of China (USTC), Hefei, China. In recent years, his research focuses on the algorithms for MDS codes and its applications to storage systems.

Po-Ning Chen (S'93-M'95-SM'01) received the B.S. and M.S. degrees in electrical engineering from National Tsing-Hua University, Taiwan, in 1985 and 1987, respectively, and the Ph.D. degree in electrical engineering from University of Maryland, College Park, in 1994. From 1985 to 1987, he was with Image Processing Laboratory in National Tsing-Hua University, where he worked on the recognition of Chinese characters. During 1989, he was with Star Tech. Inc., where he focused on the development of fingerprint recognition systems. After the reception of Ph.D. degree in 1994, he jointed Wan Ta Technology Inc. as a vice general manager, conducting several projects on Point-of-Sale systems. In 1995, he became a research staff in Advanced Technology Center, Computer and Communication Laboratory, Industrial Technology Research Institute in Taiwan, where he led a project on Java-based Network Managements. Since 1996, he has been an Associate Professor in Department of Communications Engineering at National Chiao Tung University (NCTU), Taiwan, and was promoted to a full professor in 2001. He was elected to be the Chair of IEEE Communications Society Taipei Chapter in 2006 and 2007, during which IEEE ComSoc Taipei Chapter won the 2007 IEEE ComSoc Chapter Achievement Awards (CAA) and 2007 IEEE ComSoc Chapter of the Year (CoY). He has served as the chairman of Department of Communications Engineering, NCTU, during 20072009. From 20122015, he was the associate chief director of Microelectronics and Information Systems Research Center, NCTU. In 2017, he became the Associate Dean of College of Electrical and Computer Engineering, NCTU.

Dr. Chen received the annual Research Awards from National Science Council, Taiwan, from 19962000, and received the 2000 Young Scholar Paper Award from Academia Sinica, Taiwan. His Experimental Handouts for the course of Communication Networks Laboratory have been awarded as the Annual Best Teaching Materials for Communications Education by Ministry of Education, Taiwan, in 1998. He has been selected as the Outstanding Tutor Teacher of NCTU in 2002, 2013, and 2014. He was also the recipient of Outstanding Teaching Award, NCTU, Taiwan, in 2021. His research interests generally lie in information and coding theory, large deviation theory, distributed detection and sensor networks.

Yunghsiang S. Han (S'90-M'93-SM'08-F'11) was born in Taipei, Taiwan, 1962. He received B.Sc. and M.Sc. degrees in electrical engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 1984 and 1986, respectively, and a Ph.D. degree from the School of Computer and Information Science, Syracuse University, Syracuse, NY, in 1993. He was from 1986 to 1988 a lecturer at Ming-Hsin Engineering College, Hsinchu, Taiwan. He was a teaching assistant from 1989 to 1992, and a research associate in the School of Computer and Information Science, Syracuse University from 1992 to 1993. He was, from 1993 to 1997, an Associate Professor in the Department of Electronic Engineering at Hua Fan College of Humanities and Technology, Taipei Hsien, Taiwan. He was with the Department of Computer Science and Information Engineering at National Chi Nan University, Nantou, Taiwan from 1997 to 2004. He was promoted to Professor in 1998. He was a visiting scholar in the Department of Electrical Engineering at University of Hawaii at Manoa, HI from June to October 2001, the SUPRIA visiting research scholar in the Department of Electrical Engineering and Computer Science and CASE center at Syracuse University, NY from September 2002 to January 2004 and July 2012 to June 2013, and the visiting scholar in the Department of Electrical and Computer Engineering at University of Texas at Austin, TX from August 2008 to June 2009. He was with the Graduate Institute of Communication Engineering at National Taipei University, Taipei, Taiwan from August 2004 to July 2010. From August 2010 to January 2017, he was with the Department of Electrical Engineering at National Taiwan University of Science and Technology as Chair Professor. From February 2017 to February 2021, he was with School of Electrical Engineering & Intelligentization at Dongguan University of Technology, China. Now he is with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China. He is also a Chair Professor at National Taipei University from February 2015. His research interests are in error-control coding, wireless networks, and security.

Dr. Han was a winner of the 1994 Syracuse University Doctoral Prize and a Fellow of IEEE. One of his papers won the prestigious 2013 ACM CCS Test-of-Time Award in cybersecurity.

Hanxu Hou received the B.Eng. degree in Information Security from Xidian University, Xian, China, in 2010, and Ph.D. degrees in the Dept. of Information Engineering from The Chinese University of Hong Kong in 2015 and in the School of Electronic and Computer Engineering from Peking University in 2016. He is now an Associate Professor with Dongguan University of Technology, and Honorary Post-doc of the Dept. of Computer Science and Engineering at the Chinese University of Hong Kong. He was a recipient of the 2020 Chinese Information Theory Young Rising Star Award by China Information Theory Society. He was recognized as an Exemplary Reviewer 2020 in IEEE Transactions on Communications. His research interests include erasure coding and coding for distributed storage systems.