Generalization of Minimum Storage Regenerating Codes for Heterogeneous Distributed Storage Systems

Zhengrui Li, *Student Member, IEEE,* Wai Ho Mow, *Senior Member, IEEE,* Yunghsiang S. Han *Fellow, IEEE,*, and Ting-Yi Wu, *Member, IEEE*

***Abstract*—Real-world distributed storage systems (DSSs) are heterogeneous because storage nodes may have unequal per-symbol storage costs, and network links may have unequal per-symbol transmission costs. For some general classes of heterogeneous DSSs, the optimal tradeoff between storage and repair costs achievable by functional repair codes is known (at least numerically). However, it is unclear whether exact-repair codes can achieve any point of such an optimal storage-repair tradeoff curve, especially at the point of the minimum storage cost. In this paper, we provide an affirmative answer to the question by constructing the so-called heterogeneous minimum storage repair (HMSR) codes for both the average and worst-case repair costs. To optimize storage and repair costs, a heterogeneous DSS may need to adopt irregular array codes and repair a node by downloading unequal numbers of symbols from helper nodes. However, our results show that for almost all heterogeneous DSSs, exact-repair HMSR codes are regular array codes covering an adequately chosen set of nodes. Specifically, exact-repair HMSR codes are designed by stacking conventional MSR codes and applying different repair schemes to different layers. Still, this does not work for every heterogeneous DSS. It is proven that using regular or linear irregular array codes for constructing exact-repair HMSR codes is insufficient in some cases.**

I. Introduction

The distributed storage system (DSS) is widely used to achieve high reliability, where data is encoded by some erasure code and stored in many connected unreliable storage nodes. In a DSS, a failed node can be repaired by downloading symbols from other available nodes, incurring a nonnegligible transmission cost. The minimization of the transmission cost has become a popular research topic in the last decade. In 2010, Dimakis *et al.* [2] introduced a new metric known as *repair bandwidth*, which can represent the transmission cost incurred by a single-node repair process and characterized the tradeoff curve between the storage capacity and the repair bandwidth by using the network coding theory [3]. In particular, the erasure codes corresponding to the two extreme points of the tradeoff curve are called minimum storage repair (MSR) codes and minimum bandwidth repair (MBR) codes, respectively. The two extreme points are called the MSR point and the MBR point, respectively. Besides MSR and MBR points, all points along the tradeoff curve can be achieved by leveraging network coding [3]. But network codes can only guarantee the *functional repair* of a failed node, which means that the new node (*newcomer*) replacing the failed node may have a different content from the failed node. Under functional repair, after getting the content of the newcomer, additional communication overheads among all nodes are required to update the repair schemes involving the newcomer. This may cause a significant delay. Additionally, in some applications requiring systematic codes, the functional repair will jeopardize the systematic form whenever a systematic node fails. Motivated by the limitations of functional repair, researchers started to consider exact-repair codes [4], especially the exact-repair MSR codes [5]–[9]. It has been proven that, between the functional and exact repair tradeoff curves, there is a gap for a wide range of parameters [10], but their MSR and MBR points coincide.

The research on the transmission cost of a repair process has been conducted under many different system models. Dimakis *et al.* [2] proposed the most prevalent model. This distributed storage system model, referred to as the *homogeneous model*, consists of n nodes with capacity α and allows (a) recovering any r failed nodes and (b) repairing a single failed node with the help of any d available nodes (called the helper nodes) by downloading β symbols from each helper node. Under the homogeneous model, the key issue is to optimize the per-node storage capacity α and the repair bandwidth defined as the number of symbols downloaded from helper nodes, i.e., βd. However, there are some limitations to the real-world applications of this model. In a real-world DSS, nodes may have unequal capacities, and the storage cost per symbol may vary from one node to another. That means we should not restrict different nodes to the same capacity to minimize the storage cost. Likewise, helper nodes can transmit different numbers of symbols to the newcomer, and the links from helper nodes to the newcomer can have unequal per-symbol transmission costs. That means if we want to minimize the transmission cost of a repair process, we should not restrict different helper nodes to transmitting the same number of

This work was supported by the Hong Kong Research Grants Council under General Research Fund (Project no. 16214422). An earlier version [1] of this paper was presented in part at the 2022 IEEE International Symposium on Information Theory.

Zhengrui Li and Wai Ho Mow are with the Department of Electrical and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong 999077, China (e-mail: zligt@connect.ust.hk, eewhmow@ust.hk).

Yunghsiang S. Han is with Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China (e-mail: yunghsiangh@gmail.com), and also with Theory Lab, Central Research Institute, 2012 Labs, Huawei Technology Co. Ltd., Hong Kong 999077, China (e-mail: han.yung.hsiang1@huawei.com).

Ting-Yi Wu is with Theory Lab, Central Research Institute, 2012 Labs, Huawei Technology Co. Ltd., Hong Kong 999077, China (e-mail: wu.ting.yi@huawei.com).

symbols to the newcomer.

Researchers have proposed different storage system models to relax such limitations of the homogeneous model and capture the heterogeneity of different real-world DSSs. References [11] and [12] allow the nodes to have unequal capacities. Reference [13] allows the nodes to have unequal per-symbol storage costs. Reference [14] allows the links to have unequal per-symbol transmission costs. References [11], [12], [14] allow the helper nodes to transmit different numbers of symbols to the newcomer. References [15] and [16] relax all the restrictions above of the homogeneous model and propose a model encompassing the models in [11]–[14] as special cases. Considering the unequal per-symbol storage costs and transmission costs of links, reference [16] generalizes storage capacity and repair bandwidth in the homogeneous model to the so-called storage cost and repair cost, respectively, and discusses their tradeoff. This model and the corresponding metrics have been followed by researchers [17], [18] in recent years. In all the storage system models mentioned above, the transmission from a helper node to the newcomer only involves a single hop. The storage system models allowing multi-hop transmissions have also been investigated in the literature [19]–[22]. We restrict our attention to the single-hop transmission in this paper.

This paper considers a more general model than that in [16]. We consider a fully connected DSS with n nodes, which can recover any r failed nodes. In this DSS, the nodes can have unequal per-symbol storage costs, and the links can have unequal per-symbol transmission costs. Such a DSS is denoted as an $(n, k, \mathbf{s}, \mathbf{t})$ DSS, where $k \triangleq n-r$, $\mathbf{s}$ is a vector consisting of the per-symbol storage costs of all nodes, and $\mathbf{t}$ is a vector consisting of the per-symbol transmission costs of all links.[1] If the per-symbol transmission cost only depends on the source node, this storage system model becomes the model in [16]. If nodes and links have uniform costs, this model becomes homogeneous. In this DSS, a file can be encoded by any erasure code, which allows retrieving the data file by accessing any k out of n nodes. Such erasure codes can be represented as irregular array codes [23]. For a file encoded by an irregular array code, one can easily get the storage cost from the size of the coded block in each node and the per-symbol storage cost of each node. Furthermore, the storage cost of the irregular array code is defined as the storage cost of the file normalized by the file size. As for the node failure issue of the DSS, we focus on the single-node failure and the single-node repair process, where the transmission from any helper node to the failed node involves one hop only. For an irregular array code, we define two types of repair costs: *average repair cost* and *worst-case repair cost*. The average repair cost of an irregular array code represents the average transmission cost to repair a node, and the worst-case repair cost represents the highest transmission cost among all transmissions from a helper node to a single failed node.

Note that the average repair cost of an erasure code is also defined in [16], [22], [24]. Reference [16] defines the storage cost as the same as in this paper and focuses on the average repair cost. By analyzing the min-cut of the information flow graph, Reference [16] derives the min-cut condition. The cut-set bound for functional-repair codes can be derived by formulating the problem of the storage-repair cost tradeoff as a bi-objective optimization problem subject to the min-cut condition. However, there is no polynomial time algorithm to solve this optimization problem. Whether this tradeoff curve, especially the minimum storage cost point, is achievable by an exact-repair code is also unknown. For our heterogeneous storage system model, the min-cut conditions are still valid and can be used to obtain the storage-repair tradeoff curve, at least numerically.

This paper focuses on the minimum storage cost point on the storage-repair tradeoff curve for our heterogeneous storage system model, called the heterogeneous minimum storage repair (HMSR) point, and the problem for constructing exact-repair codes to achieve this point. The contributions are as follows:

1) For any DSS, we divide all nodes into high-cost, moderate-cost, and low-cost nodes. It is shown that a DSS *almost surely*[2] has no moderate-cost node, and a DSS with at least one moderate-cost node *almost surely* has only one moderate-cost node. This motivates us to consider a DSS with no moderate-cost node and only one moderate-cost node.
2) Consider a DSS with no moderate-cost node. We show that for such a DSS, an HMSR code must be an MDS array code that only covers all low-cost nodes. For this case, exact-repair HMSR codes regarding either the average or worst-case repair cost are constructed. Our constructions leverage the conventional MSR codes, but the repair schemes need to be specially designed to download unequal numbers of symbols from helper servers according to the unequal link costs. These results demonstrate that by properly using conventional MSR codes and designing repair schemes, one can construct HMSR codes regarding either the average or worst-case repair cost for almost all heterogeneous DSS.
3) Consider a DSS with only one moderate-cost node. We show that for such a DSS, an HMSR code must be a two-valued array code[3] that only covers the moderate-cost and low-cost nodes. For such a DSS with $\mathbf{t}$ being an all-one vector, we obtain the HMSR point regarding the average repair cost and prove that any linear exact-repair code can not achieve it. On the other hand, we derive the HMSR point regarding the worst-case repair cost for such a DSS. It is shown that, for some $\mathbf{t}$, an HMSR code regarding the worst-case repair cost, which must be an MDS array code covering the moderate-cost and low-cost nodes, can be constructed. While for the other $\mathbf{t}$, the HMSR point cannot be achieved by regular array codes. These results demonstrate that for both repair costs, there exist some cases where the HMSR point cannot be achieved by regular array codes or linear

[1] $\mathbf{s}$ and $\mathbf{t}$ will be formally defined in Section II.

[2] "Almost surely" is a notion in probability theory. A property holds almost surely means the set that the property holds takes up nearly all possibilities.

[3] The two-valued array code is formally defined in Section V.

exact-repair codes.

The rest of the paper is organized as follows. Section II formally defines the storage system model and introduces the considered problem. Section III defines the high-cost, moderate-cost, and low-cost nodes and divides DSSs into categories according to the number of moderate-cost nodes. Section IV obtains HMSR codes for a DSS with no moderate-cost node. Section V investigates HMSR codes for a DSS with only one moderate-cost node. Section VI concludes this work.

II. System Model and Problem Statement

Consider a distributed storage system represented by a fully connected storage network of n nodes. Each node $i \in [n]$ has associated storage costs $s_i > 0$, representing the cost of storing one symbol. We can assume $s_1 \geq \cdots \geq s_n$ due to the symmetry of the system. Let $i \to j$ denote the network link from node i to node j. Each link $i \to j$, with $i, j \in [n]$ and $i \neq j$, has an associated transmission cost $t_{i,j} > 0$ representing the cost of transmitting one symbol. To maintain data integrity, the DSS encodes every file and stores the coded symbols distributively so that one can retrieve every file by accessing any k out of n nodes. Let $\mathbf{s} \triangleq [s_1 \ldots s_n]$, $\mathbf{t}_i \triangleq [t_{j,i}]_{j\in[n]\setminus\{i\}}$, and $\mathbf{t} \triangleq [\mathbf{t}_1 \ldots \mathbf{t}_n]$. Such a DSS is called an $(n, k, \mathbf{s}, \mathbf{t})$ DSS.

Consider storing a file with B data symbols in an $(n, k, \mathbf{s}, \mathbf{t})$ DSS. After being encoded by an erasure code, $\alpha_i B$ coded symbols are allocated in each node i for $i \in [n]$. Since the erasure code should enable us to retrieve the original file from any k out of n nodes, we can obtain some necessary conditions of $\{\alpha_i\}_{i\in[n]}$, which are

$$\sum_{i\in\mathcal{S}} \alpha_i \geq 1 \quad \forall \mathcal{S} \subset [n] \text{ with } |\mathcal{S}| = k. \tag{1}$$

As different nodes may store unequal numbers of coded symbols, such a code is called an (n, k) irregular array code with data size B and a data allocation vector $\boldsymbol{\alpha} \triangleq [\alpha_1 \ldots \alpha_n]$. Specifically, if $\boldsymbol{\alpha} = [\frac{1}{k} \ldots \frac{1}{k}]$, an (n, k) irregular array code is an (n, k) MDS array code. We define the storage cost of an (n, k) irregular array code with a data allocation vector $\boldsymbol{\alpha}$ as

$$C_S(\boldsymbol{\alpha}) \triangleq \sum_{i\in[n]} s_i \alpha_i,$$

which can be seen as the per-data-symbol storage cost of the file.

When a single node fails, we focus on the repair process that only allows one-hop transmission from a helper node to the newcomer. Unless otherwise stated, we allow the functional repair. For an (n, k) irregular array code with data size B, let $\beta_{j,i} B$ denote the number of symbols transmitted to a single failed node i from node j for all $j \in [n] \setminus \{i\}$. Let $\boldsymbol{\beta_i} \triangleq [\beta_{j,i}]_{j\in[n]\setminus\{i\}}$ and $\boldsymbol{\beta} \triangleq [\boldsymbol{\beta}_i]_{i\in[n]}$. We define two types of repair costs for an (n, k) irregular array code.

1) We define the total repair cost of node i as

$$C_{R_i}^{tot}(\boldsymbol{\beta_i}) \triangleq \sum_{j\in[n]\setminus\{i\}} t_{j,i}\beta_{j,i},$$

which is the total transmission cost of the repair process of node i normalized by the data file size B. We define the average repair cost of the code as

$$C_R^{ave}(\boldsymbol{\beta}) \triangleq \frac{1}{n} \sum_{i\in[n]} C_{R_i}^{tot}(\boldsymbol{\beta_i}).$$

2) Considering the highest transmission cost from helper nodes, we define the worst-case repair cost of node i as

$$C_{R_i}^{wor}(\boldsymbol{\beta_i}) \triangleq \max_{j\in[n]\setminus\{i\}} t_{j,i}\beta_{j,i}.$$

Furthermore, to characterize the worst performance of all single-node repair processes, we define the worst-case repair cost of the code as

$$C_R^{wor}(\boldsymbol{\beta}) \triangleq \max_{i\in[n]} C_{R_i}^{wor}(\boldsymbol{\beta_i}).$$

This paper focuses on the problem of constructing exact-repair HMSR codes for an $(n, k, \mathbf{s}, \mathbf{t})$ DSS. For an $(n, k, \mathbf{s}, \mathbf{t})$ DSS, HMSR codes are the (n, k) irregular array codes with the minimum storage cost and the optimal repair cost subject to the minimum storage cost. Note that both average and worst-case repair costs are considered. Thus, we consider two types of HMSR codes regarding the average and worst-case repair cost, respectively.

Table I presents the main symbols and definitions related to a DSS. Some of these symbols and definitions will be introduced in Sections III and IV.

III. Data allocation vectors of HMSR codes and categories of DSSs

This section characterizes $\boldsymbol{\alpha}$ of HMSR codes by minimizing the storage cost of an (n, k) irregular array code in an $(n, k, \mathbf{s}, \mathbf{t})$ DSS. For an $(n, k, \mathbf{s}, \mathbf{t})$ DSS, $\boldsymbol{\alpha}$ leading to the minimum storage cost is called an *optimal data allocation vector*. The complete characterization of the optimal data allocation vectors is important because the HMSR codes are the ones with the optimal repair cost among the codes with an optimal data allocation vector. Different $\mathbf{s}$ may lead to different types of optimal data allocation vectors. Specifically, for some $\mathbf{s}$, the optimal data allocation vector is unique, which means the data allocation vector of an HMSR must be it. Thus, one can obtain an HMSR code by optimizing the repair cost subject to this data allocation vector. For some $\mathbf{s}$, since there are many optimal data allocation vectors, the data allocation vector of an HMSR code is not fixed, which makes finding HMSR codes more difficult. This motivates us to divide DSSs into several categories according to different types of optimal data allocation vectors (cf. subsection B).

A. Optimal data allocation vectors

The minimum storage cost of an (n, k) irregular array code is determined by the following problem.

TABLE I
MAIN NOTATION RELATED TO A DSS

Symbol	Definition
s_i	The per-symbol storage cost of node i
$t_{j,i}$	The per-symbol transmission cost from node j to node i
Helper node j for node i	The node whose link to node i has the j-th highest per-symbol transmission cost
$\bar{t}_{j,i}$	The per-symbol transmission cost from helper node j for node i to node i
α_i	The normalized number of symbols stored in node i
$\beta_{j,i}$	The normalized number of symbols transmitted to the failed node i from node j
$\bar{\beta}_{j,i}$	The normalized number of symbols transmitted to the failed node i from helper node j for node i
$\mathcal{N}_H, \mathcal{N}_M, \mathcal{N}_L$	The index sets of the high-cost, moderate-cost, and low-cost nodes, respectively
$\mathcal{H}_H^{(i)}, \mathcal{H}_M^{(i)}, \mathcal{H}_L^{(i)}$	The index sets of the high-cost, moderate-cost, and low-cost helper nodes for the failed node i, respectively

Problem 1: Given n, k, and $\mathbf{s} = [s_1 \, \dots \, s_n]$,

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \quad \sum_{i \in [n]} s_i \alpha_i$$

$$\text{s.t.} \quad \alpha_i \geq 0 \quad \forall i \in [n] \tag{2}$$

$$\sum_{i \in \mathcal{S}} \alpha_i \geq 1 \quad \forall \mathcal{S} \subseteq [n] \text{ with } |\mathcal{S}| = k. \tag{3}$$

It is proven in [25] that conditions (2) and (3) form a sufficient and necessary condition to make $\boldsymbol{\alpha}$ a data allocation vector of an (n, k) irregular array code. Hence, the optimal value to Problem 1 is the minimum storage cost of (n, k) irregular array codes, and the optimal solutions of Problem 1 are all optimal data allocation vectors.

Next, we derive the optimal data allocation vectors. Since an $(n, k, \mathbf{s}, \mathbf{t})$ DSS can tolerate any $n - k$ node failures, the system should have at least $n - k + 1$ non-empty nodes. To minimize the storage cost, the non-empty nodes should have smaller per-symbol storage costs. Therefore, as $s_1 \geq \cdots \geq s_n$, node i with $i = k, \dots, n$ must be a non-empty node. For a node i with $i \in [k-1]$, we define

$$y_i(n, k, \mathbf{s}) \triangleq (k - i)s_i - \sum_{t=i+1}^{n} s_t. \tag{4}$$

Based on $y_i(n, k, \mathbf{s})$ for all $i \in [k-1]$, we divide all nodes into three sets.

$$\mathcal{N}_H \triangleq \{i \in [k-1] | y_i(n, k, \mathbf{s}) > 0\}, \tag{5}$$

$$\mathcal{N}_M \triangleq \{i \in [k-1] | y_i(n, k, \mathbf{s}) = 0\}, \tag{6}$$

$$\mathcal{N}_L \triangleq \{i \in [k-1] | y_i(n, k, \mathbf{s}) < 0\} \cup ([n] \setminus [k-1]). \tag{7}$$

The following theorem, which is obtained by solving Problem 1, provides a complete characterization of the optimal data allocation vectors.

Theorem 1: For an $(n, k, \mathbf{s}, \mathbf{t})$ DSS, $\boldsymbol{\alpha} = [\alpha_1 \dots \alpha_n]$ is an optimal data allocation vector if and only if

$$\alpha_i = 0 \quad \forall i \in \mathcal{N}_H, \tag{8}$$

$$0 \leq \alpha_i \leq \alpha_k \quad \forall i \in \mathcal{N}_M, \tag{9}$$

$$\alpha_i = \alpha_k \quad \forall i \in \mathcal{N}_L, \tag{10}$$

$$\sum_{i \in [k]} \alpha_i = 1. \tag{11}$$

Theorem 1 indicates that to minimize the storage cost, (a) node i for all $i \in \mathcal{N}_H$ must store no symbols; (b) node i for all $i \in \mathcal{N}_L$ must store the same number of symbols; (c) node i for all $i \in \mathcal{N}_L$ can store symbols no more than that in any node i with $i \in \mathcal{N}_L$. To improve readability, we use the following terminology: node i for all $i \in \mathcal{N}_H$ is referred to as a high-cost node, node i for all $i \in \mathcal{N}_M$ is referred to as a moderate-cost node, node i for all $i \in \mathcal{N}_L$ is referred to as a low-cost node.

Consider a DSS with $n = 6$, $k = 4$, and $\mathbf{s} = [4\,3\,2\,2\,1\,1]$. For this DSS, $y_1(6, 4, \mathbf{s}) = 3s_1 - \sum_{t=2}^{6} s_t = 3 > 0$, $y_2(6, 4, \mathbf{s}) = 2s_2 - \sum_{t=3}^{6} s_t = 0$, and $y_3(6, 4, \mathbf{s}) = s_3 - \sum_{t=4}^{6} s_t = -2 < 0$, which leads to $\mathcal{N}_H = \{1\}$, and $\mathcal{N}_M = \{2\}$, $\mathcal{N}_L = \{3, 4, 5, 6\}$. Thus, node 1 is a high-cost node, node 2 is a moderate-cost node, and nodes 3 to 6 are low-cost nodes. To minimize the storage cost, node 1 must be an empty node, nodes $3, \dots, 6$ must store the same number of symbols, and node 2 must store symbols no more than that in node 3. Furthermore, the optimal data allocation vectors must satisfy (11). Thus, $[0\,0\,\frac{1}{2}\,\frac{1}{2}\,\frac{1}{2}\,\frac{1}{2}]$, $[0\,\frac{1}{4}\,\frac{3}{8}\,\frac{3}{8}\,\frac{3}{8}\,\frac{3}{8}]$, and $[0\,\frac{1}{3}\,\frac{1}{3}\,\frac{1}{3}\,\frac{1}{3}\,\frac{1}{3}]$ are all optimal data allocation vectors. To illustrate how to solve Problem 1 and obtain the optimal data allocation vectors (cf. Theorem 1), we give a data allocation vector $\boldsymbol{\alpha}$ and show how to modify this $\boldsymbol{\alpha}$ to reduce the storage cost and eventually obtain an $\boldsymbol{\alpha}$ satisfying the condition given in Theorem 1. To minimize the storage cost, we should store more symbols in a node with a smaller per-symbol storage cost, which means we should focus on the solutions with $\alpha_1 \leq \cdots \leq \alpha_n$. Thus, we choose $\boldsymbol{\alpha} = [1\,2\,3\,4\,5\,6]$ and show how to modify this $\boldsymbol{\alpha}$ to reduce the storage cost while meeting condition (3).

1) Condition (3) can be rewritten as $\min_{\mathcal{S} \subseteq [n], |\mathcal{S}| = k} \sum_{i \in \mathcal{S}} \alpha_i \geq 1$. Given $\boldsymbol{\alpha} = [1\,2\,3\,4\,5\,6]$, we have $\min_{\mathcal{S} \subseteq [6], |\mathcal{S}| = 4} \sum_{i \in \mathcal{S}} \alpha_i = \sum_{i=1}^{4} \alpha_i = 10$, which is larger than 1. To reduce the storage cost, we can let $\sum_{i=1}^{4} \alpha_i$ be 1 by dividing each α_i by 10. The data allocation vector becomes $\boldsymbol{\alpha} = [\frac{1}{10}\,\frac{2}{10}\,\frac{3}{10}\,\frac{4}{10}\,\frac{5}{10}\,\frac{6}{10}]$ satisfying (11). The storage cost can be further reduced by letting α_5 and α_6 be $\frac{4}{10}$. Thus, the data allocation vector becomes $\boldsymbol{\alpha} = [\frac{1}{10}\,\frac{2}{10}\,\frac{3}{10}\,\frac{4}{10}\,\frac{4}{10}\,\frac{4}{10}]$ satisfying (11), and $\alpha_i = \alpha_k$ for all $i \in [n] \setminus [k-1]$.
2) For high-cost node 1, we have $y_1(6, 4, \mathbf{s}) = 3s_1 - \sum_{t=2}^{6} s_t > 0$, which means $s_1 > \frac{1}{3} \sum_{t=2}^{6} s_t$. If we reduce α_1 by $\frac{1}{10}$ and increase the other α_i by $\frac{1}{30}$, $\boldsymbol{\alpha}$ becomes $[0\,\frac{7}{30}\,\frac{10}{30}\,\frac{13}{30}\,\frac{13}{30}\,\frac{13}{30}]$, which still satisfies condition (3). Since $s_1 > \frac{1}{3} \sum_{t=2}^{6} s_t$, this new $\boldsymbol{\alpha}$ has smaller

storage cost. Additionally, this new $\boldsymbol{\alpha}$ satisfies (8).

3) For low-cost node 3, we have $y_3(6,4,\mathrm{s}) = s_3 - \sum_{t=4}^{6} s_t < 0$, which means $s_3 < \sum_{t=4}^{6} s_t$. If we increase α_3 by $\frac{3}{60}$ and reduce α_i with $i = 4,5,6$ by $\frac{3}{60}$, $\boldsymbol{\alpha} = [0\ \frac{7}{30}\ \frac{10}{30}\frac{13}{30}\ \frac{13}{30}\ \frac{13}{30}]$ becomes $[0\ \frac{7}{30}\ \frac{23}{60}\ \frac{23}{60}\ \frac{23}{60}\ \frac{23}{60}]$, which still satisfies condition (3). Since $s_3 < \sum_{t=4}^{6} s_t$, this new $\boldsymbol{\alpha}$ has smaller storage cost. Additionally, this new $\boldsymbol{\alpha}$ satisfies (10).
4) $\boldsymbol{\alpha} = [0\ \frac{7}{30}\ \frac{23}{60}\ \frac{23}{60}\ \frac{23}{60}\ \frac{23}{60}]$ satisfies the condition given in Theorem 1, but is not the unique optimal $\boldsymbol{\alpha}$. For moderate-cost node 2, we have $y_2(6,4,\mathrm{s}) = 2s_2 - \sum_{t=3}^{6} s_t = 0$, which means $s_2 = \frac{1}{2}\sum_{t=3}^{6} s_t$. If we reduce α_2 by x and increase α_i with $i = 3,4,5,6$ by $\frac{x}{2}$, the $\boldsymbol{\alpha}$ becomes $[0\ \ \frac{7}{30} - x\ \ \frac{23}{60} + \frac{x}{2}\ \ \frac{23}{60} + \frac{x}{2}\ \ \frac{23}{60} + \frac{x}{2}\ \ \frac{23}{60} + \frac{x}{2}]$, which still meets condition (3). Since $s_2 = \frac{1}{2}\sum_{t=3}^{6} s_t$, this new $\boldsymbol{\alpha}$ has the same optimal storage cost. Thus, if we increase α_2 by x and reduce α_i for all $i = 3,4,5,6$ by $\frac{x}{2}$, we still obtain an optimal $\boldsymbol{\alpha}$ as long as we guarantee $\alpha_2 \le \alpha_3$.

The detailed proof of Theorem 1 is given in Appendix A.

B. Categories of DSSs

Theorem 1 indicates that if there is no moderate-cost node, the optimal data allocation vector is unique; while if there are more moderate-cost nodes, there are more optimal data allocation vectors. Different numbers of moderate-cost nodes lead to different cases of optimal data allocation vectors, which lead to different problems for obtaining HMSR codes (cf. Sections IV and V). Note that there are at most $k-1$ moderate-cost nodes. To distinguish different cases, we divide (n,k) DSSs into k categories: DSSs with t moderate-cost nodes with $t = 0,\dots,k-1$.

Among the k categories, DSS with no moderate-cost node is the most important one because almost all DSSs have no moderate-cost node. Before explaining why, we suggest readers recall the definition of the moderate-cost node and $\mathcal{N}_M$ (cf. (6)). For an (n,k) DSS, we can set the value of s_i for all $i \in [n]$ freely[4]. The dimension of s space is n. But if there is one moderate-cost node, for example node 1, s must satisfy one equation $y_1(n,k,\mathrm{s}) = (k-1)s_1 - \sum_{t=2}^{n} s_t = 0$, which means we can only set the values of $s_2,\dots,s_n$ freely, and s_1 will be determined accordingly. The dimension of the corresponding s space is $n-1$. Furthermore, if there are two moderate-cost nodes, s must satisfy two equations, which means the dimension of the corresponding s space is $n-2$. Like a line has no area and a plane has no volume, a space of dimension $n-1$ has zero measure [26] in a space of dimension n. Thus, we have

Theorem 2: A DSS *almost surely* has no moderate-cost node. A DSS with at least one moderate-cost node *almost surely* has only one moderate-cost node.

Proof: The set of all possible s is denoted as $\boldsymbol{\mathcal{S}} \triangleq \{[s_1 \dots s_n] \in \mathbb{R}^n | s_1 \ge \dots \ge s_n > 0\}$. Let $\boldsymbol{\mathcal{S}}_i \triangleq \{\mathrm{s} \in \boldsymbol{\mathcal{S}} || \mathcal{N}_M| = i\}$ for $i = 0,1$. We will prove $\mathrm{s} \in \boldsymbol{\mathcal{S}}_0$ is *almost everywhere* in the set $\boldsymbol{\mathcal{S}}$. Similarly, one can prove that $\mathrm{s} \in \boldsymbol{\mathcal{S}}_1$ is *almost everywhere* in the set $\boldsymbol{\mathcal{S}}_1$. "Almost everywhere" is a notion in measure theory [26] and is analogous to the notion of almost surely in probability theory. A property holding almost everywhere means the set that the property holds takes up nearly all possibilities. Note that a property holds almost everywhere if it holds for all elements in a set except a subset of measure zero [26]. That means we only need to prove that $\boldsymbol{\mathcal{S}} \setminus \boldsymbol{\mathcal{S}}_0$ is a set of measure zero in $\boldsymbol{\mathcal{S}}$. Next, we show that the dimension of $\boldsymbol{\mathcal{S}} \setminus \boldsymbol{\mathcal{S}}_0$ is less than that of $\boldsymbol{\mathcal{S}}$, which implies $\boldsymbol{\mathcal{S}} \setminus \boldsymbol{\mathcal{S}}_0$ has zero measure. Note that $\boldsymbol{\mathcal{S}} = \{[s_1 \dots s_n] \in \mathbb{R}_+^n | s_1 \ge \dots \ge s_n\}$. Clearly, $\boldsymbol{\mathcal{S}} \cup \{[0\dots 0]\}$ is an n-dimensional convex cone. In $\boldsymbol{\mathcal{S}}$, $|\mathcal{N}_M| = 0$ happens unless $[s_1 \dots s_n]$ satisfies $is_{k-i} = \sum_{t=k-i+1}^{n} s_t$ for some $i \in [k-1]$. Let $\mathcal{C}_i = \{[s_1 \dots s_n] \in \boldsymbol{\mathcal{S}} | is_{k-i} = \sum_{t=k-i+1}^{n} s_i\}$. Consider $\mathcal{D}_i = \{[s_1 \dots s_n] \in \mathbb{R}^n | is_{k-i} = \sum_{t=k-i+1}^{n} s_i\}$, which is apparently an $(n-1)$-dimensional subspace of $\mathbb{R}^n$. Thus, $\mathcal{D}_i$ has measure 0 in $\mathbb{R}^n$. As $\mathcal{C}_i \subseteq \mathcal{D}_i$, $\mathcal{C}_i$ has measure 0, which implies $\cup_{i\in[k-1]}\mathcal{C}_i$ also has measure 0. Thus, in the n-dimensional convex cone $\boldsymbol{\mathcal{S}} \cup \{[0\dots 0]\}$, $|\mathcal{N}_M| = 0$ happens except on the set $(\cup_{i\in[k-1]}\mathcal{C}_i) \cup \{[0\dots 0]\}$ of measure 0, which means $|\mathcal{N}_M| = 0$ is *almost everywhere* in $\boldsymbol{\mathcal{S}}$. ■

Theorem 2 motivates us to discuss the two most important cases in the rest of the paper: a DSS with no moderate-cost node and a DSS with only one moderate-cost node.

Remark 1: Consider an (n,k) DSS and an $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS induced by all low-cost and moderate-cost nodes in the (n,k) DSS. In the (n,k) DSS, if the high-cost nodes are empty, they do not need to be protected, and the system only needs to tolerate any $n-k$ node failures among the low-cost and moderate-cost nodes. Consequently, a code that is designed for the (n,k) DSS but only covers the low-cost and moderate-cost nodes can be seen as a code for the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS, and vice versa. Theorem 1 has shown that an HMSR code $\mathcal{C}$ for the (n,k) DSS only covers the low-cost and moderate-cost nodes. This implies that $\mathcal{C}$ can be seen as a code for the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS. Furthermore, it is easy to show by contradiction that an HMSR code for the (n,k) DSS is also an HMSR code for the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS, and vice versa. That means we only need to find the HMSR code for the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS. Additionally, it is easy to verify that the moderate-cost nodes in (n,k) DSS are still moderate-cost nodes in the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS, and the low-cost nodes in (n,k) DSS are still low-cost nodes in the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS. Therefore, we only need to find the HMSR code for the $(|\mathcal{N}_L| + |\mathcal{N}_M|, |\mathcal{N}_L| + |\mathcal{N}_M| - n + k)$ DSS, which only contains low-cost and moderate-cost nodes.

IV. HMSR CODES FOR A DSS WITH NO MODERATE-COST NODE

A DSS with no moderate-cost node has been proven to be the most important case (cf. Theorem 2). Remark 1 indicates that HMSR codes for this DSS are equivalent to the HMSR codes for the sub-DSS induced by low-cost nodes. Thus,

[4]Although we assume $s_1 \ge \dots \ge s_n$, we can first choose $s_i' \ge 0$ for all $i \in [n]$, and then obtain $s_n = s_n'$ and $s_i = s_{i+1} + s_i'$ for all $i \in [n-1]$. Thus, in this sense, s_i can be chosen freely.

WLOG, we consider an (n,k) DSS with only low-cost nodes. From Theorem 1, this DSS has the unique optimal data allocation vector, which has $\alpha_i = \frac{1}{k}$ for all $i \in [n]$. This unique optimal data allocation vector indicates that HMSR codes are (n,k) MDS array codes with the optimal repair cost. Before discussing such codes, we introduce more definitions related to a DSS for the convenience of the discussion. Given an arbitrarily failed node i, we index the $n-1$ helper nodes as helper nodes 1 to $n-1$ so that the per-symbol transmission cost from helper nodes 1 to $n-1$ to the failed node is in descending order. Thus, helper node j for node i is the node whose link to node i has the j-th highest per-symbol transmission cost, which implies that helper node j may not be node j. The per-symbol transmission cost associated with helper node j for node i is denoted as $\bar{t}_{j,i}$. Thus, $\bar{t}_{1,i} \geq \cdots \geq \bar{t}_{n-1,i} > 0$. Let $\bar{\mathbf{t}}_i \triangleq [\bar{t}_{j,i}]_{j\in[n-1]}$, and $\bar{\mathbf{t}} \triangleq [\bar{\mathbf{t}}_1 \ \ldots \ \bar{\mathbf{t}}_n]$. Note that given $\mathbf{t} = [\mathbf{t}_1 \ \ldots \ \mathbf{t}_n]$, $\bar{\mathbf{t}} = [\bar{\mathbf{t}}_1 \ \ldots \ \bar{\mathbf{t}}_n]$ can be obtained by reordering the elements in each $\mathbf{t}_i$ in descending order. Moreover, let $\bar{\beta}_{j,i}B$ denote the number of symbols transmitted to the failed node i from helper node j with $j \in [n-1]$. Let $\bar{\boldsymbol{\beta}}_{\boldsymbol{i}} \triangleq [\bar{\beta}_{j,i}]_{j\in[n-1]}$ and $\bar{\boldsymbol{\beta}} \triangleq [\bar{\boldsymbol{\beta}}_i]_{i\in[n]}$. Thus, the total and worst-case repair costs of node i are $\sum_{j\in[n-1]} \bar{t}_{j,i}\bar{\beta}_{j,i}$ and $\max_{j\in[n-1]} \bar{t}_{j,i}\bar{\beta}_{j,i}$, respectively. For (n,k) MDS array codes, the min-cut condition in [16] can be equivalently simplified to

$$\begin{aligned} &\bar{\beta}_{j,i} \geq 0 \quad \forall i \in [n], j \in [n-1], \\ &\sum_{j\in\mathcal{S}} \bar{\beta}_{j,i} \geq \frac{1}{k} \quad \forall i \in [n], \forall \mathcal{S} \subseteq [n-1] \text{ with } |\mathcal{S}| = n-k. \end{aligned} \tag{12}$$

This $\bar{\boldsymbol{\beta}}$ condition for MDS codes is also shown in [21], [27] via different approaches. From the network coding theory, if allowing functional repair, the min-cut condition, which is (12) for (n,k) MDS array codes, is a sufficient and necessary condition of $\bar{\boldsymbol{\beta}}$. Thus, optimizing repair cost subject to this min-cut condition leads to the optimal repair cost for MDS array codes, i.e., the repair cost of HMSR codes for the considered DSS. Specifically, the average and worst-case repair costs are considered in the following two subsections. Each subsection first fully characterizes $\bar{\boldsymbol{\beta}}$ of an HMSR code and then constructs an exact-repair HMSR code with such $\bar{\boldsymbol{\beta}}$.

A. HMSR codes regarding the average repair cost

1) Characterization of $\bar{\boldsymbol{\beta}}$: From the min-cut condition (12), the optimal average repair cost for an (n,k) MDS array code, i.e., the repair cost for an HMSR code regarding the average repair cost, is given by the following optimization problem.

Problem 2.A: Given n, k, and $\bar{\mathbf{t}}$,

$$\min_{\bar{\boldsymbol{\beta}}\in\mathbb{R}^{n(n-1)}} \quad \frac{1}{n}\sum_{i\in[n]}\sum_{j\in[n-1]} \bar{t}_{j,i}\bar{\beta}_{j,i}$$

$$\text{s.t.} \quad \bar{\beta}_{j,i} \geq 0 \quad \forall i \in [n], j \in [n-1], \tag{13}$$

$$\sum_{j\in\mathcal{S}} \bar{\beta}_{j,i} \geq \frac{1}{k} \quad \forall i \in [n], \forall \mathcal{S} \subseteq [n-1] \text{ with } |\mathcal{S}| = n-k. \tag{14}$$

In Problem 2.A, the objective function is a summation of $\sum_{j\in[n-1]} \bar{t}_{j,i}\bar{\beta}_{j,i}$ for all $i \in [n]$ and each constraint is only related to one $\bar{\boldsymbol{\beta}}_i$. For example, the constraints $\bar{\beta}_{2,1} \geq 0$ and $\sum_{j=1}^{n-k} \bar{\beta}_{j,1} \geq 1/k$ are only related to $\bar{\boldsymbol{\beta}}_1$. Thus, Problem 2.A can be decoupled into n optimization problems associated with n nodes as follows.

Problem 2.B: Given n, k, a node index i, and $\bar{\mathbf{t}}_i$,

$$\min_{\bar{\boldsymbol{\beta}}_i\in\mathbb{R}^{n-1}} \quad \sum_{j\in[n-1]} \bar{t}_{j,i}\bar{\beta}_{j,i}$$

$$\text{s.t.} \quad \bar{\beta}_{j,i} \geq 0 \quad \forall j \in [n-1], \tag{15}$$

$$\sum_{j\in\mathcal{S}} \bar{\beta}_{j,i} \geq \frac{1}{k} \quad \forall \mathcal{S} \subseteq [n-1] \text{ with } |\mathcal{S}| = n-k. \tag{16}$$

The optimal value of Problem 2.B associated with node i is the optimal total repair cost for node i. Since there is no common variable in Problem 2.B associated with different nodes, the optimal total repair cost for node i for all $i \in [n]$ can be achieved simultaneously. That implies that the average of the optimal total repair cost for all nodes is the optimal average repair cost. Conversely, to achieve the optimal average repair cost, each node must have the optimal total repair cost. Thus, $\bar{\boldsymbol{\beta}}$ for MDS array codes with optimal average repair cost can be fully characterized by the optimal solutions to Problem 2.B associated with all nodes. Surprisingly, Problem 2.B and Problem 1 are equivalent and can be solved similarly. The reasons are as follows: (a) Both the objective function of Problem 1 and the objective function of Problem 2.B are weighted sums of all variables. (b) Both the first constraint (cf. (2)) of Problem 1 and the first constraint (cf. (15)) of Problem 2.B require the variables to be non-negative numbers. (c) Both the second constraint (cf. (3)) of Problem 1 and the second constraint (cf. (16)) of Problem 2.B require that the sum of a certain number of variables must be at least some constant. Thus, optimizing the storage cost and optimizing the total repair cost for a node are equivalent problems. To optimize the storage cost, each storage node is labeled as a high-cost, moderate-cost, or low-cost node, and the system treats different types of nodes differently (cf. Theorem 1). Likewise, each helper node can be labeled as a high-cost, moderate-cost, or low-cost helper node according to the associated per-symbol transmission cost. To optimize the total repair cost for node i, the system must treat different types of helper nodes differently. For example, the system must not download any symbols from a high-cost helper node. Specifically, for any $i \in [n]$, let

$$\begin{aligned} \mathcal{H}_L^{(i)} \triangleq &\{j \in [n-k-1] \mid y_j(n-1, n-k, \bar{\mathbf{t}}_i) < 0\} \\ &\cup ([n-1] \setminus [n-k-1]), \end{aligned} \tag{17}$$

$$\mathcal{H}_M^{(i)} \triangleq \{j \in [n-k-1] \mid y_j(n-1, n-k, \bar{\mathbf{t}}_i) = 0\}, \tag{18}$$

$$\mathcal{H}_H^{(i)} \triangleq \{j \in [n-k-1] \mid y_j(n-1, n-k, \bar{\mathbf{t}}_i) > 0\}, \tag{19}$$

which are referred to as the set of low-cost, moderate-cost, and high-cost helper nodes for node i, respectively. Formally, from Theorem 1, we have the following theorem fully characterizing $\bar{\boldsymbol{\beta}}$ for MDS array codes with optimal average repair cost, i.e., HMSR codes regarding the average repair cost.

Theorem 3: For an $(n,k,\mathbf{s},\mathbf{t})$ DSS with only low-cost nodes, HMSR codes regarding average repair cost must be an (n, k) MDS array code, and the $\bar{\boldsymbol{\beta}}$ of HMSR codes can be fully characterized by

$$\begin{aligned}
&\bar{\beta}_{j,i} = 0 \quad \forall i \in [n], j \in \mathcal{H}_H^{(i)}, \\
&0 \leq \bar{\beta}_{j,i} \leq \bar{\beta}_{n-k,i} \quad \forall i \in [n], j \in \mathcal{H}_M^{(i)}, \\
&\bar{\beta}_{j,i} = \bar{\beta}_{n-k,i} \quad \forall i \in [n], j \in \mathcal{H}_L^{(i)}, \\
&\sum_{j \in [n-k]} \bar{\beta}_{j,i} = \frac{1}{k} \quad \forall i \in [n].
\end{aligned} \tag{20}$$

Proof: Eq. (20) is obtained by solving Problem 2.B. We have roughly shown that Problem 2.B and Problem 1 are equivalent. Thus, the solution to Problem 1 can be transformed into the solution to Problem 2.B. This proof demonstrates the details of this transforming process. Problem 2.B has the same optimal solution set as a new problem that minimizes $k \sum_{j \in [n-1]} \bar{t}_{j,i} \bar{\beta}_{j,i}$ subject to

$$k\bar{\beta}_{j,i} \geq 0 \quad \forall j \in [n-1], \tag{21}$$

$$\sum_{j \in \mathcal{S}} k\bar{\beta}_{j,i} \geq 1 \quad \forall \mathcal{S} \subseteq [n-1] \text{ with } |\mathcal{S}| = n-k, \tag{22}$$

since the objective function of the new problem is k times the objective function of Problem 2.B and the constraints (21) and (22) are equivalent to the constraints (15) and (16). By comparing this new problem with Problem 1, we conclude that one can convert Problem 1 into this problem by replacing the parameters n, k, $\mathbf{s}$, and the variable vector $\boldsymbol{\alpha}$ with $n-1$, $n-k$, $\bar{\mathbf{t}}_i$, and $k\bar{\boldsymbol{\beta}}_i$. As such, the solution to Problem 1 can be converted to the solution to Problem 2.B. By replacing n, k, $\mathbf{s}$, and $\boldsymbol{\alpha}$ in the optimal solution to Problem 1 (cf. Theorem 1) with $n-1$, $n-k$, $\bar{\mathbf{t}}_i$ and $k\bar{\boldsymbol{\beta}}_i$, $\mathcal{N}_L$, $\mathcal{N}_M$, and $\mathcal{N}_H$ become $\mathcal{H}_L^{(i)}$, $\mathcal{H}_M^{(i)}$, and $\mathcal{H}_H^{(i)}$. Furthermore, from Theorem 1, we obtain the optimal solution to Problem 2.B as follows:

$$\begin{aligned}
&\bar{\beta}_{j,i} = 0 \quad \forall j \in \mathcal{H}_H^{(i)}, \\
&0 \leq \bar{\beta}_{j,i} \leq \bar{\beta}_{n-k,i} \quad j \in \mathcal{H}_M^{(i)}, \\
&\bar{\beta}_{j,i} = \bar{\beta}_{n-k,i} \quad \forall j \in \mathcal{H}_L^{(i)}, \\
&\sum_{j \in [n-k]} \bar{\beta}_{j,i} = \frac{1}{k}.
\end{aligned} \tag{23}$$

This completes the proof. ■

Before describing general code construction and repair schemes, we give an example showing how to achieve the optimal average repair cost for MDS array codes. Consider $(n = 6, k = 3)$ MDS array codes in a $(6, 3, \mathbf{s}, \mathbf{t})$ DSS with $\bar{\mathbf{t}}_i = [3\,2\,1\,1\,1]$. From (4), we can obtain $y_j(n-1, n-k, \bar{\mathbf{t}}_i)$ for helper node j. Note that helper node j for node i is the node whose link to node i has the j-th highest per-symbol transmission cost, which means helper node j for node i may not be node j. For helper nodes 1 and 2, we have $y_1(5, 3, \bar{\mathbf{t}}_6) = (3-1)3 - (2+1+1+1) > 0$ and $y_2(5, 3, \bar{\mathbf{t}}_6) = (3-2)2 - (1+1+1) < 0$. That means helper node 1 is a high-cost helper node, and helper nodes 2 to 5 are low-cost helper nodes. Theorem 3 indicates that we should avoid using high-cost helper nodes. Specifically, from (20), the $\bar{\boldsymbol{\beta}}$ leading to the optimal average repair cost has $\bar{\beta}_{1,i} = 0$ and $\bar{\beta}_{2,i} = \cdots = \bar{\beta}_{6,i} = \frac{1}{6}$. That means node i should be repaired by downloading $\frac{B}{6}$ symbols from helper nodes j for all $j = 2, \ldots, 5$, where B is file size.

2) Code construction and repair schemes: In Theorem 3, each helper node for node i is labeled as a high-cost, moderate-cost, and low-cost helper node. From (20), an HMSR code can have $\bar{\beta}_{j,i} = 0$ for all node i and corresponding moderate-cost helper node j. Furthermore, an HMSR code can have $\bar{\boldsymbol{\beta}}$ satisfying

$$\bar{\beta}_{j,i} = \begin{cases} \frac{1}{k} \cdot \frac{1}{|\mathcal{H}_L^{(i)}| - k + 1} & \forall i \in [n], j \in \mathcal{H}_L^{(i)}, \\ 0 & \forall i \in [n], j \in [n-1] \setminus \mathcal{H}_L^{(i)}. \end{cases} \tag{24}$$

Next, we consider an HMSR code with $\bar{\boldsymbol{\beta}}$ satisfying (24), i.e., an (n, k) MDS array code with $\bar{\boldsymbol{\beta}}$ satisfying (24). If $\bar{\boldsymbol{\beta}}$ satisfies (24), any node $i \in [n]$ can be repaired by downloading the same number of symbols from each low-cost helper node j, which is the kind of repair schemes considered in the homogeneous model [2]. Hence, some MDS array codes designed for the homogeneous model can be used to design HMSR codes.

Definition 1: Given an $(n, k, \mathbf{s}, \mathbf{t})$ DSS and a set $\mathcal{S} \subseteq [n-1]$, a repair scheme for an (n, k) MDS array code of data size B can be denoted as a repair scheme $\mathcal{R}_{\mathcal{S}}^i$ if the repair scheme repairs node i by downloading $\frac{B}{k} \frac{1}{|\mathcal{S}| - k + 1}$ symbols from helper node j for all $j \in \mathcal{S}$. For an $(n, k, \mathbf{s}, \mathbf{t})$ DSS, an (n, k) MDS array code is said to have the optimal repair property if it is equipped with a repair scheme $\mathcal{R}_{\mathcal{S}}^i$ for all $i \in [n]$ and $\mathcal{S} \subseteq [n-1]$ with $k \leq |\mathcal{S}| \leq n-1$. Note that given an (n, k) MDS array code with the optimal repair property, one can choose many repair schemes for a failed node i, such as $\mathcal{R}_{[k]}^i$ and $\mathcal{R}_{[k+1]}^i$.

From this definition and (24), an (n, k) MDS array code that uses a repair scheme $\mathcal{R}_{\mathcal{H}_L^{(i)}}^i$ for node i for all $i \in [n]$ is an HMSR code. Furthermore, since $|\mathcal{H}_L^{(i)}| \geq k$, an (n, k) MDS array code with the optimal repair property has a repair scheme $\mathcal{R}_{\mathcal{H}_L^{(i)}}^i$ for all $i \in [n]$, which leads to the following theorem.

Theorem 4: For an $(n, k, \mathbf{s}, \mathbf{t})$ DSS with only low-cost nodes, an (n, k) MDS array code with the optimal repair property, which has been explicitly constructed in [6] for all n and k, can be an HMSR code regarding the average repair cost if it chooses to use the repair scheme $\mathcal{R}_{\mathcal{H}_L^{(i)}}^i$ to repair node i for all $i \in [n]$.

Example 1: An $(n = 6, k = 3)$ MDS array code with the optimal repair property can be an HMSR code regarding the average cost for a $(6, 3, \mathbf{s}, \mathbf{t})$ DSS with only low-cost nodes. For any node i, this code can choose many repair schemes, such as $\mathcal{R}_{[3]}^i$, $\mathcal{R}_{[4]}^i$, and $\mathcal{R}_{\{2,\ldots,5\}}^i$. But this code cannot achieve the optimal average repair cost unless it chooses the right one. Given $\bar{\mathbf{t}}_i = [3\,2\,1\,1\,1]$ as an example, from (17), the low-cost helper nodes for node i are helper nodes 2 to 5. Thus, the code must use the corresponding repair scheme $\mathcal{R}_{\{2,\ldots,5\}}^i$; otherwise, it cannot achieve the optimal average repair cost.

Remark 2: For an (n, k) DSS with only low-cost nodes and time-varying $\mathbf{t}$, we can still encode the system by an (n, k) MDS array code with the optimal repair property. If node i fails, given the current $\mathbf{t}$, since this code has a repair scheme $\mathcal{R}_{\mathcal{S}}^i$ for all $i \in [n]$ and $\mathcal{S} \subseteq [n-1]$ with $k \leq |\mathcal{S}| \leq n-1$, the system can calculate $\mathcal{H}_L^{(i)}$ for the current $\mathbf{t}$ and call the repair scheme $\mathcal{R}^i_{\mathcal{H}_L^{(i)}}$. Thus, this code can achieve the optimal average repair cost for the time-varying $\mathbf{t}$.

Remark 3: In this part, we focus on constructing an HMSR code with $\bar{\boldsymbol{\beta}}$ satisfying (24). In fact, for any $\bar{\boldsymbol{\beta}}$ satisfying (20), the corresponding HMSR code can be constructed by leveraging the MDS array code with the optimal repair property. The code construction and corresponding repair schemes are introduced in the next subsection. Please refer to Remark 4 for details.

B. HMSR codes regarding the worst-case repair cost

1) Characterization of $\bar{\boldsymbol{\beta}}$: Similar to Problem 2.A, the optimal worst-case repair cost of (n,k) MDS array codes, i.e., the repair cost for an HMSR code regarding the worst-case repair cost, is the optimal value of the following problem.

Problem 3.A: Given n, k, and $\bar{\mathbf{t}}$,

$$\begin{aligned}&\min_{\bar{\boldsymbol{\beta}}\in\mathbb{R}^{n(n-1)}} \quad \max_{i\in[n]}\max_{j\in[n-1]} \bar{t}_{j,i}\bar{\beta}_{j,i}\\ &\text{s.t.} \quad \bar{\beta}_{j,i}\geq 0 \quad \forall i\in[n], j\in[n-1],\\ &\quad \sum_{j\in\mathcal{S}}\bar{\beta}_{j,i}\geq\frac{1}{k} \quad \forall i\in[n], \forall\mathcal{S}\subseteq[n-1] \text{ with } |\mathcal{S}|=n-k.\end{aligned}$$

Similar to Problem 2.A, Problem 3.A can also be decomposed into n subproblems, which minimize the worst-case repair cost for all node $i\in[n]$, respectively.

Problem 3.B: Given n, k, a node index i, and $\bar{\mathbf{t}}_i$,

$$\begin{aligned}&\min_{\bar{\boldsymbol{\beta}}_i\in\mathbb{R}^{n-1}} \quad \max_{j\in[n]\setminus\{i\}} \bar{t}_{j,i}\bar{\beta}_{j,i}\\ &\text{s.t.} \quad \bar{\beta}_{j,i}\geq 0 \quad \forall j\in[n]\setminus\{i\}, && (25)\\ &\quad \sum_{j\in\mathcal{S}}\bar{\beta}_{j,i}\geq\frac{1}{k} \quad \forall\mathcal{S}\subseteq[n-1] \text{ with } |\mathcal{S}|=n-k. && (26)\end{aligned}$$

If an (n,k) MDS array code has the optimal worst-case repair cost for all node $i\in[n]$, this (n,k) MDS array code clearly has the optimal worst-case repair cost for the system. Our results show that such an (n,k) MDS array code exists. By solving Problem 3.B associated with all nodes, $\bar{\boldsymbol{\beta}}$ for such an (n,k) MDS array code is characterized in the following theorem.

Theorem 5: Considering an $(n,k,\mathbf{s},\mathbf{t})$ DSS with only low-cost nodes, an HMSR code regarding the worst-case repair cost can be an (n,k) MDS array code with optimal worst-case repair cost for all node $i\in[n]$, whose $\bar{\boldsymbol{\beta}}$ can be fully characterized by

$$\begin{aligned}&\bar{\beta}_{j,i}=G_i/\bar{t}_{j,i} \quad \forall i\in[n], j\in[n-k]\\ &\bar{\beta}_{n-k,i}\leq\bar{\beta}_{j,i}\leq G_i/\bar{t}_{j,i}\,\forall i\in[n], j\in[n-1]\setminus[n-k],\end{aligned} \tag{27}$$

where G_i, which is defined by

$$G_i\triangleq 1/(\frac{k}{\bar{t}_{1,i}}+\cdots+\frac{k}{\bar{t}_{n-k,i}}),$$

is the optimal worst-case repair cost for node $i\in[n]$.

Note that $\bar{t}_{1,i}\geq\cdots\geq\bar{t}_{n-1,i}$ for all $i\in[n]$. Theorem 5 indicates that to achieve the optimal worst-case repair cost for node i, we must let (a) the $n-k$ helper nodes with the $n-k$ largest per-symbol transmission costs ($\bar{t}_{j,i}$) have the same transmission cost (the same $\bar{t}_{j,i}\bar{\beta}_{j,i}$), which is the optimal worst-case repair cost for node i; (b) $\bar{\beta}_{j,i}$ for the $k-1$ helper nodes with the $k-1$ smallest per-symbol transmission cost are at least $\bar{\beta}_{n-k,i}$. Consider $(n=6,k=3)$ MDS array codes in a $(6,3,\mathbf{s},\mathbf{t})$ DSS with $\bar{\mathbf{t}}_i=[3\,2\,2\,2\,2]$ for some $i\in[n]$. To illustrate how we solve Problem 3.B and obtain Theorem 5, we will choose some $\bar{\boldsymbol{\beta}}_i$ and show how to modify this $\bar{\boldsymbol{\beta}}_i$ to reduce the repair cost and eventually get a $\bar{\boldsymbol{\beta}}_i$ satisfying (27). To achieve the optimal worst-case repair cost for node i, the system should download fewer symbols from a helper node with a higher cost. Thus, we first choose $\bar{\boldsymbol{\beta}}_i=[1\,2\,3\,4\,5]$ and show how to modify it to reduce repair cost.

1) For $\bar{\boldsymbol{\beta}}_i=[1\,2\,3\,4\,5]$, we have $\min_{\mathcal{S}\subseteq[n-1]\text{ with }|\mathcal{S}|=n-k}\sum_{j\in\mathcal{S}}\bar{\beta}_{j,i}=\sum_{j\in[3]}\bar{\beta}_{j,i}=6$. For a feasible $\bar{\boldsymbol{\beta}}_i$ (cf. (26)), we only need $\min_{\mathcal{S}\subseteq[n-1]\text{ with }|\mathcal{S}|=n-k}\sum_{j\in\mathcal{S}}\bar{\beta}_{j,i}$ to be $1/k$. Thus, we can multiply the considered $\bar{\boldsymbol{\beta}}_i$ by $\frac{1}{6k}$ and obtain $[\frac{1}{18}\,\frac{2}{18}\,\frac{3}{18}\,\frac{4}{18}\,\frac{5}{18}]$, which leads to less repair cost. Futhermore, we can let $\bar{\beta}_{4,i}$ and $\bar{\beta}_{5,i}$ be the same as $\bar{\beta}_{3,i}$. Thus, $\bar{\boldsymbol{\beta}}_i$ becomes $[\frac{1}{18}\,\frac{2}{18}\,\frac{3}{18}\,\frac{3}{18}\,\frac{3}{18}]$, which satisfies (26) and leads to less repair cost.
2) Now we have a $\bar{\boldsymbol{\beta}}_i$ with $\bar{\beta}_{n-k,i}=\cdots=\bar{\beta}_{n-1,i}$. Since $\bar{t}_{1,i}\geq\cdots\geq\bar{t}_{n-1,i}$, helper node $n-k$ has the largest repair cost among the last k helper nodes. Thus, to reduce the worst-case repair cost, we only need to consider the transmission cost for the first $n-k$ helper nodes. In the considered DSS with $\bar{\mathbf{t}}_i=[3\,2\,2\,2\,2]$, given $\bar{\boldsymbol{\beta}}_i=[\frac{1}{18}\,\frac{2}{18}\,\frac{3}{18}\,\frac{3}{18}\,\frac{3}{18}]$, the first three helper nodes of the failed node i correspond to different transmission costs (different $\bar{t}_{j,i}\bar{\beta}_{j,i}$). If we can reduce the high transmission cost and increase the low transmission cost, the worst-case repair cost will be reduced. For example, for $\bar{\boldsymbol{\beta}}_i=[\frac{1}{18}\,\frac{2}{18}\,\frac{3}{18}\,\frac{3}{18}\,\frac{3}{18}]$, if we increase $\bar{\beta}_{2,i}$ by $\frac{1}{36}$ and reduce $\bar{\beta}_{j,i}$ by $\frac{1}{36}$ for all $j=3,\ldots,5$, $\bar{\boldsymbol{\beta}}_i$ becomes $[\frac{1}{18}\,\frac{5}{36}\,\frac{5}{36}\,\frac{5}{36}\,\frac{5}{36}]$ leading to less repair cost. If we continue to balance the transmission cost for the first three helper nodes while guaranteeing $\sum_{j\in[3]}\bar{\beta}_{j,i}=\frac{1}{k}$ and $\bar{\beta}_{3,i}=\cdots=\bar{\beta}_{5,i}$, the worst-case repair cost will continue to decrease. Eventually, the first three helper nodes will have the same transmission cost, $\bar{\boldsymbol{\beta}}_i$ will become $[\frac{1}{12}\,\frac{1}{8}\,\frac{1}{8}\,\frac{1}{8}\,\frac{1}{8}]$, which satisfies (27).

The formal proof of Theorem 5 will be given in Appendix B.

2) Code construction and repair schemes: If we can construct an (n,k) MDS array code with $\bar{\boldsymbol{\beta}}$ satisfying

$$\bar{\beta}_{j,i}=\begin{cases}G_i/\bar{t}_{j,i} & \forall i\in[n], j\in[n-k],\\ \bar{\beta}_{n-k,i} & \forall i\in[n], j\in[n-1]\setminus[n-k],\end{cases} \tag{28}$$

we can construct an (n,k) MDS array code with any $\bar{\boldsymbol{\beta}}$ satisfying (27). Next, we aim to construct an (n,k) MDS array code with $\bar{\boldsymbol{\beta}}$ satisfying (28), which is an HMSR code regarding the worst-case repair cost for a DSS with only low-cost nodes. To have $\bar{\boldsymbol{\beta}}$ satisfying (28), repairing a node needs to download different numbers of symbols from different helper nodes, which is no longer possible for the conventional MSR codes. However, by *stacking* an (n,k) MDS array code with the optimal repair property (cf. Definition 1) for several times, we

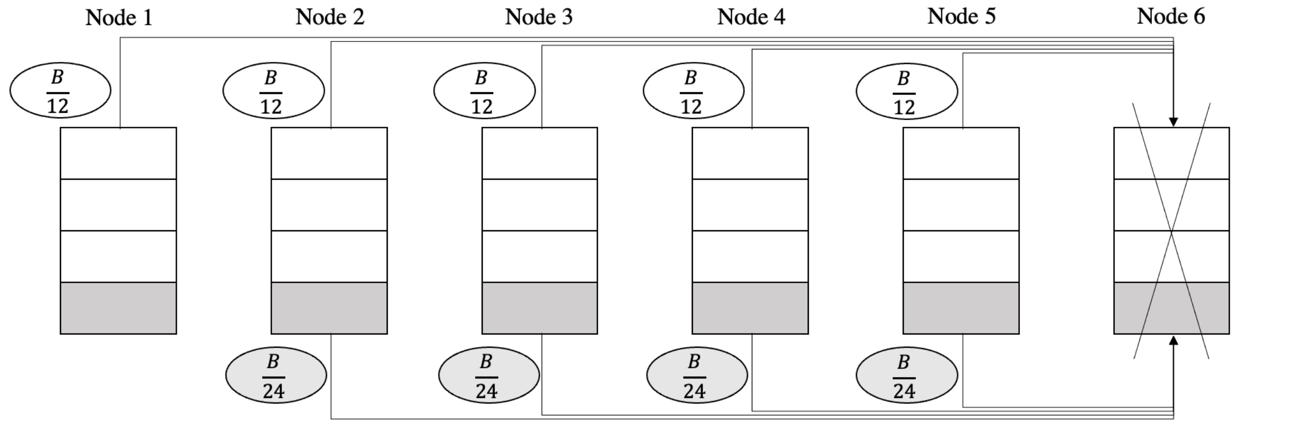

Fig. 1. This figure illustrates a repair scheme of the failed node 6 for $\mathcal{C}^{(4)}$. The 4 layers of a codeword are divided into two parts, colored white and gray, respectively. The repair schemes $\mathcal{R}^{6}_{\mathcal{S}_{1,6}}$ and $\mathcal{R}^{6}_{\mathcal{S}_{2,6}}$ (cf. Definition 1) are applied on the layers of the white and gray parts, respectively. Thus, the failed white part is repaired by downloading $\frac{B}{12}$ symbols from helper nodes $1,\ldots,5$, where B is the file size, and the failed gray part is repaired by downloading $\frac{B}{24}$ symbols from helper nodes $2,\ldots,5$.

can obtain an (n,k) MDS array code with $\bar{\boldsymbol{\beta}}$ satisfying (28). Specifically, given any positive integer W and an (n,k) MDS array code $\mathcal{C}_{base}$ with the optimal repair property, by stacking W code instances of $\mathcal{C}_{base}$, we obtain the (n,k) MDS array code $\mathcal{C}^{(W)}$ defined as

$$\mathcal{C}^{(W)} \triangleq \left\{ \begin{bmatrix} \mathbf{c}_{1,1} & \ldots & \mathbf{c}_{1,n} \\ \vdots & \ddots & \vdots \\ \mathbf{c}_{W,1} & \ldots & \mathbf{c}_{W,n} \end{bmatrix} \middle| [\mathbf{c}_{i,j}]_{j\in[n]} \in \mathcal{C}_{base}, \forall i \in [W] \right\}. \tag{29}$$

Note that a codeword of $\mathcal{C}^{(W)}$ has W layers which are W codewords of $\mathcal{C}_{base}$. Since $\mathcal{C}_{base}$ has the optimal repair property, many different repair schemes leveraging different helper nodes exist for a single failed node. For a failed node of $\mathcal{C}^{(W)}$, we can apply different repair schemes to different layers to accomplish download different numbers of symbols from different helper nodes. For a clearer explanation, we first present an example.

Example 2: Consider $(n=6,k=3)$ MDS array codes in a $(6,3,\mathbf{s},\mathbf{t})$ DSS with $\bar{t}_{1,6}=3, \bar{t}_{2,6}=\cdots=\bar{t}_{5,6}=2$, $\bar{t}_{1,1}=\bar{t}_{2,1}=5$, and $\bar{t}_{3,1}=\bar{t}_{4,1}=\bar{t}_{5,1}=2$. With these parameters, $\bar{\boldsymbol{\beta}}$ given in (28) has $\bar{\beta}_{1,6}=\frac{1}{12}$, $\bar{\beta}_{2,6}=\cdots=\bar{\beta}_{5,6}=\frac{1}{8}$, $\bar{\beta}_{1,1}=\bar{\beta}_{2,1}=\frac{2}{27}$, and $\bar{\beta}_{3,1}=\bar{\beta}_{4,1}=\bar{\beta}_{5,1}=\frac{5}{27}$. Thus, to achieve the optimal worst-case repair cost of node 6, we want to construct a $(6,3)$ MDS array code with data size B that can repair node 6 by downloading $\frac{B}{12}$ symbols from helper node 1 and $\frac{B}{8}$ symbols from each helper node $j=2,\ldots,5$. First, choose a $(6,3)$ MDS array code $\mathcal{C}_{base}$ with data size B' and the optimal repair property. The codeword of $\mathcal{C}_{base}$ can be written as an array $[\mathbf{c}_1 \ldots \mathbf{c}_6]$ where $\mathbf{c}_j$ is a column vector of length B' for all $j\in[6]$. From the optimal-repair property, $\mathbf{c}_6$ can be repaired by downloading $\frac{B'}{9}$ symbols from 5 helper nodes, or $\frac{B'}{6}$ symbols from any 4 helper nodes, or $\frac{B'}{3}$ symbols from any 3 helper nodes. By stacking 4 instances of $\mathcal{C}_{base}$, we can obtain a $(6,3)$ MDS array code $\mathcal{C}^{(4)}$ given by

$$\mathcal{C}^{(4)} = \left\{ \begin{bmatrix} \mathbf{c}_{1,1} & \ldots & \mathbf{c}_{1,6} \\ \vdots & \ddots & \vdots \\ \mathbf{c}_{4,1} & \ldots & \mathbf{c}_{4,6} \end{bmatrix} \middle| [\mathbf{c}_{i,j}]_{j\in[6]} \in \mathcal{C}_{base}, \forall i \in [4] \right\}. \tag{30}$$

Clearly, the data size associated with $\mathcal{C}^{(4)}$ is $B=4B'$. Note that the codeword of $\mathcal{C}^{(4)}$ has four layers which are four codewords of $\mathcal{C}_{base}$. We divide the four layers into two groups. The first group contains the first three layers, and the second group contains the fourth layer. We then apply different repair schemes of $\mathcal{C}_{base}$ to different layer groups to constitute a repair scheme of $\mathcal{C}^{(4)}$ (cf. Fig. 1). For the layers in the first layer group, using the repair scheme $\mathcal{R}^{6}_{\{1,\ldots,5\}}$, we repair $\{\mathbf{c}_{t,6}\}_{t\in[3]}$ by downloading $3\cdot\frac{B'}{9}=\frac{B'}{3}$ symbols from helper node j for all $j\in[5]$. For the layer in the second group, using the repair scheme $\mathcal{R}^{6}_{\{2,\ldots,5\}}$, we repair $\mathbf{c}_{4,6}$ by downloading $\frac{B'}{6}$ symbols from helper node j for all $j=2,\ldots,5$. Overall, this repair scheme downloads $\frac{B'}{3}=\frac{4B'}{12}=\frac{B}{12}$ symbols from helper node 1 and $\frac{3B'}{9}+\frac{B'}{6}=\frac{B'}{2}=\frac{4B'}{8}=\frac{B}{8}$ from each helper node j with $j\in[5]\setminus\{1\}$. Thus, $\mathcal{C}^{(4)}$ with this repair scheme for node 6 has $\bar{\boldsymbol{\beta}}_6$ satisfying (28). Furthermore, $\mathcal{C}^{(W)}$ with W divisible by 4 can also have $\bar{\boldsymbol{\beta}}_6$ satisfying (28) by similarly dividing W layers into two groups consisting of $\frac{3W}{4}$ and $\frac{W}{4}$ layers, respectively, and applying $\mathcal{R}^{6}_{\{1,\ldots,5\}}$ and $\mathcal{R}^{6}_{\{2,\ldots,5\}}$ on the layers of the two groups accordingly.

Next, we consider the repair of node 1. From the given $\mathbf{t}$, $\bar{\boldsymbol{\beta}}$ given in (28) has $\bar{\beta}_{1,1}=\bar{\beta}_{2,1}=\frac{2}{27}$, and $\bar{\beta}_{3,1}=\bar{\beta}_{4,1}=\bar{\beta}_{5,1}=\frac{5}{27}$. Thus, we want to construct a $(6,3)$ MDS array code with data size B that can repair node 1 by downloading $\frac{2B}{27}$ symbols from each helper node $j=1,2$ and $\frac{5B}{27}$ symbols from each helper node $j=3,4,5$. $\mathcal{C}^{(3)}$ with data size $B=3B'$ can have such a repair scheme for node 1. In the first two layers of $\mathcal{C}^{(3)}$, using the repair scheme $\mathcal{R}^{1}_{\{1,\ldots,5\}}$, we repair node 1 by downloading $2\cdot\frac{B'}{9}$ symbols from all helper nodes. In the third layer, using the repair scheme $\mathcal{R}^{1}_{\{3,4,5\}}$, we can repair node 1 by downloading $\frac{B'}{3}$ symbols from each helper node $j=3,4,5$. Overall, this repair scheme of node 1 downloads $\frac{2}{27}\cdot 3B'$ symbols from each helper node $j=1,2$ and $2\cdot\frac{B'}{9}+\frac{B'}{3}=\frac{5}{27}\cdot 3B'$ symbols from each helper node $j=3,4,5$. Thus, $\mathcal{C}^{(3)}$ with this repair scheme for node 1 has $\bar{\boldsymbol{\beta}}_1$ satisfying (28). We have mentioned that $\mathcal{C}^{(W)}$ with W divisible by 4 can have $\bar{\boldsymbol{\beta}}_6$ satisfying (28). Likewise, $\mathcal{C}^{(W)}$ with W

TABLE II
NOTATION RELATED TO THE REPAIR SCHEME OF NODE i FOR $\mathcal{C}^{(W)}$

Symbol	Definition
m_i	The number of groups that the W layers are divided into
$E_{t,i}$	The number of layers in the t-th layer group normalized by the number of all layers
$\mathcal{S}_{t,i}$	The index set of the helper nodes involved in the repair of the layers in the t-th layer group
$\mathcal{R}^i_{\mathcal{S}_{t,i}}$	The repair scheme applied on the layers of the t-th layer group

divisible by 3 can have $\bar{\boldsymbol{\beta}}_1$ satisfying (28). Thus, $\mathcal{C}^{(12)}$ can achieve the optimal worst-case repair costs for both nodes 1 and 6 since 12 is divisible by both 3 and 4. This implies that by stacking the base code $\mathcal{C}_{base}$, we may obtain an MDS code with optimal worst-case repair costs for multiple nodes and even all nodes.

This example implies that by properly choosing W and designing repair schemes, $\mathcal{C}^{(W)}$ can be an MDS array code with optimal worst-case repair cost for all nodes, which is an HMSR code regarding the worst-case repair cost for a DSS with only low-cost nodes. The repair scheme of $\mathcal{C}^{(W)}$ can be formally described as follows. If node i fails, the system separates the W layers into m_i groups such that the t-th group consists of $E_{t,i}W$ layers. Then, a repair scheme $\mathcal{R}^i_{\mathcal{S}_{t,i}}$ (cf. Definition 1) is applied on each layer in the t-th group. Thus, the repair scheme of $\mathcal{C}^{(W)}$ can be characterized by W, $\{m_i\}_{i\in[n]}$, $\{\mathcal{S}_{t,i}\}_{i\in[n],t\in[m_i]}$, and $\{E_{t,i}\}_{i\in[n],t\in[m_i]}$. These new symbols and definitions related to the repair scheme are summarized in Table II for clarity. The question that needs to be answered is how to choose W, $\{m_i\}_{i\in[n]}$, $\{\mathcal{S}_{t,i}\}_{i\in[n],t\in[m_i]}$, and $\{E_{t,i}\}_{i\in[n],t\in[m_i]}$ to let $\mathcal{C}^{(W)}$ have $\bar{\boldsymbol{\beta}}$ satisfying (28). Before formally answer that by Theorem 6, we revisit the case considered in Example 2 to illustrate how to choose these parameters.

Example 2 (continued): For a $(6, 3, \mathbf{s}, \mathbf{t})$ DSS with $\bar{t}_{1,6} = 3$ and $\bar{t}_{2,6} = \cdots = \bar{t}_{5,6} = 2$, it has been shown in Example 2 that $\mathcal{C}^{(W)}$ can have $[\bar{\beta}_{j,6}]_{j\in[5]}$ satisfying (28) by selecting specific values for W, m_6, $\mathcal{S}_{1,6}$, $\mathcal{S}_{2,6}$, $E_{1,6}$, and $E_{2,6}$. Specifically, we can choose $W = 4$, $m_6 = 2$, $\mathcal{S}_{1,6} = \{1, \ldots, 5\}$, $\mathcal{S}_{2,6} = \{2, \ldots, 5\}$, $E_{1,6} = \frac{3}{4}$, and $E_{2,6} = \frac{1}{4}$. Next, we explain how to determine these parameters. Let $\bar{\beta}^*_{j,6}$ for all $j \in [5]$ denote the $\bar{\beta}_{j,6}$ satisfying (28). It has been shown in Example 2 that $\bar{\beta}^*_{1,6} = \frac{1}{12}$, $\bar{\beta}^*_{2,6} = \cdots = \bar{\beta}^*_{5,6} = \frac{1}{8}$. Note that $\bar{\beta}^*_{1,6} < \bar{\beta}^*_{2,6} = \cdots = \bar{\beta}^*_{5,6}$. Let us first design a repair scheme with $\bar{\beta}_{1,6} < \bar{\beta}_{2,6} = \cdots = \bar{\beta}_{5,6}$. First, divide all layers into 2 groups: one containing $E_{1,6}W$ layers and the other containing $E_{2,6}W$ layers. Then, apply $\mathcal{R}^6_{\{1,\ldots,5\}}$ and $\mathcal{R}^6_{\{2,\ldots,5\}}$ onto the layers of groups 1 and 2, respectively (cf. Fig. 2). This scheme leads to $\bar{\beta}_{1,6} < \bar{\beta}_{2,6} = \cdots = \bar{\beta}_{5,6}$. This implies that (a) m_6 should be 2, which is the number of different positive values in $[\bar{\beta}^*_{j,6}]_{j\in[5]}$; (b) $\mathcal{S}_{1,6}$ should be $\{j \in [n-1] | \bar{\beta}^*_{j,6} \geq \bar{\beta}^*_{(1),6}\}$, where $\bar{\beta}^*_{(1),6}$ be the smallest positive value in $[\bar{\beta}^*_{j,6}]_{j\in[5]}$; (c) $\mathcal{S}_{2,6}$ should be $\{j \in [n-1] | \bar{\beta}^*_{j,6} \geq \bar{\beta}^*_{(2),6}\}$, where $\bar{\beta}^*_{(2),6}$ be the second smallest positive value in $[\bar{\beta}^*_{j,6}]_{j\in[5]}$. Next, our objective is to achieve $\bar{\beta}_{1,6} = \bar{\beta}^*_{(1),6} = \frac{1}{12}$ and $\bar{\beta}_{2,6} = \cdots = \bar{\beta}_{5,6} = \bar{\beta}^*_{(2),6} = \frac{1}{8}$ by designing the dividing ratios $E_{1,6}$ and $E_{2,6}$. One can tell from Fig. 2 that the repair scheme downloads $\frac{E_{1,6}B}{k(|\mathcal{S}_{1,6}|-k+1)} = \frac{E_{1,6}B}{9}$ symbols from helper node 1, where B is the file size. If this repair scheme achieves $\bar{\beta}_{1,6} = \bar{\beta}^*_{1,6} = \bar{\beta}^*_{(1),6}$, $E_{1,6}$ should satisfy $\frac{E_{1,6}B}{k(|\mathcal{S}_{1,6}|-k+1)} = \bar{\beta}^*_{(1),6}B$, which indicates that $E_{1,6}$ should be $\bar{\beta}^*_{(1),6}k(|\mathcal{S}_{1,6}| - k + 1)$. Next, we want to guarantee $\bar{\beta}_{2,6} = \bar{\beta}^*_{2,6} = \bar{\beta}^*_{(2),6}$. Since the number of symbols downloaded from helper node 2 is $\frac{E_{1,6}B}{k(|\mathcal{S}_{1,6}|-k+1)} + \frac{E_{2,6}B}{k(|\mathcal{S}_{2,6}|-k+1)}$ (cf. Fig. 2), $E_{1,6}$ and $E_{2,6}$ should satisfy $\frac{E_{1,6}}{k(|\mathcal{S}_{1,6}|-k+1)} + \frac{E_{2,6}}{k(|\mathcal{S}_{2,6}|-k+1)} = \bar{\beta}^*_{(2),6}$. Since we have already let $\frac{E_{1,6}}{k(|\mathcal{S}_{1,6}|-k+1)} = \bar{\beta}^*_{(1),6}$, $E_{2,6}$ only need to satisfy $\frac{E_{2,6}}{k(|\mathcal{S}_{2,6}|-k+1)} = \bar{\beta}^*_{(2),6} - \bar{\beta}^*_{(1),6}$, which indicates that $E_{2,6}$ should be $k(\bar{\beta}^*_{(2),6} - \bar{\beta}^*_{(1),6})(|\mathcal{S}_{2,6}| - k + 1)$. One last thing we need to verify is whether $E_{1,6} + E_{2,6} = 1$. Since $E_{1,6} = \bar{\beta}^*_{(1),6}k(|\mathcal{S}_{1,6}| - k + 1) = \frac{3(5-3+1)}{12} = \frac{3}{4}$ and $E_{2,6} = k(\bar{\beta}^*_{(2),6} - \bar{\beta}^*_{(1),6})(|\mathcal{S}_{2,6}| - k + 1) = 3(\frac{1}{8} - \frac{1}{12})(4-3+1) = \frac{1}{4}$, we have $E_{1,6} + E_{2,6} = 1$. It is worth mentioning that the equation $\sum_{t=1}^{2} E_{t,6} = 1$ holds because of some structure (cf. (31) and (32)) of the $[\bar{\beta}_{j,i}]_{j\in[n-1]}$ satisfying (28), which will be discussed in the proof of Theorem 6. As for the number of layers W, we only need to guarantee that $E_{1,6}W$ and $E_{2,6}W$ are integers, which indicates that W should be a multiple of 4.

Example 2 (continued) illustrates how to determine $\{m_i\}_{i\in[n]}$, $\{\mathcal{S}_{t,i}\}_{i\in[n],t\in[m_i]}$, $\{E_{t,i}\}_{i\in[n],t\in[m_i]}$, and W to let $\mathcal{C}^{(W)}$ have $\bar{\boldsymbol{\beta}}$ satisfying (28). Formally, we have the following theorem.

Theorem 6: Given an $(n, k, \mathbf{s}, \mathbf{t})$ DSS with only low-cost nodes, let

$$\bar{\beta}^*_{j,i} \triangleq \begin{cases} G_i/\bar{t}_{j,i} & \forall i \in [n], j \in [n-k], \\ \bar{\beta}_{n-k,i} & \forall i \in [n], j \in [n-1] \setminus [n-k], \end{cases}$$

$\bar{\beta}^*_{(0),i} \triangleq 0$, and $\bar{\beta}^*_{(t),i}$ be the t-th smallest positive value in $[\bar{\beta}^*_{j,i}]_{j\in[n-1]}$. For this DSS, $\mathcal{C}^{(W)}$ has $\bar{\boldsymbol{\beta}}$ satisfying (28) and hence is an HMSR code regarding the worst-case repair cost if m_i is the number of different positive values in $[\bar{\beta}^*_{j,i}]_{j\in[n-1]}$ for all $i \in [n]$,

$$\begin{aligned} &\mathcal{S}_{t,i} = \{j \in [n-1] | \bar{\beta}^*_{j,i} \geq \bar{\beta}^*_{(t),i}\} \quad \forall i \in [n], t \in [m_i], \\ &E_{t,i} = k(\bar{\beta}^*_{(t),i} - \bar{\beta}^*_{(t-1),i})(|\mathcal{S}_{t,i}| - k + 1) \, \forall i \in [n], t \in [m_i], \\ &E_{t,i}W \in \mathbb{Z} \quad \forall i \in [n], t \in [m_i]. \end{aligned}$$

Proof: First, we show $\sum_{t\in[m_i]} E_{t,i} = 1$. From the definition of $\bar{\beta}^*_{j,i}$, we have

$$\bar{\beta}^*_{1,i} \leq \cdots \leq \bar{\beta}^*_{n-k,i} = \cdots = \bar{\beta}^*_{n-1,i} \quad \forall i \in [n], \tag{31}$$

$$\sum_{j=1}^{n-k} \bar{\beta}^*_{j,i} = \frac{1}{k} \quad \forall i \in [n], \tag{32}$$

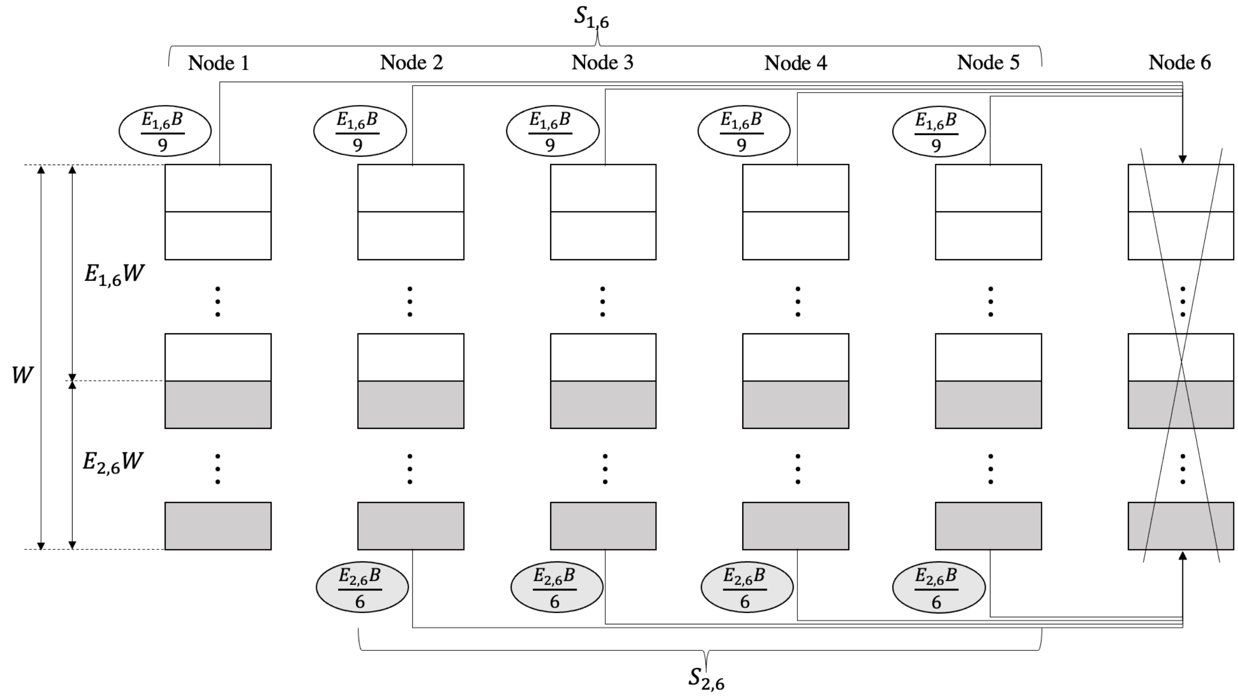

Fig. 2. This figure illustrates the code $\mathcal{C}^{(W)}$ and corresponding repair scheme of node 6 discussed in Example 2 (continued). The W layers of a codeword are divided into $m_6 = 2$ parts colored by white and gray, respectively. The white and gray parts have $E_{1,6}W$ and $E_{2,6}W$ layers, respectively. Then, the repair schemes $\mathcal{R}^6_{\mathcal{S}_{1,6}}$ and $\mathcal{R}^6_{\mathcal{S}_{2,6}}$ are applied on the layers of the white and gray parts, respectively, where $\mathcal{S}_{1,6} = \{1, \ldots, 5\}$ and $\mathcal{S}_{2,6} = \{2, \ldots, 5\}$. Thus, from the definition of the repair schemes $\mathcal{R}^6_{\mathcal{S}_{1,6}}$ and $\mathcal{R}^6_{\mathcal{S}_{2,6}}$ (cf. Definition 1), the failed white part is repaired by downloading $\frac{E_{1,6}B}{9}$ symbols from helper nodes $1, \ldots, 5$, where B is the file size, and the failed gray part is repaired by downloading $\frac{E_{2,6}B}{6}$ symbols from helper nodes $2, \ldots, 5$.

which indicates $0 < \bar{\beta}^*_{(1),i} < \cdots < \bar{\beta}^*_{(m_i),i} = \bar{\beta}^*_{n-k,i}$ and $m_i \le n-k$. From the definition of $\mathcal{S}_{t,i}$, we have

$$\sum_{j\in[n-1]} \bar{\beta}^*_{j,i} = |\mathcal{S}_{m_i,i}|\bar{\beta}^*_{(m_i),i} + \sum_{t=1}^{m_i-1} (|\mathcal{S}_{t,i}| - |\mathcal{S}_{t+1,i}|)\bar{\beta}^*_{(t),i},$$

which, together with $\bar{\beta}^*_{(m_i),i} = \bar{\beta}^*_{n-k,i} = \cdots = \bar{\beta}^*_{n-1,i}$, implies

$$\sum_{j\in[n-k]} \bar{\beta}^*_{j,i} = (|\mathcal{S}_{m_i,i}| - k + 1)\bar{\beta}^*_{(m_i),i} + \sum_{t=1}^{m_i-1} (|\mathcal{S}_{t,i}| - |\mathcal{S}_{t+1,i}|)\bar{\beta}^*_{(t),i}. \tag{33}$$

Furthermore, from (31) and (32), we have

$$\begin{aligned}\sum_{t\in[m_i]} E_{t,i} = &\sum_{t\in[m_i]} \left(k(\bar{\beta}^*_{(t),i} - \bar{\beta}^*_{(t-1),i})(|\mathcal{S}_{t,i}| - k + 1)\right) \\ =&k\Big((|\mathcal{S}_{m_i,i}| - k + 1)\bar{\beta}^*_{(m_i),i} \\ &+ \sum_{t=1}^{m_i-1} (|\mathcal{S}_{t,i}| - |\mathcal{S}_{t+1,i}|)\bar{\beta}^*_{(t),i}\Big) \\ =&k\sum_{j\in[n-k]} \bar{\beta}^*_{j,i} = 1.\end{aligned} \tag{34}$$

Next, we prove that $\mathcal{C}^{(W)}$ considered in the theorem repairs node i by downloading $\bar{\beta}^*_{j,i}B$ symbols from each helper node $j \in [n-1]$, where B is the data size. Note that the system applies the repair scheme $\mathcal{R}^i_{\mathcal{S}_{t,i}}$ on the layers in the t-th group. That means, for the repair of each layer in the t-th group, the system downloads $\frac{B}{Wk(|\mathcal{S}_{t,i}|-k+1)}$ from helper node j for all $j \in \mathcal{S}_{t,i}$. Thus, as $E_{t,i} = k(\bar{\beta}^*_{(t),i} - \bar{\beta}^*_{(t-1),i})(|\mathcal{S}_{t,i}| - k + 1)$, the system downloads $\frac{BE_{t,i}W}{Wk(|\mathcal{S}_{t,i}|-k+1)} = (\bar{\beta}^*_{(t),i} - \bar{\beta}^*_{(t-1),i})B$ from helper node j for all $j \in \mathcal{S}_{t,i}$ for the repair of all $E_{t,i}W$ layers in t-th group. Next, consider a helper node j'. From the definition of $\mathcal{S}_{t,i}$, we obtain that $[n-1]$ can be partitioned into $m_i + 1$ subsets: $[n-1] \setminus \mathcal{S}_{1,i}, \mathcal{S}_{1,i} \setminus \mathcal{S}_{2,i}, \ldots, \mathcal{S}_{m_i,i} \setminus \mathcal{S}_{m_i+1,i}$, where $\mathcal{S}_{m_i+1,i} \triangleq \emptyset$. Thus, j' must be in one of these subsets. If $j' \in [n-1] \setminus \mathcal{S}_{1,i}$, from the definition of $\mathcal{S}_{t,i}$, we obtain that the repair scheme downloads $\beta^*_{j',i}B$ symbol from helper node j', where $\beta^*_{j',i} = 0$. If $j' \in \mathcal{S}_{t',i} \setminus \mathcal{S}_{t'+1,i}$ for some $t' \in [m_i]$, from the definition of $\mathcal{S}_{t,i}$, we have $j' \in \mathcal{S}_{t,i}$ for all $t \le t'$ and $j \notin \mathcal{S}_{t,i}$ for all $t > t'$. Thus, helper node j' is only involved in the repair of the layers in the t-th group for all $t \le t'$. That means the repair scheme downloads $\sum_{t=1}^{t'} (\bar{\beta}^*_{(t),i} - \bar{\beta}^*_{(t-1),i})B = (\bar{\beta}^*_{(t'),i} - \bar{\beta}^*_{(0),i})B = \bar{\beta}^*_{(t'),i}B$ from helper node j'. Since $j' \in \mathcal{S}_{t',i} \setminus \mathcal{S}_{t'+1,i}$, from the definition of $\mathcal{S}_{t,i}$ and $\bar{\beta}^*_{(t),i}$, we obtain $\bar{\beta}^*_{j',i} \ge \bar{\beta}^*_{(t'),i}$ and $\bar{\beta}^*_{j',i} < \bar{\beta}^*_{(t'+1),i}$, which leads to $\bar{\beta}^*_{(t'),i} = \bar{\beta}^*_{j',i}$. As such, the repair scheme downloads $\bar{\beta}^*_{j',i}B$ symbols from helper node j'. This completes the proof. ∎

Remark 4: Theorem 6 shows that $\mathcal{C}^{(W)}$ with properly designed $\{m_i\}_{i\in[n]}$, $\{\mathcal{S}_{t,i}\}_{i\in[n],t\in[m_i]}$, $\{E_{t,i}\}_{i\in[n],t\in[m_i]}$, and W can have $\bar{\boldsymbol{\beta}}$ satisfying (28). In fact, one can tell from the proof of Theorem 6 that $\mathcal{C}^{(W)}$ with properly chosen parameters can have $\bar{\boldsymbol{\beta}}$ satisfying (31) and (32), which are more general conditions than (28). In particular, given any $\bar{\boldsymbol{\beta}}$ satisfying (20), by reindexing the moderate-cost helper node for each failed node, this $\bar{\boldsymbol{\beta}}$ can be equivalently transformed into one satisfying (31) and (32). Thus, $\mathcal{C}^{(W)}$ with properly chosen parameters (cf. Theorem 6) can achieve this $\bar{\boldsymbol{\beta}}$, which leads to an HMSR code regarding the average repair cost for a DSS with only low-cost nodes. Specifically, the HMSR code give in Theorem 4 is a $\mathcal{C}^{(W)}$ with $W = 1$, $m_i = 1$, and $\mathcal{S}_{1,i} = \mathcal{H}_L^{(i)}$ for all $i \in [n]$.

Remark 5: The code array of the HMSR code $\mathcal{C}^{(W)}$ can be much larger than the homogeneous MSR array code. However, this larger code array will not jeopardize the decoding speed because a decoder does not need to process the whole array but only processes one layer at a time, which is a codeword of the homogeneous MSR code. In fact, if a real-world system is encoded by an MDS array code $\mathcal{C}_{base}$ with the optimal repair property, it can be seen as encoded by $\mathcal{C}^{(W)}$ with some very large W because the system must have tons of files and hence store many codewords of $\mathcal{C}_{base}$. Thus, for such a system, to achieve optimal worst-case repair cost of node i, we need to divide the stored codewords of $\mathcal{C}_{base}$ into m_i groups according to the ratios $\{E_{t,i}\}_{t\in[m_i]}$ and apply corresponding repair schemes $\mathcal{R}^i_{\mathcal{S}_{t,i}}$ for all $t \in [m_i]$. This means that to achieve optimal worst-case repair cost, a system does not need to implement $\mathcal{C}^{(W)}$; it just needs to implement $\mathcal{C}_{base}$ and the proposed repair scheme according to $\mathbf{t}$.

V. HMSR codes for a DSS with one moderate-cost node

HMSR codes for a DSS with no moderate-cost node have been obtained in the previous section. This section considers a DSS with only one moderate-cost node, which is the most important case among DSSs with at least one moderate-cost node (cf. Theorem 2). Remark 1 indicates that HMSR codes for a DSS are equivalent to the HMSR codes for the sub-DSS induced by the low-cost and moderate-cost nodes. Thus, WLOG, we consider an (n, k) DSS with one moderate-cost node and $n-1$ low-cost nodes. To characterize the HMSR codes for this DSS, we define *two-valued array codes* as follows.

Definition 2: An (n, k) irregular array code is called an (n, k) two-valued array code if the associated $\boldsymbol{\alpha}$ satisfies

$$\alpha_1 \le \alpha_2 = \alpha_3 = \cdots = \alpha_n, \tag{35}$$

$$\alpha_1 + (k-1)\alpha_2 = 1. \tag{36}$$

From Theorem 1 and Definition 2, an HMSR code for the considered $(n, k, \mathbf{s}, \mathbf{t})$ DSS must be a two-valued array code that only covers the moderate-cost and low-cost nodes. Furthermore, HMSR codes are the (n, k) two-valued array codes with the optimal repair cost. The average and worst-case repair costs are considered respectively in the following two subsections.

Note that we can also define t-valued array codes to characterize the HMSR codes for a DSS with $t-1$ moderate-cost nodes for $t \ge 3$. However, DSSs with multiple moderate-cost nodes are rare in comparison to the DSSs investigated in this paper, and therefore, they are beyond the scope of this paper.

A. HMSR codes regarding the average repair cost

In this subsection, we mainly focus on a DSS with one moderate-cost node, $n-1$ low-cost nodes, and $\mathbf{t}$ being an all-one vector. Since $\mathbf{t}$ is an all-one vector, the repair cost becomes the *repair bandwidth* defined for the homogeneous model [2]. For such a DSS, we first characterize $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ of an HMSR code regarding the average repair cost and discuss the construction for an exact-repair HMSR code with such $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

1) Characterization of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$: HMSR codes for the considered DSS are the (n, k) two-valued array codes with the optimal repair cost. To obtain the optimal repair cost, we need to study the min-cut condition for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. As we have mentioned before, [16] has already found the min-cut condition for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as presented in the next theorem.

Theorem 7 (cf. Theorems 1 and 2 in [16]): Let $\mathcal{F}$ be the set of k-vectors $\mathbf{f} = [f_1 \ldots f_k]$ whose components are chosen from $[n]$ and $f_i \neq f_j$ for $i \neq j$. In an $(n, k, \mathbf{s}, \mathbf{t})$ DSS, the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ of an (n, k) irregular array code satisfy

$$\min_{\mathbf{f}\in\mathcal{F}} \sum_{i=1}^{k} \min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}} \beta_{j,f_i}\} \ge 1. \tag{37}$$

Conversely, if allowing functional repair, there exists an (n, k) irregular array code with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as long as the condition (37) holds.

The min-cut condition in (37) is for general (n, k) irregular array codes. For (n, k) two-valued array codes, $\boldsymbol{\alpha}$ needs to satisfy (35) and (36). Combining (37) with (35) and (36), the min-cut condition for (n, k) two-valued array codes was shown in the following lemma.

Lemma 1: $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ of an (n, k) two-valued array code satisfy

$$\beta_{j,i} \ge 0 \quad \forall i, j \in [n] \text{ with } i \neq j, \tag{38}$$

$$\sum_{j\in\mathcal{S}} \beta_{j,1} \ge \alpha_1 \quad \forall \mathcal{S} \subseteq [n] \setminus \{1\} \text{ with } |\mathcal{S}| = n-k, \tag{39}$$

$$\sum_{j\in\mathcal{S}} \beta_{j,i} \ge \alpha_2 \quad \begin{array}{l}\forall i \in [n] \setminus \{1\}, \forall \mathcal{S} \subseteq [n] \setminus \{1, i\} \\ \text{with } |\mathcal{S}| = n-k,\end{array} \tag{40}$$

$$\beta_{1,i} + \sum_{j\in\mathcal{S}} \beta_{j,i} \ge \alpha_1 \quad \begin{array}{l}\forall i \in [n] \setminus \{1\}, \forall \mathcal{S} \subseteq [n] \setminus \{1, i\} \\ \text{with } |\mathcal{S}| = n-k-1.\end{array} \tag{41}$$

Conversely, there exists an (n, k) functional-repair irregular array code with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as long as the conditions (38), (39), (40), and (41) hold.

The detailed proof of Lemma 1 is given in Appendix C. Based on the simplified min-cut condition derived in Lemma 1, we formulate the following optimization problem whose optimal solutions fully characterize $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for the HMSR codes regarding the average repair cost for the considered DSS.

Problem 4: Given n, k, and $\mathbf{t}$,

$$\begin{aligned} \min_{\boldsymbol{\beta}\in\mathbb{R}^{n(n-1)}} \quad & C_R^{ave}(\boldsymbol{\beta}) \\ \text{s.t.} \quad & 0 \le \alpha_1 \le \alpha_2, \quad (k-1)\alpha_2 + \alpha_1 = 1 \\ & (38),\ (39),\ (40), \text{and } (41) \end{aligned} \tag{42}$$

Let us compare the average repair cost optimization problems for MDS array codes (cf. Problem 2.A) and two-valued array codes (cf. Problem 4). In Problem 2.A, node capacity α_i is not involved because it is a constant; while in Problem 4, α_i becomes variables. For MDS array codes, the optimal total repair cost for each node can be achieved simultaneously, and the optimal average repair cost can be achieved accordingly. While for (n,k) two-valued array codes, the optimal repair cost for each node cannot be achieved simultaneously because the repair cost for node i depends on α_i, which is not a constant for (n,k) two-valued array codes, and the node capacities cannot be minimized simultaneously. Specifically, for node 1, the optimal total repair cost is zero achieved by letting $\alpha_1 = 0$. If α_1 increases, α_2 and the repair cost for node 2 will decrease; however, the repair cost for node 1 will increase and is no longer optimal. Thus, it is unclear whether the average repair cost will increase or decrease. To see how the average repair cost changes as α_1 changes, we first derive the optimal average repair cost for an (n,k) two-valued array code with fixed α_1. If α_1 is fixed, Problem 4 can be decoupled into repair cost optimization problems for all nodes. By solving these sub-problems, we obtain the following theorem.

Theorem 8: For a DSS with $\mathbf{t}$ being an all one vector and an (n,k) two-valued array code with fixed α_1,

1) if $\alpha_1 \leq \frac{(n-k-1)\alpha_2}{n-k}$, the optimal average repair cost is $\frac{1}{n}(\frac{(n-1)\alpha_1}{n-k} + \frac{(n-1)(n-2)\alpha_2}{n-k}) = \frac{1}{n}((\frac{n-1}{n-k} - \frac{(n-1)(n-2)}{(n-k)(k-1)})\alpha_1 + \frac{(n-1)(n-2)}{(n-k)(k-1)})$. Furthermore, if $k \geq 3$, the optimal average repair cost can be achieved if and only if $\boldsymbol{\beta}$ satisfies

$$\begin{aligned}
&\beta_{j,1} = \frac{\alpha_1}{n-k} \quad \forall j \in [n]\setminus\{1\},\\
&\beta_{1,i} = 0 \quad \forall i \in [n]\setminus\{1\},\\
&\beta_{j,i} = \frac{\alpha_2}{n-k} \quad \forall i \in [n]\setminus\{1\}, \forall j \in [n]\setminus\{1,i\}.
\end{aligned} \tag{43}$$

2) If $\alpha_1 > \frac{(n-k-1)\alpha_2}{n-k}$, the optimal average repair cost is $\frac{1}{n}(\frac{(n-1)\alpha_1}{n-k} + (n-1)(\alpha_1 + \frac{(k-1)\alpha_2}{n-k})) = \frac{1}{n}((n-1)\alpha_1 + \frac{n-1}{n-k})$, which can be achieved if and only if $\boldsymbol{\beta}$ satisfies

$$\begin{aligned}
&\beta_{j,1} = \frac{\alpha_1}{n-k} \quad \forall j \in [n]\setminus\{1\},\\
&\beta_{1,i} = \alpha_1 - \frac{(n-k-1)\alpha_2}{n-k} \quad \forall i \in [n]\setminus\{1\},\\
&\beta_{j,i} = \frac{\alpha_2}{n-k} \quad \forall i \in [n]\setminus\{1\}, \forall j \in [n]\setminus\{1,i\}.
\end{aligned} \tag{44}$$

The detailed proof is given in Appendix D. Theorem 8 shows that if $\alpha_1 \leq \frac{(n-k-1)\alpha_2}{n-k}$, the optimal average repair cost for a two-valued array code with this α_1 is a monotonically decreasing function of α_1; while $\alpha_1 > \frac{(n-k-1)\alpha_2}{n-k}$, the optimal average repair cost for a two-valued array code with this α_1 is a monotonically increasing function of α_1; That means that if α_1 increases from 0, the optimal average repair cost for a two-valued array code with this α_1 will first decrease and then increase. Thus, Theorem 8 indicates that the optimal average repair cost for a two-valued array code will appear when $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k}$ as shown in the following theorem.

Theorem 9: In an $(n,k,\mathbf{s},\mathbf{t})$ DSS with one moderate-cost node and $n-1$ low-cost nodes, the HMSR codes regarding the average repair cost are the (n,k) two-valued array codes with the optimal average repair cost. If $k \leq n-2$ and $\mathbf{t}$ is an all-one vector, the optimal average repair cost among all (n,k) two-valued array codes is $\frac{1}{n}((n-1)\frac{n-k-1}{k(n-k)-1} + \frac{n-1}{n-k})$, which can be achieved if and only if an (n,k) two-valued array code has $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfying $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k} = \frac{n-k-1}{k(n-k)-1}$, and (43).

Proof: For a two-valued data allocation vector $\boldsymbol{\alpha}$, we have (42). Thus, $\alpha_2 = \frac{1-\alpha_1}{k-1}$. If $\alpha_1 < \frac{(n-k-1)\alpha_2}{n-k}$, we have $\alpha_1 < \frac{(n-k-1)}{n-k} \cdot \frac{1-\alpha_1}{k-1}$, which implies $0 \leq \alpha_1 < \frac{n-k-1}{k(n-k)-1}$. Define a function of α_1 as $g_1(\alpha_1) \triangleq \frac{(n-1)\alpha_1}{n-k} + \frac{(n-1)(n-2)\alpha_2}{n-k}$, where $\alpha_2 = \frac{1-\alpha_1}{k-1}$. From Theorem 8, if $\alpha_1 < \frac{(n-k-1)\alpha_2}{n-k}$, we have $nC_R^{ave}(\boldsymbol{\beta}) \geq g_1(\alpha_1) = \frac{(n-1)\alpha_1}{n-k} + \frac{(n-1)(n-2)}{n-k} \cdot \frac{1-\alpha_1}{k-1} = (\frac{n-1}{n-k} - \frac{(n-1)(n-2)}{(n-k)(k-1)})\alpha_1 + \frac{(n-1)(n-2)}{(n-k)(k-1)}$. Since $0 \leq \alpha_1 < \frac{n-k-1}{k(n-k)-1}$ and $\frac{n-1}{n-k} - \frac{(n-1)(n-2)}{(n-k)(k-1)} < 0$, we have $nC_R^{ave}(\boldsymbol{\beta}) \geq g_1(\alpha_1) > g_1(\frac{n-k-1}{k(n-k)-1})$. If $\frac{(n-k-1)\alpha_2}{n-k} \leq \alpha_1 \leq \alpha_2$, we have $\frac{n-k-1}{k(n-k)-1} \leq \alpha_1 \leq \frac{1}{k}$. Define a function of α_1 as $g_2(\alpha_1) \triangleq \frac{(n-1)\alpha_1}{n-k} + (n-1)(\alpha_1 + \frac{(k-1)\alpha_2}{n-k})$, where $\alpha_2 = \frac{1-\alpha_1}{k-1}$. From Theorem 8, if $\alpha_1 \geq \frac{(n-k-1)\alpha_2}{n-k}$, we have $nC_R^{ave}(\boldsymbol{\beta}) \geq g_2(\alpha_1) = \frac{(n-1)\alpha_1}{n-k} + (n-1)(\alpha_1 + \frac{(k-1)}{n-k} \cdot \frac{1-\alpha_1}{k-1}) = (n-1)\alpha_1 + \frac{n-1}{n-k}$. Since $\frac{n-k-1}{k(n-k)-1} \leq \alpha_1 \leq \frac{1}{k}$ and $n-1 > 0$, we have $nC_R^{ave}(\boldsymbol{\beta}) \geq g_2(\alpha_1) \geq g_2(\frac{n-k-1}{k(n-k)-1}) = (n-1)\frac{n-k-1}{k(n-k)-1} + \frac{n-1}{n-k}$, and the equality holds if and only if $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfy $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k} = \frac{n-k-1}{k(n-k)-1}$ and (44). When $\alpha_1 = \frac{n-k-1}{k(n-k)-1}$, we have $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k}$ and $g_2(\alpha_1) = \frac{(n-1)\alpha_1}{n-k} + (n-1)(\alpha_1 + \frac{(k-1)\alpha_2}{n-k}) = \frac{(n-1)\alpha_1}{n-k} + (n-1)\frac{(n-2)\alpha_2}{n-k} = g_1(\alpha_1)$. In conclusion, if $\alpha_1 < \frac{(n-k-1)\alpha_2}{n-k}$, we have $nC_R^{ave}(\boldsymbol{\beta}) > g_1(\frac{n-k-1}{k(n-k)-1}) = g_2(\frac{n-k-1}{k(n-k)-1})$. If $\alpha_1 \geq \frac{(n-k-1)\alpha_2}{n-k}$, we have $nC_R^{ave}(\boldsymbol{\beta}) \geq g_2(\frac{n-k-1}{k(n-k)-1})$. Thus, $nC_R^{ave}(\boldsymbol{\beta}) \geq g_2(\frac{n-k-1}{k(n-k)-1}) = (n-1)\frac{n-k-1}{k(n-k)-1} + \frac{n-1}{n-k}$ and the equality holds if $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k} = \frac{n-k-1}{k(n-k)-1}$ and (44). Note that, when $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k}$, conditions (44) and (43) are the same. As (43) has a simpler form, we use (43) in the theorem. ∎

2) Non-existence of linear exact-repair codes: The optimal average repair cost derived in Theorem 9 is for functional-repair codes. If restricting to the linear exact-repair codes, we will show that this optimal average repair cost is unachievable when $k \geq 3$, which means there are no linear exact-repair HMSR codes for this case. Assuming there is an (n,k) linear exact-repair two-valued array code $\mathcal{C}_O$ achieving the optimal $C_R^{ave}(\boldsymbol{\beta})$ in Theorem 9. It is shown in Theorem 9 that $\mathcal{C}_O$ has $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfying $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k} = \frac{n-k-1}{k(n-k)-1}$ and (43). Assume $\mathcal{C}_O$ is over a finite field $\mathbb{F}$ and has data size B. Let $\mathbf{c}_i$ denote the coded vector in node i. Clearly, $\mathbf{c}_1 \in \mathbb{F}^{\alpha_1 B}$ and $\mathbf{c}_i \in \mathbb{F}^{\alpha_2 B}$ for all $i \in [n]\setminus\{1\}$. Then, $\mathbf{c}_{k+1}, \ldots, \mathbf{c}_n$ can all be written as linear combinations of $\mathbf{c}_1, \ldots, \mathbf{c}_k$. Formally,

$$\mathbf{c}_i = \sum_{j=1}^{k} \mathbf{A}_{i,j}\mathbf{c}_j \quad \forall i \in [n]\setminus[k]. \tag{45}$$

Note that $\mathbf{A}_{i,1} \in \mathbb{F}^{\alpha_2 B\times\alpha_1 B}$ for all $i \in [n]\setminus[k]$ and $\mathbf{A}_{i,j} \in \mathbb{F}^{\alpha_2 B\times\alpha_2 B}$ for all $i \in [n]\setminus[k]$ and $j \in [k]\setminus\{1\}$. Since $\mathcal{C}_O$ enables us to reconstruct the file from any k out of n nodes, we have the following lemma showing every $\mathbf{A}_{i,j}$ has full rank.

Lemma 2: $\mathsf{rank}(\mathbf{A}_{i,1}) = \alpha_1 B$ for all $i \in [n]\setminus[k]$ and $\mathsf{rank}(\mathbf{A}_{i,j}) = \alpha_2 B$ for all $i \in [n]\setminus[k]$ and $j \in [k]\setminus\{1\}$.

Proof: For any $i \in [n]\setminus[k]$ and any $j \in [k]$, we have

$$\mathbf{A}_{i,j}\mathbf{c}_j = \mathbf{c}_i - \sum_{t\in[k]\setminus\{j\}} \mathbf{A}_{i,t}\mathbf{c}_t. \tag{46}$$

Given $\{\mathbf{c}_t\}_{t\in[k]\setminus\{j\}}$ and $\mathbf{c}_i$, $\mathbf{c}_j$ should be recovered from (46). However, if $\mathsf{rank}(\mathbf{A}_{i,j}) < \alpha_j B$, we cannot obtain a unique $\mathbf{c}_j$ by (46). Thus, $\mathsf{rank}(\mathbf{A}_{i,j}) \geq \alpha_j B$, which, together with $\mathbf{A}_{i,j} \in \mathbb{F}^{\alpha_i B\times\alpha_j B}$, implies $\mathsf{rank}(\mathbf{A}_{i,j}) = \alpha_j B$. This completes the proof. ∎

Note that the $\boldsymbol{\beta}$ of $\mathcal{C}_O$ satisfies (43). Thus, for any single failed node i with $i \in [k]\setminus\{1\}$, node 1 transmits nothing. Let $\mathbf{S}_{i,j}\mathbf{c}_j$ denote the transmission symbols from node j for all $j \in [n]\setminus\{1,i\}$ to the failed node i. Thus, $\mathbf{S}_{i,j}$ is an $\frac{\alpha_2 B}{n-k}$-by-$\alpha_2 B$ matrix. Via the concept of interference alignment [4], we obtain the following lemma about $\mathbf{S}_{i,j}$.

Lemma 3:

1) $\mathsf{rank}(\mathbf{S}_{i,j}) = \frac{\alpha_2 B}{n-k}$ for all $i \in [n]\setminus\{1\}$ and $j \in [n]\setminus\{1,i\}$.
2) $\mathbf{S}_{i,j}\mathbf{A}_{j,1}$ is zero matrix for all $i \in [k]\setminus\{1\}$ and $j \in [n]\setminus[k]$.
3) For all $i \in [k]\setminus\{1\}$, we have $\mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,i}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,i}\end{bmatrix}) = \alpha_2 B$ and $\mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,j}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,j}\end{bmatrix}) = \frac{\alpha_2 B}{n-k}$ for all $j \in [k]\setminus\{1,i\}$.

Proof:

1) For an $i \in [n]\setminus\{1\}$, let us assume there is $t \in [n]\setminus\{1,i\}$ such that $\mathsf{rank}(\mathbf{S}_{i,t}) \neq \frac{\alpha_2 B}{n-k}$. Since $\mathbf{S}_{i,t} \in \mathbb{F}^{\frac{\alpha_2 B}{n-k}\times\alpha_2 B}$, we have $\mathsf{rank}(\mathbf{S}_{i,t}) < \frac{\alpha_2 B}{n-k}$. Let $T = \mathsf{rank}(\mathbf{S}_{i,t})$. $\mathbf{S}_{i,t}$ can be decomposed into $\mathbf{S}_{i,t} = \mathbf{L}\mathbf{R}$ where $\mathbf{L}$ is an $\frac{\alpha_2 B}{n-k}$-by-T matrix and $\mathbf{R}$ is a T-by-$\alpha_2 B$ full rank matrix. Thus, we can repair node i by downloading $\mathbf{S}_{i,j}\mathbf{c}_j$ from node j for all $j \in [n]\setminus\{1,i,t\}$ and downloading $\mathbf{R}\mathbf{c}_t$ from node t. That means this code has $\boldsymbol{\beta}$ satisfying $\beta_{t,i} = \frac{T}{B} < \frac{\alpha_2}{n-k}$ and $\beta_{j,i} = \frac{\alpha_2}{n-k}$ for all $j \in [n]\setminus\{1,i,t\}$, which violates (43). Thus, we have $\mathsf{rank}(\mathbf{S}_{i,j}) = \frac{\alpha_2 B}{n-k}$ for all $i \in [n]\setminus\{1\}$ and $j \in [n]\setminus\{1,i\}$.
2) To repair node i with $i \in [k]\setminus\{1\}$, we download $\mathbf{S}_{i,j}\mathbf{c}_j$ for all $j \in [n]\setminus\{1,i\}$. From (45), we have

$$\mathbf{S}_{i,j}\mathbf{c}_j = \mathbf{S}_{i,j}\sum_{t=1}^{k}\mathbf{A}_{j,t}\mathbf{c}_t \quad \forall j \in [n]\setminus[k]. \tag{47}$$

To repair $\mathbf{c}_i$ with $i \in [k]\setminus\{1\}$ from (47), we need to eliminate the interference of $\mathbf{c}_j$ for $j \in [k]\setminus\{i\}$ by using $\mathbf{S}_{i,j}\mathbf{c}_j$ for $j \in [k]\setminus\{1,i\}$. Thus, since we can repair $\mathbf{c}_i$ using $\mathbf{S}_{i,j}\mathbf{c}_j$ for all $j \in [n]\setminus\{1,i\}$ via (47), we have

$$\mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,1}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,1}\end{bmatrix}) \leq \mathsf{rank}(\mathbf{S}_{i,1}) = 0, \tag{48}$$

$$\mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,i}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,i}\end{bmatrix}) = \alpha_2 B, \tag{49}$$

$$\mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,j}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,j}\end{bmatrix}) \leq \mathsf{rank}(\mathbf{S}_{i,j})\, \forall j \in [k]\setminus\{1,i\}. \tag{50}$$

From (48), we obtain that $\mathbf{S}_{i,j}\mathbf{A}_{j,1}$ is zero matrix for all $i \in [k]\setminus\{1\}$ and $j \in [n]\setminus[k]$.
3) From Lemma 2, we know that $\mathbf{A}_{t,j}$ is invertible for all $t \in [n]\setminus[k]$ and for all $j \in [k]\setminus\{1,i\}$. Furthermore, since we have proven Lemma 3.(1), we obtain that $\mathsf{rank}(\mathbf{S}_{i,t}\mathbf{A}_{t,j}) = \frac{\alpha_2 B}{n-k}$ for all $t \in [n]\setminus[k]$ and for all $j \in [k]\setminus\{1,i\}$. Thus, from (50), we derive $\frac{\alpha_2 B}{n-k} \leq \mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,j}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,j}\end{bmatrix}) \leq \mathsf{rank}(\mathbf{S}_{i,j}) = \frac{\alpha_2 B}{n-k}$ for all $j \in [k]\setminus\{1,i\}$, which means $\mathsf{rank}(\begin{bmatrix}\mathbf{S}_{i,k+1}\mathbf{A}_{k+1,j}\\ \vdots \\ \mathbf{S}_{i,n}\mathbf{A}_{n,j}\end{bmatrix}) = \frac{\alpha_2}{n-k}$ for all $j \in [k]\setminus\{1,i\}$.

∎

From Lemmas 2 and 3, the next theorem shows that $\mathcal{C}_O$ does not exist.

Theorem 10: Consider an $(n,k,\mathbf{s},\mathbf{t})$ DSS with one moderate-cost node and $n-1$ low-cost nodes. If $3 \leq k \leq n-2$ and $\mathbf{t}$ is an all-one vector, the optimal average repair cost among all (n,k) two-valued array codes cannot be achieved by an (n,k) linear exact-repair two-valued array code. That means no linear exact-repair code can achieve the HMSR point regarding the average repair cost for such a DSS.

Proof: We must prove that $\mathcal{C}_O$ does not exist. For $\mathcal{C}_O$, we already prove Lemmas 2 and 3. From Lemma 3, $\mathbf{S}_{2,j}\mathbf{A}_{j,1}$ and $\mathbf{S}_{3,j}\mathbf{A}_{j,1}$ are zero matrices for all $j \in [n]\setminus[k]$. Given a matrix $\mathbf{A}$, let $<\mathbf{A}>$ and $<\mathbf{A}>^{\perp}$ denote the vector space spanned by all the column vectors in $\mathbf{A}$ and the corresponding orthogonal space. Then, we have $<\mathbf{S}_{2,j}^{\mathsf{T}}>\subseteq<\mathbf{A}_{j,1}>^{\perp}$ for all $j \in [n]\setminus[k]$. From Lemma 2, we know $\mathsf{rank}(\mathbf{A}_{j,1}) = \alpha_1 B$, which, together with $\mathbf{A}_{j,1}$ is an $\alpha_2 B$-by-$\alpha_1 B$ matrix, leads to $\dim(<\mathbf{A}_{j,1}>^{\perp}) = \alpha_2 B - \alpha_1 B$. As $\alpha_1 = \frac{(n-k-1)\alpha_2}{n-k} = \frac{n-k-1}{k(n-k)-1}$, we have $\dim(<\mathbf{A}_{j,1}>^{\perp}) = \alpha_2 B - \alpha_1 B = \frac{\alpha_2 B}{n-k}$. From Lemma 3, we have $\mathsf{rank}(\mathbf{S}_{2,j}) = \frac{\alpha_2 B}{n-k}$, which implies $\dim(<\mathbf{S}_{2,j}^{\mathsf{T}}>) = \frac{\alpha_2 B}{n-k}$. Since $<\mathbf{S}_{2,j}^{\mathsf{T}}>\subseteq<\mathbf{A}_{j,1}>^{\perp}$ for all $j \in [n]\setminus[k]$, we obtain $<\mathbf{S}_{2,j}^{\mathsf{T}}>=<\mathbf{A}_{j,1}>^{\perp}$ for all $j \in [n]\setminus[k]$. Similarly, we have $<\mathbf{S}_{3,j}^{\mathsf{T}}>=<\mathbf{A}_{j,1}>^{\perp}$, which implies $<\mathbf{S}_{2,j}^{\mathsf{T}}>=<\mathbf{S}_{3,j}^{\mathsf{T}}>$ for all $j \in [n]\setminus[k]$. Then, $\mathbf{S}_{2,j}$ can be written as $\mathbf{S}_{2,j} = \mathbf{U}_j\mathbf{S}_{3,j}$ where $\mathbf{U}_j$ is an $\frac{\alpha_2 B}{n-k}$-by-$\frac{\alpha_2 B}{n-k}$ invertible matrix for all $j \in [n] \setminus [k]$. Thus, we have $\mathsf{rank}(\begin{bmatrix} \mathbf{S}_{2,k+1}\mathbf{A}_{k+1,2} \\ \vdots \\ \mathbf{S}_{2,n}\mathbf{A}_{n,2} \end{bmatrix}) = \mathsf{rank}(\begin{bmatrix} \mathbf{U}_{k+1}\mathbf{S}_{3,k+1}\mathbf{A}_{k+1,2} \\ \vdots \\ \mathbf{U}_{n}\mathbf{S}_{3,n}\mathbf{A}_{n,2} \end{bmatrix}) = \mathsf{rank}(\begin{bmatrix} \mathbf{S}_{3,k+1}\mathbf{A}_{k+1,2} \\ \vdots \\ \mathbf{S}_{3,n}\mathbf{A}_{n,2} \end{bmatrix})$.

However, from Lemma 3, we have $\mathsf{rank}(\begin{bmatrix} \mathbf{S}_{2,k+1}\mathbf{A}_{k+1,2} \\ \vdots \\ \mathbf{S}_{2,n}\mathbf{A}_{n,2} \end{bmatrix}) = \alpha_2 B \neq \frac{\alpha_2 B}{n-k} = \mathsf{rank}(\begin{bmatrix} \mathbf{S}_{3,k+1}\mathbf{A}_{k+1,2} \\ \vdots \\ \mathbf{S}_{3,n}\mathbf{A}_{n,2} \end{bmatrix})$, which is a contradiction. ■

B. HMSR codes regarding the worst-case repair cost

A DSS with one moderate-cost node and $n-1$ low-cost nodes is considered. For the considered DSS, the HMSR codes regarding the worst-case repair cost are the (n,k) two-valued array codes with optimal worst-case repair cost. Similar to the last subsection, it is important to discuss how the repair cost changes as α_1 increases. If $\alpha_1 = 0$, the repair cost for node 1 is zero, which is optimal. However, the capacities of the other nodes are large, which leads to high repair costs. To reduce the worst-case repair cost for the whole system, we should increase the repair cost for node 1 and reduce the repair cost for the other nodes by increasing α_1. The following two lemmas fully characterize how the repair cost for each node changes as α_1 increases.

Lemma 4: For an $(n,k,\mathbf{s},\mathbf{t})$ DSS, there exists $\tau_1 : [n-1] \to [n]\setminus\{1\}$ such that $t_{\tau_1(1),1} \geq \cdots \geq t_{\tau_1(n-1),1}$ In this DSS, the optimal worst-case repair cost for node 1 of (n,k) two-valued array codes with fixed α_1 is $J_1(\alpha_1) \triangleq \alpha_1/(\frac{1}{t_{\tau_1(1),1}} + \cdots + \frac{1}{t_{\tau_1(n-k),1}})$.

Proof: From Lemma 1, for an (n,k) irregular array code with $\boldsymbol{\alpha}$, we have

$$\beta_{\tau_1(j),1} \geq 0 \quad \forall j \in [n-1] \tag{51}$$

$$\sum_{j\in\mathcal{S}} \beta_{\tau_1(j),1} \geq \alpha_1 \quad \forall \mathcal{S} \subseteq [n-1] \text{ with } |\mathcal{S}| = n-k. \tag{52}$$

The minimization of $C^{wor}_{R_1}(\boldsymbol{\beta}) = \max_{j\in[n-1]} t_{\tau_1(j),1}\beta_{\tau_1(j),1}$ subject to (51) and (52) has the same problem form as Problem 3.B. From Theorem 5, which gives the optimal value G_i to Problem 3.B, the optimal worst-case repair cost for node 1 of (n,k) two-valued array codes with fixed α_1 is $\alpha_1/(\frac{1}{t_{\pi_i(1),i}} + \cdots + \frac{1}{t_{\pi_i(n-k),i}})$. ■

Lemma 5: For an $(n,k,\mathbf{s},\mathbf{t})$ DSS and all $i \in [n]\setminus\{1\}$, there exists $\tau_i : [n-2] \to [n]\setminus\{1,i\}$ such that $t_{\tau_i(1),i} \geq \cdots \geq t_{\tau_i(n-2),i}$. For all $i \in [n]\setminus\{1\}$, let $\alpha^*_{1,i}$ be the largest α_1 that satisfies (42) and $\alpha_1 \leq \alpha_2 \frac{t^{-1}_{\tau_i(1),i}+\cdots+t^{-1}_{\tau_i(n-k-1),i}+t^{-1}_{1,i}}{t^{-1}_{\tau_i(1),i}+\cdots+t^{-1}_{\tau_i(n-k),i}}$. The optimal worst-case repair cost for node i for all $i \in [n]\setminus\{1\}$ of (n,k) two-valued array codes with fixed α_1 is

$$J_i(\alpha_1) \triangleq \begin{cases} \frac{1-\alpha_1}{k-1}/K & 0 \leq \alpha_1 \leq \alpha^*_{1,i}, \\ \alpha_1/L & \alpha^*_{1,i} < \alpha_1 \leq \frac{1}{k}, \end{cases} \tag{53}$$

where $K = (\frac{1}{t_{\tau_i(1),i}} + \cdots + \frac{1}{t_{\tau_i(n-k),i}})$ and $L = (\frac{1}{t_{1,i}} + \frac{1}{t_{\tau_i(1),i}} + \cdots + \frac{1}{t_{\tau_i(n-k-1),i}})$.

Proof: The proof of Lemma 5 is given in Appendix E. ■

Lemma 4 shows that the optimal worst-case repair cost for node 1 will increase as α_1 increases. Lemma 5 indicates that if $\alpha^*_{1,i} < \frac{1}{k}$, the optimal worst-case repair cost for node i will first decrease then increase as α_1 increases; while if $\alpha^*_{1,i} = \frac{1}{k}$, the optimal worst-case repair cost for node i will decrease as α_1 increases. Note that $\alpha^*_{1,i}$ depends on $\mathbf{t}$ and there exists $\mathbf{t}$ such that $\alpha^*_{1,i} = \frac{1}{k}$ for all $i \in [n]\setminus\{1\}$. If $\alpha^*_{1,i} = \frac{1}{k}$ for all $i \in [n]\setminus\{1\}$, the optimal worst-case repair cost for node i for all $i \in [n]\setminus\{1\}$ will decrease as α_1 increase. Furthermore, if the repair cost of node 1 is always smaller than the repair costs for the other nodes, the worst-case repair cost for the whole system can be minimized only if $\alpha_1 = \frac{1}{k}$. That means an (n,k) two-valued array code with the optimal worst-case repair cost must have $\alpha_1 = \frac{1}{k}$, which leads to an (n,k) MDS array code. Thus, in this case, the HMSR code regarding the worst-case repair cost is an (n,k) MDS array code with the optimal worst-case repair cost, which has been constructed in Section IV-B. However, for some other cases, an (n,k) two-valued array code with the optimal worst-case repair cost cannot have $\alpha_1 = \frac{1}{k}$, which implies that a regular array code cannot achieve the HMSR point. Specifically, the α_1 for the (n,k) two-valued array code with the optimal worst-case repair cost, which is the HMSR code regarding the worst-case repair cost for a DSS with only one moderate-cost node, is characterized in the following theorem.

Theorem 11: In an $(n,k,\mathbf{s},\mathbf{t})$ DSS with one moderate-cost node and $n-1$ low-cost nodes, the HMSR codes regarding the worst-case repair cost are the (n,k) two-valued array codes with the optimal worst-case repair cost. The optimal worst-case repair cost of an (n,k) two-valued array code is

$$\min_{0\leq\alpha_1\leq\frac{1}{k}} \max\{J_1(\alpha_1),\ldots,J_n(\alpha_1)\}.$$

There is a unique α_1, denoted as α^*_1, that minimizes $\max\{J_1(\alpha_1),\ldots,J_n(\alpha_1)\}$. Thus, an HMSR code regarding the worst-case repair cost must be an (n,k) two-valued array code with $\alpha_1 = \alpha^*_1$.

VI. Conclusion

In this paper, we consider a general class of heterogeneous DSSs, denoted as $(n,k,\mathbf{s},\mathbf{t})$ DSSs. We consider the average and worst-case repair costs and investigate whether an exact-repair code can achieve the HMSR point. We show that a DSS *almost surely* has no moderate-cost node and a DSS with at least one moderate-cost node *almost surely* has only one moderate-cost node. Thus, we focus on the DSSs with no moderate-cost node or only one moderate-cost node. For a DSS with no moderate-cost node, it is shown that an HMSR code must be an MDS array code. Furthermore, an exact-repair HMSR code regarding the average repair cost has been constructed. Also, an exact-repair HMSR code regarding the worst-case repair cost has been constructed. For a DSS with only one moderate-cost node, an HMSR code must be a two-valued MDS array code. For the worst-case repair cost and some $\mathbf{t}$, the exact-repair HMSR code can be constructed. For the average repair cost and $\mathbf{t}$ being an all-one vector, we show that a linear exact-repair code cannot achieve the HMSR point.

There are some challenging issues that remain unsolved:

1) For a DSS with only one moderate-cost node, the HMSR point for some $\mathbf{t}$ cannot be achieved by regular array codes. How do we construct exact-repair codes to achieve such an HMSR point?
2) The HMSR point for some $(\mathbf{s}, \mathbf{t})$ cannot be achieved by linear exact-repair codes. What is the "HMSR" point for linear exact-repair codes for such $(\mathbf{s}, \mathbf{t})$?
3) We only consider one-hop repair schemes. What is the HMSR point if allowing multi-hop transmissions in a repair process?

Appendix A
Proof of Theorem 1

In this section, we explicitly obtain the set of optimal solutions to Problem 1,

Problem 1: Given n, k, and $\mathbf{s} = [s_1 \, \dots \, s_n]$,

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \quad & C_S(\boldsymbol{\alpha}) \\ \text{s.t.} \quad & \alpha_i \geq 0 \quad \forall i \in [n], & (54) \\ & \sum_{i \in \mathcal{S}} \alpha_i \geq 1 \quad \forall \mathcal{S} \subseteq [n] \text{ with } |\mathcal{S}| = k. & (55) \end{aligned}$$

First, we simplify this problem by analyzing the characteristics of optimal solutions. It is obvious that if $s_1 = s_2$, two data allocation vectors $[\alpha_1 \, \alpha_2 \, \alpha_3 \, \dots \, \alpha_n]$ and $[\alpha_2 \, \alpha_1 \, \alpha_3 \, \dots \, \alpha_n]$ lead to the same storage cost. Thus, given an optimal solution $\boldsymbol{\alpha} = [\alpha_1 \, \dots \, \alpha_n]$, $[\alpha_2 \, \alpha_1 \, \alpha_3 \dots \alpha_n]$ is also an optimal solution. This means whether there exists $i, j \in [n]$ satisfying $s_i = s_j$ affects the set of optimal solutions. Based on this consideration, given $\mathbf{s}$, we define an equivalence relation on the set $[n]$: $i \sim j$ if and only if $s_i = s_j$. A permutation ϕ on $[n]$ is said to preserve the equivalence if $\phi(i) \sim i$ for all $i \in [n]$. Given a permutation ϕ on $[n]$ and a vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ satisfying (54) and (55), it is easy to show that $\boldsymbol{\alpha}_\phi \triangleq [\alpha_{\phi(1)} \, \dots \, \alpha_{\phi(n)}]$ also satisfies (54) and (55). Furthermore, if ϕ preserves equivalence, we have $C_S(\boldsymbol{\alpha}) = C_S(\boldsymbol{\alpha}_\phi)$, which means if $\boldsymbol{\alpha}$ is an optimal solution, so is $\boldsymbol{\alpha}_\phi$. Based on this, we obtain the following lemma, revealing more properties about the optimal solutions.

Lemma 6: There exists an optimal solution $\boldsymbol{\alpha}$ to Problem 1 satisfying $\alpha_1 \leq \dots \leq \alpha_k = \alpha_{k+1} = \dots = \alpha_n$ and $\sum_{i \in [k]} \alpha_i = 1$. Let $\mathcal{Z}$ denote the set of optimal solutions satisfying the conditions above. Let Φ denote the set of the permutations preserving the equivalence. The optimal solution set $\mathcal{V}$ of Problem 1 is

$$\mathcal{V} = \left\{ \boldsymbol{\alpha}_\phi \middle| \boldsymbol{\alpha} \in \mathcal{Z}, \phi \in \Phi \right\}. \tag{56}$$

Proof: Given an optimal solution $\boldsymbol{\alpha} = [\alpha_1 \dots \alpha_n]$ to Problem 1, there exists a permutation $\phi \in \Phi$ such that $\alpha_{\phi(i)} \leq \alpha_{\phi(j)}$ if $\phi(i) \sim \phi(j)$ and $\phi(i) \leq \phi(j)$. Thus, $\boldsymbol{\alpha}_\phi$ is also an optimal solution. Let $\boldsymbol{\alpha}' = [\alpha'_1 \, \dots \, \alpha'_n] = \boldsymbol{\alpha}_\phi$, and we have $\alpha'_i \leq \alpha'_j$ if $i \sim j$ and $i \leq j$. Then, we prove by contradiction that $\alpha'_i \leq \alpha'_j$ also holds if $i \leq j$ and $i \not\sim j$. Let us assume there exist $i, j \in [n]$ such that $i \leq j$, $i \not\sim j$, and $\alpha'_i > \alpha'_j$. Since $i \leq j$ and $s_1 \geq \dots \geq s_n$, we have $s_i \geq s_j$. As $i \not\sim j$, we further have $s_i > s_j$. Consider the permutation ψ on $[n]$ that only swaps i and j. Then, $\boldsymbol{\alpha}'_\psi \triangleq [\alpha'_{\psi(1)} \, \dots \, \alpha'_{\psi(n)}]$, which is obtained from $\boldsymbol{\alpha}'$ by swapping the value of α'_i and α'_j, is also a solution to Problem 1. From the relation between $\boldsymbol{\alpha}'$ and $\boldsymbol{\alpha}'_\psi$, we have $C_S(\boldsymbol{\alpha}'_\psi) = C_S(\boldsymbol{\alpha}') - s_i\alpha'_i - s_j\alpha'_j + s_i\alpha'_j + s_j\alpha'_i = C_S(\boldsymbol{\alpha}') + (s_i - s_j)(\alpha'_j - \alpha'_i) < C_S(\boldsymbol{\alpha}')$, which implies that $\boldsymbol{\alpha}'$ is not an optimal solution. This is a contradiction. Thus, $\alpha'_i \leq \alpha'_j$ if $i \leq j$ and $i \not\sim j$. Furthermore, $\alpha'_i \leq \alpha'_j$ if $i \leq j$, which means $\boldsymbol{\alpha}'$ satisfies $\alpha'_1 \leq \dots \leq \alpha'_n$.

Next, we will prove that if $\sum_{i \in [k]} \alpha'_i \neq 1$ or $\alpha'_t \neq \alpha'_k$ for some $t \in [n]$ with $t > k$, we can obtain another feasible solution $\boldsymbol{\alpha}^*$ with $C_S(\boldsymbol{\alpha}^*) < C_S(\boldsymbol{\alpha}')$, which is a contradiction.

1) Assume $\alpha'_t \neq \alpha'_k$ for some $t > k$. As $\alpha'_1 \leq \dots \leq \alpha'_n$, we have $\alpha'_t > \alpha'_k$. Let $\boldsymbol{\alpha}^* = [\alpha^*_1 \dots \alpha^*_n] = [\alpha'_1 \dots \alpha'_k \, \alpha'_k \, \dots \, \alpha'_k]$, where $\alpha^*_i = \alpha'_i$ for all $i \in [k]$ and $\alpha^*_i = \alpha'_k$ for all $i \in [n] \setminus [k]$. Based on the fact that $\boldsymbol{\alpha}'$ is a feasible solution, it is easy to verify that $\boldsymbol{\alpha}^*$ is also a feasible solution to Problem 1. First, $\boldsymbol{\alpha}^*$ clearly satisfies (54). Second, as $\alpha'_1 \leq \dots \leq \alpha'_n$, we have $\sum_{i \in \mathcal{S}} \alpha^*_i \geq \sum_{i \in [k]} \alpha^*_i = \sum_{i \in [k]} \alpha'_i$ for all $\mathcal{S} \subseteq [n]$ with $|\mathcal{S}| = k$. Since $\boldsymbol{\alpha}'$ satisfies (55), we obtain $\sum_{i \in [k]} \alpha'_i \geq 1$, which leads to $\sum_{i \in \mathcal{S}} \alpha^*_i \geq 1$ for all $\mathcal{S} \subseteq [n]$ with $|\mathcal{S}| = k$. Thus, $\boldsymbol{\alpha}^*$ is a solution to Problem 1 with $C_S(\boldsymbol{\alpha}^*) = C_S(\boldsymbol{\alpha}') - \sum_{i=k+1}^n s_i(\alpha'_i - \alpha'_k) \leq C_S(\boldsymbol{\alpha}') - s_t(\alpha'_t - \alpha'_k) < C_S(\boldsymbol{\alpha}')$. This is a contradiction.
2) Assume $\sum_{i \in [k]} \alpha'_i \neq 1$. Let $X \triangleq \sum_{i \in [k]} \alpha'_i$. Since $\boldsymbol{\alpha}'$ satisfies (55), $X > 1$. Consider $\boldsymbol{\alpha}^* = \frac{1}{X} \cdot \boldsymbol{\alpha}'$. Clearly, $\boldsymbol{\alpha}^*$ satisfies (54). Then, for all $\mathcal{S} \subseteq [n]$ with $|\mathcal{S}| = k$, we have $\sum_{i \in \mathcal{S}} \alpha^*_i \geq \sum_{i \in [k]} \alpha^*_i = \frac{1}{X} \cdot \sum_{i \in [k]} \alpha'_i = 1$. Thus, $\boldsymbol{\alpha}^*$ is a feasible solution to Problem 1 with $C_S(\boldsymbol{\alpha}^*) = \frac{1}{X} \cdot C_S(\boldsymbol{\alpha}') < C_S(\boldsymbol{\alpha}')$. This is a contradiction.

Given an optimal solution $\boldsymbol{\alpha} = [\alpha_1 \dots \alpha_n] \in \mathbb{R}^n$ to Problem 1, we have proven that there exists a permutation $\phi \in \Phi$ such that $\boldsymbol{\alpha}_\phi \in \mathcal{Z}$. Since there definitely exists an optimal solution to Problem 1, there exists an optimal solution $\boldsymbol{\alpha}$ satisfying $\alpha_1 \leq \dots \leq \alpha_k = \alpha_{k+1} = \dots = \alpha_n$ and $\sum_{i \in [k]} \alpha_i = 1$.

Next, we discuss the relation between $\mathcal{V}$ and $\mathcal{Z}$. Note that $\mathcal{V}$ is defined to be the optimal solution set of Problem 1. Given any element $\boldsymbol{\alpha}$ in $\mathcal{V}$, we have proven that there exists a permutation $\phi \in \Phi$ such that $\boldsymbol{\alpha}_\phi \in \mathcal{Z}$. Let $\boldsymbol{\alpha}' = [\alpha'_1 \dots \alpha'_n] = [\alpha_{\phi(1)} \dots \alpha_{\phi(n)}]$. Then, we have $\alpha'_i = \alpha_{\phi(i)}$ and further $\alpha'_{\phi^{-1}(i)} = \alpha_{\phi(\phi^{-1}(i))} = \alpha_i$. Thus, we have $\boldsymbol{\alpha}'_{\phi^{-1}} = [\alpha'_{\phi^{-1}(1)} \dots \alpha'_{\phi^{-1}(n)}] = [\alpha_1 \dots \alpha_n] = \boldsymbol{\alpha}$. As $\boldsymbol{\alpha}' = \boldsymbol{\alpha}_\phi \in \mathcal{Z}$ and $\phi^{-1} \in \Phi$, there are $\boldsymbol{\alpha}' \in \mathcal{Z}$ and $\phi^{-1} \in \Phi$ such that $\boldsymbol{\alpha} = \boldsymbol{\alpha}'_{\phi^{-1}} \in \left\{ \boldsymbol{\alpha}_\phi \middle| \boldsymbol{\alpha} \in \mathcal{Z}, \phi \in \Phi \right\}$ for any given $\boldsymbol{\alpha} \in \mathcal{V}$. Thus, $\mathcal{V} \subseteq \left\{ \boldsymbol{\alpha}_\phi \middle| \boldsymbol{\alpha} \in \mathcal{Z}, \phi \in \Phi \right\}$. Conversely, given any $\boldsymbol{\alpha} \in \mathcal{Z}$ and any $\phi \in \Phi$, as $\boldsymbol{\alpha}$ is an optimal solution and ϕ preserves equivalence, $\boldsymbol{\alpha}_\phi$ is also an optimal solution and thus in $\mathcal{V}$. Thus, $\left\{ \boldsymbol{\alpha}_\phi \middle| \boldsymbol{\alpha} \in \mathcal{Z}, \phi \in \Phi \right\} \subseteq \mathcal{V}$ and further $\mathcal{V} = \left\{ \boldsymbol{\alpha}_\phi \middle| \boldsymbol{\alpha} \in \mathcal{Z}, \phi \in \Phi \right\}$. ∎

Lemma 6 demonstrates that to obtain the optimal solution set $\mathcal{V}$, we only need to obtain $\mathcal{Z}$. For the optimal solutions in $\mathcal{Z}$, the objective function of Problem 1 can be written as $C_S(\boldsymbol{\alpha}) = \sum_{i\in[k-1]} s_i\alpha_i + \alpha_k \sum_{i=k}^n s_i$. Hence, to obtain $\mathcal{Z}$, we turn to the following problem.

Problem 5: Given n, k, and $\mathbf{s} = [s_1 \dots s_n]$ with $s_1 \geq s_2 \cdots \geq s_n > 0$,

$$\min_{[\alpha_1\dots\alpha_k]\in\mathbb{R}^k} \quad \sum_{i\in[k-1]} s_i\alpha_i + \alpha_k \sum_{i=k}^n s_i$$

$$\text{s.t.} \quad 0 \leq \alpha_1 \leq \cdots \leq \alpha_k, \tag{57}$$

$$\sum_{i\in[k]} \alpha_i = 1. \tag{58}$$

Clearly, $\mathcal{Z}$ is the optimal solution set of Problem 5. Before showing the optimal solutions to Problem 5, we prove some properties of $y_p(n,k,\mathbf{s})$ (cf. (4)) for all $p \in [k-1]$.

Lemma 7:

1) $y_1(n,k,\mathbf{s}) \geq y_2(n,k,\mathbf{s}) \geq \cdots \geq y_{k-1}(n,k,\mathbf{s})$.
2) Given $p \in [k-1]\setminus\{1\}$, $y_p(n,k,\mathbf{s}) = y_{p-1}(n,k,\mathbf{s})$ if and only if $s_{p-1} = s_p$.

Proof: Since $s_1 \geq \cdots \geq s_n$, $y_p(n,k,\mathbf{s}) - y_{p-1}(n,k,\mathbf{s}) = (k-p)s_p - \sum_{i=p+1}^n s_i - ((k-p+1)s_{p-1} - \sum_{i=p}^n s_i) = (k-p)(s_p - s_{p-1}) - s_{p-1} - \sum_{i=p+1}^n s_i + \sum_{i=p}^n s_i = (k-p)(s_p - s_{p-1}) - s_{p-1} + s_p = (k-p+1)(s_p - s_{p-1}) \leq 0$, and the equality holds if and only if $s_{p-1} = s_p$. ∎

With Lemma 7, we can solve Problem 5.

Theorem 12: $[\alpha_1 \dots \alpha_k]$ is an optimal solution to Problem 5 if and only if

$$0 \leq \alpha_1 \leq \cdots \leq \alpha_k, \tag{59}$$

$$\sum_{i\in[k]} \alpha_i = 1, \tag{60}$$

$$\alpha_i = 0 \text{ if } y_i(n,k,\mathbf{s}) > 0, \tag{61}$$

$$\alpha_i = \alpha_k \text{ if } y_i(n,k,\mathbf{s}) < 0. \tag{62}$$

Proof: The Lagrangian [28] associated with Problem 5 is defined as $L = \sum_{i\in[k-1]} s_i\alpha_i + \alpha_k \sum_{i=k}^n s_i + \lambda_1(-\alpha_1) + \lambda_2(\alpha_1 - \alpha_2) + \cdots + \lambda_k(\alpha_{k-1} - \alpha_k) + u(\alpha_1 + \cdots + \alpha_k - 1)$ where u and λ_i for all $i \in [k]$ are dual variables associated with the constraints. The Karush-Kuhn-Tucker (KKT) conditions [28] besides the constraints (57) and (58) are as follows:

$$s_i - \lambda_i + \lambda_{i+1} + u = 0, \quad \forall i \in [k-1] \tag{63}$$

$$\sum_{i=k}^n s_i - \lambda_k + u = 0 \tag{64}$$

$$\lambda_1\alpha_1 = 0 \tag{65}$$

$$\lambda_i(\alpha_{i-1} - \alpha_i) = 0, \quad \forall i \in [k]\setminus\{1\} \tag{66}$$

$$\lambda_i \geq 0 \quad i \in [k]. \tag{67}$$

Note that $[\alpha_1 \dots \alpha_k]$ is an optimal solution to Problem 5 if and only if there exists $[u\,\lambda_1 \dots \lambda_k]$ that, together with $[\alpha_1 \dots \alpha_k]$, satisfies the KKT conditions.

Given an optimal solution $[\alpha_1 \dots \alpha_k]$ and some $p \in [k-1]$, from the KKT conditions, we prove that (a) if $y_{k-p}(n,k,\mathbf{s}) > 0$, $\alpha_{k-p} = 0$; (b) if $y_{k-p}(n,k,\mathbf{s}) < 0$, $\alpha_{k-p} = \alpha_k$.

1) Suppose $y_{k-p}(n,k,\mathbf{s}) > 0$. From (63) and (64), if $p \in [k-1]\setminus\{1\}$, we have

$$\begin{aligned}(\sum_{i=k}^n s_i - \lambda_k + u) + \sum_{i=k-p+1}^{k-1} (s_i - \lambda_i + \lambda_{i+1} + u) \\ = \sum_{i=k-p+1}^n s_i - \lambda_{k-p+1} + pu = 0.\end{aligned}$$

If $p = 1$, from (64), we also have $\sum_{i=k-p+1}^n s_i - \lambda_{k-p+1} + pu = 0$. Thus, as $p \in [k-1]$, we have

$$\sum_{i=k-p+1}^n s_i - \lambda_{k-p+1} + pu = 0. \tag{68}$$

From (63), we have

$$ps_{k-p} - p\lambda_{k-p} + p\lambda_{k-p+1} + pu = 0. \tag{69}$$

From (68) and (69), we have

$$\begin{aligned}p\lambda_{k-p} - (p+1)\lambda_{k-p+1} &= ps_{k-p} - \sum_{i=k-p+1}^n s_i \\ &= y_{k-p}(n,k,\mathbf{s}).\end{aligned} \tag{70}$$

Since $y_{k-p}(n,k,\mathbf{s}) > 0$, we have $p\lambda_{k-p} > (p+1)\lambda_{k-p+1}$. From (67), we obtain $\lambda_{k-p} > \frac{p+1}{p}\lambda_{k-p+1} \geq 0$. When $p = k-1$, from (65), $\lambda_{k-p} > 0$ means $\alpha_1 = 0$. When $p < k-1$, from (66), $\lambda_{k-p} > 0$ means $\alpha_{k-p-1} = \alpha_{k-p}$. Since $y_{k-(k-1)}(n,k,\mathbf{s}) \geq y_{k-(k-2)}(n,k,\mathbf{s}) \geq \dots y_{k-p}(n,k,\mathbf{s}) > 0$ (cf. Lemma 7), we have $\alpha_{k-p} = \alpha_{k-p-1} = \cdots = \alpha_{k-(k-1)} = 0$.

2) Suppose $y_{k-p}(n,k,\mathbf{s}) < 0$. From (70), we derive $p\lambda_{k-p} < (p+1)\lambda_{k-p+1}$, which, together with (67), leads to $\lambda_{k-p+1} > 0$. From (66) and $\lambda_{k-p+1} > 0$, we obtain $\alpha_{k-p} = \alpha_{k-p+1}$. Since $y_{k-1}(n,k,\mathbf{s}) \leq y_{k-2}(n,k,\mathbf{s}) \leq \dots y_{k-p}(n,k,\mathbf{s}) < 0$ (cf. Lemma 7), we have $\alpha_{k-p} = \alpha_{k-p+1} = \cdots = \alpha_k$.

Thus, we obtain the necessary conditions to make $[\alpha_1 \dots \alpha_k]$ an optimal solution, as follows:

$$\alpha_i = 0 \text{ if } y_i(n,k,\mathbf{s}) > 0, \tag{71}$$

$$\alpha_i = \alpha_k \text{ if } y_i(n,k,\mathbf{s}) < 0, \tag{72}$$

$$0 \leq \alpha_1 \leq \cdots \leq \alpha_k, \tag{73}$$

$$\sum_{i\in[k]} \alpha_i = 1. \tag{74}$$

Next, to prove that the solutions satisfying these necessary conditions ((71), (72), (73), and (74)) are optimal, we show that these solutions lead to the same value that must be optimal. If there is no i satisfying $y_i(n,k,\mathbf{s}) = 0$, there is a unique solution satisfying (71), (72), (73), and (74). Since the optimal solution exists, this solution is the optimal solution. Next, we consider the case that there exists i satisfying $y_i(n,k,\mathbf{s}) = 0$. Suppose $y_i(n,k,\mathbf{s}) > 0$ for all $i \in [p_1]$, $y_i(n,k,\mathbf{s}) = 0$ for all $i \in [p_2]\setminus[p_1]$, and $y_i(n,k,\mathbf{s}) < 0$ for all

$i \in [k-1] \setminus [p_2]$. Given a solution satisfying (71), (72), (73), and (74), the value of the objective function is

$$\begin{aligned}
&\sum_{i\in[k-1]} s_i\alpha_i + \alpha_k \sum_{i=k}^{n} s_i \\
&\overset{(a)}{=} \sum_{i=p_1+1}^{p_2} s_i\alpha_i + \sum_{i=p_2+1}^{n} s_i\alpha_k && (75)\\
&\overset{(b)}{=} s_{p_2} \sum_{i=p_1+1}^{p_2} \alpha_i + \sum_{i=p_2+1}^{n} s_i\alpha_k && (76)\\
&\overset{(c)}{=} s_{p_2}(1-(k-p_2)\alpha_k) + \sum_{i=p_2+1}^{n} s_i\alpha_k && (77)\\
&= s_{p_2} - \alpha_k((k-p)s_p - \sum_{i=p+1}^{n} s_i) && (78)\\
&= s_{p_2} - \alpha_k y_{p_2}(n,k,\mathbf{s}) = s_{p_2}, && (79)
\end{aligned}$$

where (a) is due to (72), (b) is due to the second claim of Lemma 7, and (c) is due to (72) and (73). Thus, for all solutions satisfying (71), (72), (73), and (74), the values of the objective function are the same, which implies these solutions are all optimal. ■

Theorem 12 derives all optimal solutions to Problem 5, which leads to

$$\mathcal{Z} = \left\{ [\alpha_1 \, \dots \, \alpha_n] \middle| (59), (60), (61), (62), \alpha_i = \alpha_k \, \forall i \in [n] \setminus [k] \right\}. \quad (80)$$

With the derived $\mathcal{Z}$, from Lemma 6, we can obtain that $\boldsymbol{\alpha} = [\alpha_1 \dots \alpha_n]$ is an optimal solution to Problem 1 if and only if

$$\begin{aligned}
&\alpha_i = 0 \quad \forall i \in \mathcal{N}_H, && (81)\\
&0 \le \alpha_i \le \alpha_k \quad \forall i \in \mathcal{N}_M, && (82)\\
&\alpha_i = \alpha_k \quad \forall i \in \mathcal{N}_L, && (83)\\
&\sum_{i\in[k]} \alpha_i = 1. && (84)
\end{aligned}$$

The proof is as follows. Since $y_1(n,k,\mathbf{s}) \ge y_2(n,k,\mathbf{s}) \ge \dots \ge y_{k-1}(n,k,\mathbf{s})$ (cf. Lemma 7), one can derive that, if $i_1 \in \mathcal{N}_H$, $i_2 \in \mathcal{N}_M$, and $i_3 \in \mathcal{N}_L$, $i_1 < i_2 < i_3$. As $s_1 \ge \dots \ge s_n$, $s_{i_1} \ge s_{i_2} \ge s_{i_3}$. Assuming $s_{i_1} = s_{i_2}$, we have $s_i = s_{i_2}$ for all i with $i_1 \le i \le i_2$. From Lemma 7, we have $y_{i_1}(n,k,\mathbf{s}) = \dots = y_{i_2}(n,k,\mathbf{s})$ that contradicts to $y_{i_1}(n,k,\mathbf{s}) > 0 = y_{i_2}(n,k,\mathbf{s})$. Thus, $s_{i_1} \ne s_{i_2}$. Similarly, one can prove $s_{i_2} \ne s_{i_3}$. As such,

$$s_{i_1} > s_{i_2} > s_{i_3} \quad \forall i_1 \in \mathcal{N}_H, \forall i_2 \in \mathcal{N}_M, \forall i_3 \in \mathcal{N}_L. \quad (85)$$

Given a permutation $\phi \in \Phi$ and an optimal solution $\boldsymbol{\alpha} \in \mathcal{Z}$ (cf. (80)), let us consider $\boldsymbol{\alpha}_\phi = [\alpha_{\phi(1)} \dots \alpha_{\phi(n)}]$. If $i_1 \in \mathcal{N}_H$, by assuming $\phi(i_1) \notin \mathcal{N}_H$, we have $\phi(i_1) \in \mathcal{N}_M \cup \mathcal{N}_L$. As $i_1 \in \mathcal{N}_H$ and $\phi(i_1) \in \mathcal{N}_M \cup \mathcal{N}_L$, from (85), we have $s_{\phi(i_1)} \ne s_{i_1}$, which implies that $i_1 \not\sim \phi(i_1)$. This contradicts to $\phi \in \Phi$. Thus, if $i_1 \in \mathcal{N}_H$, $\phi(i_1) \in \mathcal{N}_H$. Similarly, one can prove that, if $i_2 \in \mathcal{N}_M$ and $i_3 \in \mathcal{N}_L$, $\phi(i_2) \in \mathcal{N}_M$ and $\phi(i_3) \in \mathcal{N}_L$. Note that $k \in \mathcal{N}_L$, which implies $\phi(k) \in \mathcal{N}_L$ and $\alpha_{\phi(k)} = \alpha_k$. Thus, we derive (a) $\alpha_{\phi(i)} = 0$ if $i \in \mathcal{N}_H$; (b) $0 \le \alpha_{\phi(i)} \le \alpha_k = \alpha_{\phi(k)}$ if $i \in \mathcal{N}_M$; (c) $\alpha_i = \alpha_k = \alpha_{\phi(k)}$ if $i \in \mathcal{N}_L$; (d) $\sum_{i\in[k]} \alpha_{\phi(i)} = \sum_{i\in[k]} \alpha_i = 1$. Thus, given a permutation $\phi \in \Phi$ and an optimal solution $\boldsymbol{\alpha} \in \mathcal{Z}$, $\boldsymbol{\alpha}_\phi$ satisfies (81), (82), (83), and (84). From (56), we conclude that any optimal solution to Problem 1 satisfies (81), (82), (83), and (84). Furthermore, by using similar proof to that of Theorem 12, we can show that all solutions satisfying (81), (82), (83), and (84) have the same storage cost (one needs to verify that Eq. (75) to (79) also hold for this case). This completes the proof.

Appendix B
Proof of Theorem 5

We only need to solve Problem 3.B. In order to simplify the notation, we use b_j and γ_j to represent $\bar{t}_{j,i}$ and $\bar{\beta}_{j,i}$, respectively. Thus, Problem 3.B can be written as

Problem 6: Given n, k, and $\{b_j\}_{j\in[n-1]}$ with $b_1 \ge \dots \ge b_{n-1} > 0$,

$$\begin{aligned}
&\min_{\gamma_j\in\mathbb{R}\ \forall j\in[n-1]} \ \max_{j\in[n-1]} b_j\gamma_j \\
&\text{s.t.} \quad \gamma_j \ge 0 \quad \forall j \in [n-1], && (86)\\
&\qquad \sum_{j\in\mathcal{S}} \gamma_j \ge \frac{1}{k} \quad \forall \mathcal{S} \subseteq [n-1] \text{ with } |\mathcal{S}| = n-k. && (87)
\end{aligned}$$

Similar to Lemma 6, we can prove that there exists a special optimal solution to Problem 6 and then find this optimal solution from a subdomain of the feasible domain.

Lemma 8: There exists an optimal solution to Problem 6 satisfying

$$\begin{aligned}
&0 \le \gamma_j \le \frac{1}{k} \quad \forall j \in [n-k], && (88)\\
&\gamma_j = \gamma_{n-k} \quad \forall j \in [n-1] \setminus [n-k], && (89)\\
&\sum_{j=1}^{n-k} \gamma_j = \frac{1}{k}. && (90)
\end{aligned}$$

Proof: Given a variable vector $\boldsymbol{\gamma} \triangleq [\gamma_1 \dots \gamma_{n-1}] \in \mathbb{R}^{n-1}$, the objective function to Problem 6 is $G(\boldsymbol{\gamma}) \triangleq \max_{j\in[n-1]} b_j\gamma_j$. Given an optimal solution $\boldsymbol{\gamma}^* = [\gamma_1^* \dots \gamma_{n-1}^*]$ to Problem 6. If there are $e \in [n-2]$ such that $\gamma_e^* > \gamma_{e+1}^*$, we can obtain $\boldsymbol{\gamma}' \triangleq [\gamma_j']_{j\in[n-1]}$ from $\boldsymbol{\gamma}^*$ by swapping the values of γ_e^* and γ_{e+1}^*, i.e., letting $\gamma_e' = \gamma_{e+1}^*$, $\gamma_{e+1}' = \gamma_e^*$, and $\gamma_j' = \gamma_j^*$ for all $j \in [n-1] \setminus \{e, e+1\}$. Clearly, as $\boldsymbol{\gamma}^*$ is a feasible solution to Problem 6, $\boldsymbol{\gamma}'$ is also a feasible solution. As $b_e \ge b_{e+1}$, we have $b_e\gamma_e' = b_e\gamma_{e+1}^* < b_e\gamma_e^* \le G(\boldsymbol{\gamma}^*)$ and $b_{e+1}\gamma_{e+1}' = b_{e+1}\gamma_e^* \le b_e\gamma_e^* \le G(\boldsymbol{\gamma}^*)$. Furthermore, as $b_j\gamma_e' = b_j\gamma_e^* \le G(\boldsymbol{\gamma}^*)$ for all $j \in [n-1] \setminus \{e, e+1\}$, we have $G(\boldsymbol{\gamma}') \le G(\boldsymbol{\gamma}^*)$, which means $\boldsymbol{\gamma}'$ is also an optimal solution. In conclusion, given an optimal solution $\boldsymbol{\gamma}^*$ with $\gamma_e^* > \gamma_{e+1}^*$ for some $e \in [n-2]$, we can obtain another optimal solution by swapping the values of γ_e^* and γ_{e+1}^*. Thus, by applying the bubble sorting algorithm to the given optimal solution vector, we can obtain a new optimal solution vector with elements in ascending order.

Consider an optimal solution $\boldsymbol{\gamma}^*$ with $\gamma_1^* \le \dots \le \gamma_{n-1}^*$. Let $\alpha \triangleq \sum_{j=1}^{n-k} \gamma_j^*$. From (87), $\alpha \ge \frac{1}{k}$. Let $\boldsymbol{\gamma}'' = \frac{1}{k\alpha}\boldsymbol{\gamma}^*$. Clearly, $\boldsymbol{\gamma}'' = [\gamma_1'' \dots \gamma_{n-1}'']$ is a feasible solution with $G(\boldsymbol{\gamma}'') = \frac{1}{k\alpha}G(\boldsymbol{\gamma}^*)$. Since $\boldsymbol{\gamma}^*$ is an optimal solution, we have $\frac{1}{k\alpha}G(\boldsymbol{\gamma}^*) \ge G(\boldsymbol{\gamma}^*)$, which, together with $\alpha \ge \frac{1}{k}$ and

$G(\boldsymbol{\gamma}^*) > 0$, implies $\alpha = \frac{1}{k}$. Thus, $\sum_{j=1}^{n-k} \gamma_j^* = \alpha = 1/k$. Let $\boldsymbol{\gamma}''' = [\gamma_1''' \dots \gamma_{n-1}''']$, where $\gamma_j''' = \gamma_j^*$ for all $j \in [n-k]$ and $\gamma_j''' = \gamma_{n-k}^*$ for all $j \in [n-1] \setminus [n-k]$. Clearly, $\boldsymbol{\gamma}'''$ is a feasible solution. Since $\gamma_{n-k}^* \leq \gamma_j^*$ for all $j \in [n-1] \setminus [n-k]$, we have $\gamma_j''' \leq \gamma_j^*$ for all $j \in [n-1]$, which implies $G(\boldsymbol{\gamma}''') \leq G(\boldsymbol{\gamma}^*)$. Thus, $\boldsymbol{\gamma}'''$ is an optimal solution with $\sum_{i=1}^{n-k} \gamma_j''' = \sum_{i=1}^{n-k} \gamma_j^* = \frac{1}{k}$ and $\gamma_j''' = \gamma_{n-k}^* = \gamma_{n-k}'''$ for all $j \in [n-1] \setminus [n-k]$. From $\gamma_j''' \geq 0$ for all $j \in [n-1]$ and $\sum_{j=1}^{n-k} \gamma_j''' = \frac{1}{k}$, we have $0 \leq \gamma_j''' \leq \frac{1}{k}$ for all $j \in [n-k]$. In conclusion, given any optimal solution $\boldsymbol{\gamma}^*$ with $\gamma_1^* \leq \dots \leq \gamma_{n-1}^*$, we can obtain an optimal solution γ_j''' which satisfies (88), (89), and (90). ∎

If a variable vector $\boldsymbol{\gamma} \in \mathbb{R}^{n-1}$ satisfies (88), (89) and (90), $\boldsymbol{\gamma}$ satisfies (86) and (87), which means the conditions (88), (89) and (90) characterize a subdomain of the feasible domain to Problem 6. Lemma 8 implies that Problem 6 has an optimal solution in this subdomain, which means we can minimize the objective function in this subdomain (cf. Problem 7) and get the same optimal value. In this subdomain, as $b_1 \geq \dots \geq b_{n-1} > 0$ and $\gamma_j = \gamma_{n-k}$ for all $j \in [n-1] \setminus [n-k]$, the original objective function can be written as $\max_{j \in [n-k]} b_j \gamma_j$, which leads us to the following problem.

Problem 7: Given n, k, and $\{b_j\}_{j \in [n-1]}$ with $b_1 \geq \dots \geq b_{n-1} > 0$,

$$\begin{aligned} \min_{\gamma_j \in \mathbb{R}, \forall j \in [n-k]} \quad & \max_{j \in [n-k]} b_j \gamma_j \\ \text{s.t.} \quad & 0 \leq \gamma_j \leq \frac{1}{k} \quad \forall j \in [n-k], & (91) \\ & \sum_{j \in [n-k]} \gamma_j = \frac{1}{k}. & (92) \end{aligned}$$

Clearly, the optimal value to Problem 7 is the optimal value to Problem 6. Equivalently, Problem 7 can be converted to a standard linear programming problem as follows:

$$\begin{aligned} \min_{\gamma_j \in \mathbb{R}, \forall j \in [n-k]} \quad & x \\ \text{s.t.} \quad & b_j \gamma_j \leq x \quad \forall j \in [n-k], \\ & 0 \leq \gamma_j \leq \frac{1}{k} \quad \forall j \in [n-k], & (93) \\ & \sum_{j \in [n-k]} \gamma_j = \frac{1}{k}. & (94) \end{aligned}$$

The associated Lagrangian is

$$\begin{aligned} L = & x + \sum_{j=1}^{n-k} \lambda_j (b_j \gamma_j - x) + \sum_{j=1}^{n-k} \nu_j (0 - \gamma_j) \\ & + \sum_{j=1}^{n-k} \mu_j (\gamma_j - \frac{1}{k}) + u (\sum_{j=1}^{n-k} \gamma_j - \frac{1}{k}). \end{aligned}$$

Accordingly, the KKT conditions, besides the original constraints, are

$$\lambda_j, \nu_j, \mu_j \geq 0 \quad \forall j \in [n-k], \tag{95}$$
$$\lambda_j (b_j \gamma_j - x) = 0 \quad \forall j \in [n-k], \tag{96}$$
$$\nu_j \gamma_j = \mu_j (\gamma_j - \frac{1}{k}) = 0 \quad \forall j \in [n-k], \tag{97}$$
$$1 - \sum_{j \in [n-k]} \lambda_j = 0, \tag{98}$$
$$\lambda_j b_j - \nu_j + \mu_j + u = 0. \tag{99}$$

Given an optimal solution $[\gamma_1 \dots \gamma_{n-k}]$, there exist x, u, and $\{\lambda_i, \nu_i, \mu_i\}_{i \in [n-k]}$ that, together with the optimal solution, satisfy the KKT condition and x is the optimal value. Clearly, $x > 0$. From (95) and (98), we know that there exists $j' \in [n-k]$ such that $\lambda_{j'} > 0$. From (96), we have $b_{j'} \gamma_{j'} - x = 0$. Since $b_{j'} > 0$ and $x > 0$, we obtain

$$\gamma_{j'} > 0, \tag{100}$$

which, together with (95) and (97), implies $\nu_{j'} = 0$ and $\mu_{j'} \geq 0$. From (99), we have $\lambda_{j'} b_{j'} - \nu_{j'} + \mu_{j'} + u = 0$, which, together with $\lambda_{j'} > 0$, $b_{j'} > 0$, $\nu_{j'} = 0$, and $\mu_{j'} \geq 0$, leads to $u < 0$. Let us assume there is $j'' \in [n-k]$ with $\lambda_{j''} = 0$. From (99), we can derive that $\nu_{j''} - \mu_{j''} = u < 0$, which, together with (95), implies that $\mu_{j''} > 0$. From (97), we have

$$\gamma_{j''} = \frac{1}{k}. \tag{101}$$

From (93), (100), and (101), we have $\sum_{j \in [n-k]} \gamma_j \geq \gamma_{j''} + \gamma_{j'} > \frac{1}{k}$, which contradicts to (94). Thus, we have $\lambda_j > 0$ for all $j \in [n-k]$. From (96), we have

$$x = b_j \gamma_j \quad \forall j \in [n-k]. \tag{102}$$

From (94) and (102), we have $x = \frac{1}{k}(b_1^{-1} + \dots + b_{n-k}^{-1})^{-1}$ and $\gamma_j = \frac{x}{b_j}$. Since an optimal solution exists, this must be an optimal solution to Problem 7 and the optimal value is $G \triangleq \frac{1}{k}(b_1^{-1} + \dots + b_{n-k}^{-1})^{-1}$, which is also the optimal value to Problem 6.

Next, we fully characterize the optimal solutions to Problem 6. Consider an optimal solution $[\gamma_1^* \dots \gamma_{n-1}^*]$. Since $G = \frac{1}{k}(b_1^{-1} + \dots + b_{n-k}^{-1})^{-1}$ is the optimal value, we derive $\gamma_j^* \leq G/b_j$ for all $j \in [n-1]$. If $\gamma_j^* < G/b_j$ for some $j \in [n-k]$, we have $\sum_{j \in [n-k]} \gamma_j^* < \frac{1}{k}$, which is a contradiction. Thus, $\gamma_j^* = G/b_j$ for all $j \in [n-k]$. Furthermore, if $\gamma_t^* < \gamma_{n-k}^*$ for some $t \in [n-1] \setminus [n-k]$, we have $\gamma_t^* + \sum_{j \in [n-k-1]} \gamma_j^* < \frac{1}{k}$, which is a contradiction. Thus, we have $\gamma_{n-k}^* \leq \gamma_j^* \leq G/b_j$ for all $j \in [n-1] \setminus [n-k]$. This completes the proof.

Appendix C
Proof of Lemma 1

All we need to prove is that (37) is equivalent to conditions (39), (40), and (41). First, we prove (39), (40), and (41) lead to (37). For any $i \in [k-1]$, as $|[n] \setminus \{f_1 \dots f_i\}| \geq n-k+1$, there exists $\mathcal{S} \subseteq [n] \setminus \{f_1 \dots f_i\}$ such that $\mathcal{S} \subseteq [n] \setminus \{1, f_i\}$ and $|\mathcal{S}| = n-k$. From (35), (39) and (40), we have

$\sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i} \geq \sum_{j\in\mathcal{S}}\beta_{j,f_i} \geq \alpha_{f_i}$. Thus, for all $i\in[k-1]$, we have

$$\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} = \alpha_{f_i}. \tag{103}$$

If $f_k = 1$, from (39), we have $\sum_{j\in[n]\setminus\{f_1,\ldots,f_k\}}\beta_{j,f_k} \geq \alpha_{f_k}$, which, together with (103), leads to

$$\sum_{i=1}^{k}\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} = \sum_{i=1}^{k}\alpha_{f_i} \geq 1.$$

If $f_k \neq 1$ and $1 \in \{f_1,\ldots,f_{k-1}\}$, we have $[n]\setminus\{f_1,\ldots,f_k\} \subseteq [n]\setminus\{1,f_k\}$. From (40) and (35), $\sum_{j\in[n]\setminus\{f_1,\ldots,f_k\}}\beta_{j,f_k} \geq \alpha_2 = \alpha_{f_k}$, which, together with (103), leads to (37). If $f_k \neq 1$ and $1 \notin \{f_1,\ldots,f_{k-1}\}$, from (41), we have $\sum_{j\in[n]\setminus\{f_1,\ldots,f_k\}}\beta_{j,f_k} \geq \alpha_1$. Since $1\notin\{f_1,\ldots,f_{k-1}\}$, (103) implies $\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} = \alpha_2$. Thus, from (36), we have $\sum_{i=1}^{k}\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} = (k-1)\alpha_2 + \alpha_1 = 1$. Thus, for all $\mathbf{f}\in\mathcal{F}$, we have proven $\sum_{i=1}^{k}\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} \geq 1$, which means (37) holds.

Conversely, given (37), we respectively prove (39), (40), and (41) by contradictions.

1) Assume there exists $\mathcal{S}'\subseteq[n]\setminus\{1\}$ with $|\mathcal{S}'| = n-k$ such that $\sum_{j\in\mathcal{S}'}\beta_{j,1} < \alpha_1$. Consider $\mathbf{f}\in\mathcal{F}$ such that $f_i\in[n]\setminus\mathcal{S}'$ for all $i\in[k]$ and $f_k = 1$. Note that $f_i \neq 1$ for all $i\in[k-1]$ that means $\alpha_{f_i} = \alpha_2$ for all $i\in[k-1]$. Then, we have $\sum_{i=1}^{k}\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} \leq \sum_{i=1}^{k-1}\alpha_{f_i} + \sum_{j\in[n]\setminus\{f_1,\ldots,f_k\}}\beta_{j,f_k} = (k-1)\alpha_2 + \sum_{j\in\mathcal{S}'}\beta_{j,1} < (k-1)\alpha_2+\alpha_1 = 1$, which contradicts to (37). Thus, (39) holds.
2) Assume there exist $i'\in[n]\setminus\{1\}$ and $\mathcal{S}'\subseteq[n]\setminus\{1,i'\}$ with $|\mathcal{S}'| = n-k$ such that $\sum_{j\in\mathcal{S}'}\beta_{j,i'} < \alpha_2$. Consider $\mathbf{f}\in\mathcal{F}$ such that $f_i\in[n]\setminus\mathcal{S}'$ for all $i\in[k]$ and $f_k = i'$. As $1\in[n]\setminus\mathcal{S}'$ and $f_i\in[n]\setminus\mathcal{S}'$ for all $i\in[k]$, there exists $t\in[k-1]$ such that $\alpha_{f_t} = \alpha_1$. Then, we have $\sum_{i=1}^{k}\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} \leq \sum_{i=1}^{k-1}\alpha_{f_i} + \sum_{j\in[n]\setminus\{f_1,\ldots,f_k\}}\beta_{j,f_k} = (k-2)\alpha_2+\alpha_1+\sum_{j\in\mathcal{S}'}\beta_{j,i'} < (k-2)\alpha_2+\alpha_1+\alpha_2 = 1$, which contradicts to (37). Thus, (40) holds.
3) Assume there exist $i'\in[n]\setminus\{1\}$ and $\mathcal{S}'\subseteq[n]\setminus\{1,i'\}$ with $|\mathcal{S}'| = n-k-1$ such that $\beta_{1,i'} + \sum_{j\in\mathcal{S}'}\beta_{j,i'} < \alpha_1$. Consider $\mathbf{f}\in\mathcal{F}$ such that $f_i\in[n]\setminus(\mathcal{S}'\cup\{1\})$ for all $i\in[k]$ and $f_k = i'$. Note that $f_i\neq 1$ for all $i\in[k-1]$ that means $\alpha_{f_i} = \alpha_2$ for all $i\in[k-1]$. Then, we have $\sum_{i=1}^{k}\min\{\alpha_{f_i}, \sum_{j\in[n]\setminus\{f_1,\ldots,f_i\}}\beta_{j,f_i}\} \leq \sum_{i=1}^{k-1}\alpha_{f_i} + \sum_{j\in[n]\setminus\{f_1,\ldots,f_k\}}\beta_{j,f_k} = (k-1)\alpha_2+\sum_{j\in\mathcal{S}'\cup\{1\}}\beta_{j,i'} < (k-1)\alpha_2+\alpha_1 = 1$, which contradicts to (37). Thus, (41) holds.

Appendix D
Proof of Theorem 8

We first present a helpful lemma from [27] for clarity.

Lemma 9: [Lemma 3 in [27]] Given a set $\mathcal{S}$ with $|\mathcal{S}| = m$, if $\sum_{s\in\mathcal{S}'} s \geq c$ for all $\mathcal{S}'\subset\mathcal{S}$ with $|\mathcal{S}'| = m' < m$, we have $\sum_{s\in\mathcal{S}} s \geq \frac{mc}{m'}$ and the equality holds if and only if $s = \frac{c}{m'}$ for all $s\in\mathcal{S}$.

The following two lemmas show the optimal $C_{R_i}^{tot}(\boldsymbol{\beta}_i)$ for all $i\in[n]$.

Lemma 10: For an (n,k) two-valued array code with $\boldsymbol{\alpha}$, we have $C_{R_1}^{tot}(\boldsymbol{\beta}_1) \geq \frac{(n-1)\alpha_1}{n-k}$, and the equality holds if and only if

$$\beta_{j,1} = \frac{\alpha_1}{n-k} \quad \forall j\in[n]\setminus\{1\}. \tag{104}$$

Proof: From Lemma 1, the constraints on $\boldsymbol{\beta}_1$ are $\beta_{j,1}\geq 0$ for all $j\in[n]\setminus\{1\}$ and $\sum_{j\in\mathcal{S}}\beta_{j,1}\geq\alpha_1$ for all $\mathcal{S}\subseteq[n]\setminus\{1\}$ with $|\mathcal{S}| = n-k$. Then, we can minimize $\sum_{j\in[n]\setminus\{1\}}\beta_{j,1}$ subject to these constraints. Note that this problem has the same form as Problem 2.B, whose optimal solution has been given. However, it is tedious to discuss the whole optimal solution set. Alternatively, this lemma can be directly given by Lemma 9. ■

Lemma 11: Given any (n,k) two-valued array code and $i\in[n]\setminus\{1\}$, we have the following results:

1) If $\alpha_1 \leq \frac{(n-k-1)\alpha_2}{n-k}$, we have $C_R^{tot}(\boldsymbol{\beta}_i) \geq \frac{(n-2)\alpha_2}{n-k}$. Furthermore, if $k\geq 3$, the equality holds if and only if $\beta_{1,i} = 0$ and $\beta_{j,i} = \frac{\alpha_2}{n-k}$ for all $j\in[n]\setminus\{1,i\}$.
2) If $\alpha_1 > \frac{(n-k-1)\alpha_2}{n-k}$, we have $C_{R_i}^{tot}(\boldsymbol{\beta}_i) \geq \alpha_1 + \frac{(k-1)\alpha_2}{n-k}$, and the equality holds if and only if $\beta_{1,i} = \alpha_1 - \frac{(n-k-1)\alpha_2}{n-k}$ and $\beta_{j,i} = \frac{\alpha_2}{n-k}$ for all $j\in[n]\setminus\{1,i\}$.

Proof: Given $i\in[n]\setminus\{1\}$, from Lemma 1, the constraints on $\boldsymbol{\beta}_i$ are $\beta_{j,i}\geq 0$ for all $j\in[n]\setminus\{i\}$, $\sum_{j\in\mathcal{S}}\beta_{j,i}\geq\alpha_2$ for all $\mathcal{S}\subseteq[n]\setminus\{1,i\}$ with $|\mathcal{S}| = n-k$, and $\beta_{1,i}+\sum_{j\in\mathcal{S}}\beta_{j,i}\geq\alpha_1$ for all $\mathcal{S}\subseteq[n]\setminus\{1,i\}$ with $|\mathcal{S}| = n-k-1$. Note that an (n,k) two-valued array code has $k\geq 2$. From $k\geq 2$ and Lemma 9, we have $\sum_{j\in[n]\setminus\{1,i\}}\beta_{j,i} \geq \frac{(n-2)\alpha_2}{n-k}$. When $k\geq 3$, the equality holds if and only if $\beta_{j,i} = \frac{\alpha_2}{n-k}$ for all $j\in[n]\setminus\{1,i\}$. Thus, we have

$$\begin{aligned} C_{R_i}^{tot}(\boldsymbol{\beta}_i) &= \beta_{1,i} + \sum_{j\in[n]\setminus\{1,i\}}\beta_{j,i} \\ &\geq \beta_{1,i} + \frac{(n-2)\alpha_2}{n-k} \geq \frac{(n-2)\alpha_2}{n-k}. \end{aligned} \tag{105}$$

When $k\geq 3$, the equality holds if and only if $\beta_{1,i} = 0$ and $\beta_{j,i} = \frac{\alpha_2}{n-k}$ for all $j\in[n]\setminus\{1,i\}$.

Given $i\in[n]\setminus\{1\}$, we can create a sequence $q_{1,i}\leq\cdots\leq q_{n-2,i}$ by arranging the elements in $\{\beta_{j,i}\}_{j\in[n]\setminus\{1,i\}}$ in ascending order. Since $\sum_{j\in\mathcal{S}}\beta_{j,i}\geq\alpha_2$ for all $\mathcal{S}\subseteq[n]\setminus\{1,i\}$ with $|\mathcal{S}| = n-k$, we have $\sum_{t=1}^{n-k}q_{t,i}\geq\alpha_2$, which implies $q_{n-k,i}\geq\frac{\alpha_2}{n-k}$ and the equality holds if and only if $q_{1,i} = \cdots = q_{n-k,i} = \frac{\alpha_2}{n-k}$. Also, as $\beta_{1,i}+\sum_{j\in\mathcal{S}}\beta_{j,i}\geq\alpha_1$ for all $\mathcal{S}\subseteq[n]\setminus\{1,i\}$ with $|\mathcal{S}| = n-k-1$, we have $\beta_{1,i}+\sum_{t=1}^{n-k-1}q_{t,i}\geq\alpha_1$. Thus, we obtain

$$\begin{aligned} C_{R_i}^{tot}(\boldsymbol{\beta}_i) &= \beta_{1,i} + \sum_{j\in[n]\setminus\{1,i\}}\beta_{j,i} = \beta_{1,i} + \sum_{t=1}^{n-2}q_{t,i} \\ &= \beta_{1,i} + \sum_{t=1}^{n-k-1}q_{t,i} + \sum_{t=n-k}^{n-2}q_{t,i} \\ &\geq \alpha_1 + \frac{(k-1)\alpha_2}{n-k}, \end{aligned} \tag{106}$$

and the equality holds if and only if $\beta_{j,i} = \frac{\alpha_2}{n-k}$ for all $j \in [n]\backslash\{1,i\}$ and $\beta_{1,i} = \alpha_1 - \frac{(n-k-1)\alpha_2}{n-k}$. (105) and (106) present two lower bounds of $C_{R_i}^{tot}(\boldsymbol{\beta}_i)$. If $\alpha_1 \le \frac{(n-k-1)\alpha_2}{n-k}$, $\boldsymbol{\beta}'_i$, with $\beta'_{1,i} = 0$ and $\beta'_{j,i} = \frac{\alpha_2}{n-k}$ for all $j \in [n] \setminus \{1,i\}$, satisfies all constraints on $\boldsymbol{\beta}_i$. Since $C_R^{tot}(\boldsymbol{\beta}'_i) = \frac{(n-2)\alpha_2}{n-k}$, the lower bound in (105) can be achieved if $\alpha_1 \le \frac{(n-k-1)\alpha_2}{n-k}$. If $\alpha_1 > \frac{(n-k-1)\alpha_2}{n-k}$, $\boldsymbol{\beta}''_i$, with $\beta''_{1,i} = \alpha_1 - \frac{(n-k-1)\alpha_2}{n-k}$ and $\beta''_{j,i} = \frac{\alpha_2}{n-k}$ for all $j \in [n] \setminus \{1,i\}$, satisfies all constraints on $\boldsymbol{\beta}$. Since $C_R^{tot}(\boldsymbol{\beta}''_i) = \alpha_1 + \frac{(k-1)\alpha_2}{n-k}$, the lower bound in (106) can be achieved if $\alpha_1 > \frac{(n-k-1)\alpha_2}{n-k}$. ∎

From Lemmas 10 and 11, we can directly obtain that

1) If $\alpha_1 \le \frac{(n-k-1)\alpha_2}{n-k}$, we have $nC_R^{ave}(\boldsymbol{\beta}) \ge \frac{(n-1)\alpha_1}{n-k} + \frac{(n-1)(n-2)\alpha_2}{n-k}$. Furthermore, if $k \ge 3$, the equality holds if and only if $\boldsymbol{\beta}$ satisfies (104) and

$$\beta_{1,i} = 0, \beta_{j,i} = \frac{\alpha_2}{n-k} \quad \forall i \in [n]\backslash\{1\}, \forall j \in [n]\backslash\{1,i\}. \tag{107}$$

2) If $\alpha_1 > \frac{(n-k-1)\alpha_2}{n-k}$, we have $nC_R^{ave}(\boldsymbol{\beta}) \ge \frac{(n-1)\alpha_1}{n-k} + (n-1)(\alpha_1 + \frac{(k-1)\alpha_2}{n-k})$, and the equality holds if and only if $\boldsymbol{\beta}$ satisfies (104) and

$$\begin{aligned} \beta_{1,i} &= \alpha_1 - \frac{(n-k-1)\alpha_2}{n-k} \\ \beta_{j,i} &= \frac{\alpha_2}{n-k} \quad \forall i \in [n] \setminus \{1\}, \forall j \in [n] \setminus \{1,i\}. \end{aligned} \tag{108}$$

Furthermore, since $\alpha_2 = \frac{1-\alpha_1}{k-1}$, we can obtain $\frac{(n-1)\alpha_1}{n-k} + \frac{(n-1)(n-2)\alpha_2}{n-k} = (\frac{n-1}{n-k} - \frac{(n-1)(n-2)}{(n-k)(k-1)})\alpha_1 + \frac{(n-1)(n-2)}{(n-k)(k-1)}$ and $\frac{(n-1)\alpha_1}{n-k} + (n-1)(\alpha_1 + \frac{(k-1)\alpha_2}{n-k}) = (n-1)\alpha_1 + \frac{n-1}{n-k}$, which completes the proof.

Appendix E
Proof of Lemma 5

Let $W_i(\alpha_1) \triangleq \frac{1-\alpha_1}{k-1}/(\frac{1}{t_{\tau_i(1),i}} + \cdots + \frac{1}{t_{\tau_i(n-k),i}})$ and $Q_i(\alpha_1) \triangleq \alpha_1/(\frac{1}{t_{1,i}} + \frac{1}{t_{\tau_i(1),i}} + \cdots + \frac{1}{t_{\tau_i(n-k-1),i}})$. Thus, $J_i(\alpha_1) = W_i(\alpha_1)$ if $\alpha_1 \le \alpha^*_{1,i}$ and $J_i(\alpha_1) = Q_i(\alpha_1)$ if $\alpha_1 > \alpha^*_{1,i}$. Given any $\boldsymbol{\alpha}$ for a two-valued array code, we have $\alpha_2 = \frac{1-\alpha_1}{k-1}$, which implies that $W_i(\alpha_1) = \alpha_2/(\frac{1}{t_{\tau_i(1),i}} + \cdots + \frac{1}{t_{\tau_i(n-k),i}})$.

Consider the case that the given $\boldsymbol{\alpha}$ satisfies $\alpha_1 \le \alpha^*_{1,i}$. We have $\alpha_2 \frac{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i} + t^{-1}_{1,i}}{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i}} \ge \alpha_1$. From Lemma 1, for an (n,k) irregular array code with the given $\boldsymbol{\alpha}$ and all $i \in [n] \setminus \{1\}$, we have,

$$\beta_{\tau_i(j),i} \ge 0 \quad \forall j \in [n-2], \tag{109}$$

$$\sum_{j \in \mathcal{S}} \beta_{\tau_i(j),i} \ge \alpha_2 \quad \forall \mathcal{S} \subseteq [n-2] \text{ with } |\mathcal{S}| = n-k. \tag{110}$$

The minimization of $\max_{j \in [n]\backslash\{1,i\}} t_{j,i}\beta_{j,i} = \max_{j \in [n-2]} t_{\tau_i(j),i}\beta_{\tau_i(j),i}$ subject to (109) and (110) has the same problem form as Problem 3.B. From Theorem 5, which gives the optimal value of Problem 3.B, we have $\max_{j \in [n]\backslash\{1,i\}} t_{j,i}\beta_{j,i} \ge \alpha_2/(\frac{1}{t_{\tau_i(1),i}} + \cdots + \frac{1}{t_{\tau_i(n-k),i}}) = W_i(\alpha_1)$ and the equality holds if

$$\beta_{j,i} = \begin{cases} W_i(\alpha_1)/t_{j,i} & \forall j \text{ with } \tau_i^{-1}(j) \in [n-k], \\ \beta_{\tau_i(n-k),i} & \forall j \text{ with } \tau_i^{-1}(j) \in [n-2] \setminus [n-k]. \end{cases} \tag{111}$$

Thus, $C_{R_i}^{wor}(\boldsymbol{\beta}_i) = \max_{j \in [n]\backslash\{i\}} t_{j,i}\beta_{j,i} \ge \max_{j \in [n]\backslash\{1,i\}} t_{j,i}\beta_{j,i} \ge W_i(\alpha_1)$ and the equality holds if

$$\beta_{j,i} = \begin{cases} J_i(\alpha_1)/t_{j,i} & \forall j \text{ with } \tau_i^{-1}(j) \in [n-k], \\ \beta_{\tau_i(n-k),i} & \forall j \text{ with } \tau_i^{-1}(j) \in [n-2] \setminus [n-k], \\ J_i(\alpha_1)/t_{1,i} & j = 1. \end{cases} \tag{112}$$

Next, we show that there exists an (n,k) two-valued array codes with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfying (112). From Lemma 1, we only need to show that for all $i \in [n] \setminus \{1\}$, $\boldsymbol{\beta}_i$ satisfying (112) satisfies (38), (40), and (41). Given any $i \in [n] \setminus \{1\}$, consider $\boldsymbol{\beta}_i$ satisfying (112). Clearly, it satisfies (38) and (40). We still need to check if (41) holds. Since $t_{\tau_i(1),i} \ge \cdots \ge t_{\tau_i(n-2),i}$, we have $\beta_{\tau_i(1),i} \le \cdots \le \beta_{\tau_i(n-2),i}$. For all $\mathcal{S} \subseteq [n] \setminus \{1,i\}$ with $|\mathcal{S}| = n-k-1$, as $\alpha_2 \frac{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i} + t^{-1}_{1,i}}{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i}} \ge \alpha_1$, we have $\beta_{1,i} + \sum_{j \in \mathcal{S}} \beta_{j,i} \ge \beta_{1,i} + \sum_{j=1}^{n-k-1} \beta_{\tau_i(j),i} = W_i(\alpha_1)(t^{-1}_{1,i} + t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i}) = \alpha_2 \frac{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i} + t^{-1}_{1,i}}{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i}} \ge \alpha_1$. This completes the proof for this case.

Consider the case that given $\boldsymbol{\alpha}$ satisfies $\alpha_2 \frac{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i} + t^{-1}_{1,i}}{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i}} < \alpha_1$. First, we show that there exists an (n,k) two-valued array codes with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfying (112). From Lemma 1, we only need to show that for all $i \in [n] \setminus \{1\}$, $\boldsymbol{\beta}_i$ satisfying (112) satisfies (38), (40), and (41). Consider $\boldsymbol{\beta}_i$ satisfying (112). Clearly, $\boldsymbol{\beta}_i$ satisfies (38). For all $\mathcal{S} \subseteq [n] \setminus \{1,i\}$ with $|\mathcal{S}| = n-k-1$, as $\beta_{\tau_i(1),i} \le \cdots \le \beta_{\tau_i(n-2),i}$, we have $\beta_{1,i} + \sum_{j \in \mathcal{S}} \beta_{j,i} \ge \beta_{1,i} + \sum_{j=1}^{n-k-1} \beta_{\tau_i(j),i} = J_i(\alpha_1)(t^{-1}_{1,i} + t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i}) = Q_i(\alpha_1)(t^{-1}_{1,i} + t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i}) = \alpha_1$, which means $\boldsymbol{\beta}_i$ satisfies (41). Similarly, for all $\mathcal{S} \subseteq [n] \setminus \{1,i\}$ with $|\mathcal{S}| = n-k$, we derive $\sum_{j \in \mathcal{S}} \beta_{j,i} \ge \sum_{j=1}^{n-k} \beta_{\tau_i(j),i} = Q_i(\alpha_1)(t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i})$. As $\alpha_2 \frac{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i} + t^{-1}_{1,i}}{t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i}} < \alpha_1$, we have $Q_i(\alpha_1) > W_i(\alpha_1)$. Thus, for all $\mathcal{S} \subseteq [n] \setminus \{1,i\}$ with $|\mathcal{S}| = n-k$, we have $\sum_{j \in \mathcal{S}} \beta_{j,i} > W_i(\alpha_1)(t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k),i}) = \alpha_2$, which means $\boldsymbol{\beta}_i$ satisfies (40). To conclude, there exists an (n,k) two-valued array codes with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfying (112), which means the optimal worst-case repair cost of node i is at most $Q_i(\alpha_1)$. Assume $Q_i(\alpha_1)$ is not the optimal worst-case repair cost of node i. There exists an (n,k) two-valued array code with given $\boldsymbol{\alpha}$ and $C_{R_i}^{wor} \triangleq Q_i(\alpha_1)' < Q_i(\alpha_1)$. Let $\boldsymbol{\beta}'_i \triangleq [\beta'_{j,i}]_{j \in [n]\backslash\{i\}}$ be the $\boldsymbol{\beta}_i$ of this code. Thus, $t_{1,i}\beta'_{1,i} \le Q_i(\alpha_1)'$, which leads to $\beta'_{1,i} \le Q_i(\alpha_1)'/t_{1,i}$. Furthermore, from (41), we have $\sum_{j \in \mathcal{S}} \beta'_{j,i} \ge \alpha_1 - Q_i(\alpha_1)'/t_{1,i}$ for all $\mathcal{S} \subseteq [n] \setminus \{1,i\}$ with $|\mathcal{S}| = n-k-1$. From $Q_i(\alpha_1) = \alpha_1/(t^{-1}_{1,i} + t^{-1}_{\tau_i(1),i} +$

$\cdots + t^{-1}_{\tau_i(n-k-1),i})$ and $Q_i(\alpha_1)' < Q_i(\alpha_1)$, we have $\alpha_1 - Q_i(\alpha_1)'/t_{1,i} > \alpha_1 - Q_i(\alpha_1)/t_{1,i} > 0$. Now we have $\sum_{j\in\mathcal{S}} \beta'_{j,i} \geq \alpha_1 - Q_i(\alpha_1)'/t_{1,i} > 0$ for all $\mathcal{S} \subseteq [n]\setminus\{1,i\}$ with $|\mathcal{S}| = n-k-1$ and $\beta'_{j,i} \geq 0$ for all $j \in [n]\setminus\{i,1\}$. Furthermore, from Problem 3.B and Theorem 5, we can obtain $C^{wor}_{R_i}(\boldsymbol{\beta}'_i) \geq \max_{j\in[n]\setminus\{i,1\}} t_{j,i}\beta'_{j,i} \geq (\alpha_1 - Q_i(\alpha_1)'/t_{1,i})/(t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i}) > (\alpha_1 - Q_i(\alpha_1)/t_{1,i})/(t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i}) = \alpha_1/(t^{-1}_{1,i} + t^{-1}_{\tau_i(1),i} + \cdots + t^{-1}_{\tau_i(n-k-1),i}) = Q_i(\alpha_1) > Q_i(\alpha_1)'$, which is a contradiction. This completes the proof.

REFERENCES

[1] Z. Li, W. H. Mow, L. Deng, and T.-Y. Wu, "Optimal-repair-cost MDS array codes for a class of heterogeneous distributed storage systems," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2022, pp. 2379–2384.

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

[3] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on information theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[4] C. Suh and K. Ramchandran, "Exact-repair MDS code construction using interference alignment," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1425–1442, 2011.

[5] 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 Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, 2011.

[6] M. Ye and A. Barg, "Explicit constructions of high-rate MDS array codes with optimal repair bandwidth," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2001–2014, 2017.

[7] ——, "Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization," *IEEE Transactions on Information Theory*, vol. 63, no. 10, pp. 6307–6317, 2017.

[8] I. Tamo, M. Ye, and A. Barg, "The repair problem for Reed–Solomon codes: Optimal repair of single and multiple erasures with almost optimal node size," *IEEE Transactions on Information Theory*, vol. 65, no. 5, pp. 2673–2695, 2018.

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

[10] S. Mohajer and R. Tandon, "New bounds on the (n, k, d) storage systems with exact repair," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 2056–2060.

[11] S. Akhlaghi, A. Kiani, and M. R. Ghanavati, "Cost-bandwidth tradeoff in distributed storage systems," *Computer Communications*, vol. 33, no. 17, pp. 2105–2115, 2010.

[12] T. Ernvall, S. E. Rouayheb, C. Hollanti, and H. V. Poor, "Capacity and security of heterogeneous distributed storage systems," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2013, pp. 1247–1251.

[13] Q. Yu, K. W. Shum, and C. W. Sung, "Minimization of storage cost in distributed storage systems with repair consideration," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM)*, 2011, pp. 1–5.

[14] N. B. Shah, K. V. Rashmi, and P. V. Kumar, "A flexible class of regenerating codes for distributed storage," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2010, pp. 1943–1947.

[15] Q. Yu, C. W. Sung, and T. H. Chan, "Irregular fractional repetition code optimization for heterogeneous cloud storage," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 1048–1060, 2014.

[16] Q. Yu, K. W. Shum, and C. W. Sung, "Tradeoff between storage cost and repair cost in heterogeneous distributed storage systems," *Transactions on Emerging Telecommunications Technologies*, vol. 26, no. 10, pp. 1201–1211, 2015.

[17] P. Liu, L. Zheng, Q. Yu, and H. Ye, "Tradeoff between storage cost and repair cost for cloud storage," in *Proceedings of IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, 2018, pp. 169–173.

[18] K. G. Benerjee and M. K. Gupta, "Trade-off for heterogeneous distributed storage systems between storage and repair cost," *Problems of Information Transmission*, vol. 57, no. 1, pp. 33–53, 2021.

[19] Y. Wang, D. Wei, X. Yin, and X. Wang, "Heterogeneity-aware data regeneration in distributed storage systems," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2014, pp. 1878–1886.

[20] Y. Hu, P. P. Lee, and X. Zhang, "Double regenerating codes for hierarchical data centers," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2016, pp. 245–249.

[21] Z. Chen and A. Barg, "Explicit constructions of MSR codes for clustered distributed storage: The rack-aware storage model," *IEEE Transactions on Information Theory*, vol. 66, no. 2, pp. 886–899, 2020.

[22] A. Patra and A. Barg, "Regenerating codes on graphs," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2021, pp. 2197–2202.

[23] F. Tosato and M. Sandell, "Irregular MDS array codes," *IEEE Transactions on Information Theory*, vol. 60, no. 9, pp. 5304–5314, 2014.

[24] M. Gerami, M. Xiao, and M. Skoglund, "Optimal-cost repair in multi-hop distributed storage systems," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2011, pp. 1437–1441.

[25] M. Naor and R. M. Roth, "Optimal file sharing in distributed networks," *SIAM Journal on Computing*, vol. 24, no. 1, pp. 158–183, 1995.

[26] P. R. Halmos, *Measure theory*. Springer, 2013, vol. 18.

[27] Z. Li, Y. S. Han, T.-Y. Wu, H. Hou, B. Bai, and G. Zhang, "On the repair bandwidth and repair access of two storage systems: Large-scale and uniform rack-aware storage systems," in *Proceedings of IEEE Information Theory Workshop (ITW)*, 2021, pp. 1–6.

[28] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

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 in 2018 and 2021, respectively. He is now a PhD candidate at the Hong Kong University of Science and Technology. His research interests include coding theory and its applications.

Wai Ho Mow received the Ph.D. degree in information engineering from the Chinese University of Hong Kong in 1993. From 1997 to 1999, he was with Nanyang Technological University, Singapore. Since 2000, he has been with the Hong Kong University of Science and Technology, where he is currently a Professor. He unified all known constructions of perfect roots-of-unity (also known as CAZAC) sequences, which have been widely used as communication preambles and radar signals. He published two books and 220+ journal/conference publications and is the inventor of 38 patents. He pioneered the lattice approach to signal detection problems, including sphere decoding and complex lattice reduction-aided detection. His research areas include coding and information theory, wireless communications, optical camera communications, and thermographic signal processing. His joint work won the top prizes of 10+ project/paper competitions, including the 2014 HK U-21 IoT Gold Award for Revolutional Concept, the Best Paper Award of 2013 and 2016 Asia–Pacific Communications Conference, and the Best Mobile App Award at ACM MobiCom'2013. His co-invented picture barcode PiCode was highlighted as one of the four local innovations in the 2015 International IT Fest organized by the Office of the Government Chief Information Officer, Hong Kong. He served on the editorial boards of six journals, including the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He is the Past Chair of the Hong Kong Chapter of the IEEE Information Theory Society, and was the General/Program Chair of six conferences, including SETAŠ2018 held in Hong Kong. He is a Past Member of the Radio Spectrum Advisory Committee, Office of the Telecommunications Authority of the Hong Kong SAR Government.

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 and as a consultant of Huawei Technology company. 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.

Ting-Yi Wu received the Ph.D. degree in communication engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2013. He worked with the Network Technology Laboratory, National Chiao Tung University, as a Post-Doctoral Fellow, until 2015. He then joined the Signal-Processing and Communication Laboratory, HKUST; the Coordinated Science Laboratory, UIUC, Illinois; and the School of Electronics and Communication Engineering, SYSU, Guangzhou. He is currently a Principal Engineer with the Theory Laboratory, Central Research Institute, 2012 Labs, Huawei, who mainly focuses on erasure coding for storage, network, and communication.