Maximum Distance Separable Array Codes Allowing Partial Collaboration

Yuejia Zhang, Shiqiu Liu, *Member, IEEE*, and Li Chen, *Senior Member, IEEE*

***Abstract*— This letter considers the problem of repairing multiple node failures through partial collaboration in a distributed storage system (DSS). In the repair process, each failed node firstly connects to $d \geq k$ alive nodes to download data, and then exchanges data with other repairing nodes. Partial collaboration allows each failed node to only connect to some (not all) of the other repairing nodes to exchange data. Constructions of partially collaborative regenerating codes with $d = k$ at the minimum-storage regime have been studied before.We propose a code construction using the maximum distance separable (MDS) array codes to achieve $d > k$, and show that the constructed code asymptotically approaches the minimum storage repair point as the number of failed nodes grows.**

***Index Terms*— Array codes, partial collaboration, regenerating codes.**

I. Introduction

DISTRIBUTED storage systems (DSS) built on huge number of storage nodes, have been applied widely such as Google file system [1], Ocean Store [2] and Total Recall [3]. In DSS, a data file is encoded and stored over n nodes, each of which stores α symbols. The data collector can retrieve the data file by connecting to any k nodes, which is called the (n, k) *recovery property*, or sometimes called the *MDS*[1] *property*. While node failures occur, the system will trigger a repair process to protect the data by recovering the data in the failed nodes. When the regenerated data is the same as the one in the failed node, the repair is called an *exact repair*. Each failure will be repaired by using γ amount of data, where γ is called as the *repair bandwidth*. *Regenerating codes* were introduced by Dimarks *et al.* in [4] aiming at reducing the repair bandwidth. A tradeoff curve between storage and repair bandwidth was given. Two extremal cases on the tradeoff curve corresponding to the minimum storage and the minimum repair bandwidth are called the *Minimum Storage Repair* (MSR) and the *Minimum Bandwidth Repair* (MBR) respectively. Constructions of regenerating codes for those two special cases have been considered in the literature, in [5].

The storage systems are equipped with a huge number of storage nodes. It is more likely to occur multiple failures simultaneously. A *(fully) collaborative repair* mechanism for multiple failures was introduced in [6], [7], which allows the repairing nodes to exchange data among themselves. The (full) collaboration reduces the repair bandwidth compared with repairing multiple failures separately.

Code constructions for *Minimum Storage Collaborative Repair* (MSCR) and *Minimum Bandwidth Collaborative Repair* (MBCR) have been considered, for instance in [6]–[9]. In [9], the construction for MSCR is compatible with all parameters.

To give the system more flexible repair scenes, *partial collaboration* was introduced in [10]. Unlike the full collaboration, partial collaboration allows the repairing nodes to exchange data with some (but all) of the other repairing nodes. When t failures occur, each failed node first downloads β amount of data from each of d alive nodes, and then exchanges β' amount of data with the $t - s$ $(1 \leq s \leq t)$ other repairing nodes. The total repair bandwidth γ for one failed node is $d\beta + (t - s)\beta'$. When $s = 1$, the partial collaboration corresponds to the full collaboration, and when $s = t$, the collaboration vanishes. Code constructions for the two extremal cases *Minimum Storage Partially Collaborative Repair* (MSPR) and *Minimum Bandwidth Partially Collaborative Repair* (MBPR) have been considered in [10]–[12]. For the MSPR case, only the construction with $d = k$ has been considered.

In this letter, we construct the partially collaborative regenerating codes for exact repair at the minimum-storage regime with $d = k + s > k$. The existing constructions have not considered the case of $d > k$. We note that for a fixed system, the relation between d and k is fixed. For the code of the system with $d = k$, the repair bandwidth is $\gamma = d\beta + (t - s)\beta'$. When applying this code to the system with $d > k$, each failed node needs to download $(d - k)\beta$ redundant data to repair, so the repair bandwidth is not optimal. We aim to construct codes for the system with $d > k$ with less repair cost. The construction is based on an (n, k, l) *Maximum Distance Separable* (MDS) array code [13] over finite field $\mathbb{F}$, where l is the storage capacity per node. When node failures happen, the failures can be repaired through a partially collaborative repair process. In the following sections, we will define this MDS-array-code based partially collaborative regenerating code and show that the code achieves the MSPR asymptotically.

II. Repair of the First t Nodes

In this section, we construct an explicit MDS array code allowing partial collaboration repairing the first t failed nodes. Our main construction in Section III can be reduced to this construction when knowing the failed nodes' indices. Therefore, the construction of repairing the first t nodes can help to simplify the main construction. Before stating the construction,

Manuscript received February 27, 2020; revised April 9, 2020; accepted April 26, 2020. Date of publication May 6, 2020; date of current version August 12, 2020. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Project ID 61671486, and in part by the Shenzhen Science and Technology Innovation Council. The associate editor coordinating the review of this letter and approving it for publication was M. Baldi. *(Corresponding author: Shiqiu Liu.)*

Yuejia Zhang and Shiqiu Liu are with the School of Electronics and Communication Engineering, Sun Yat-sen University, Guangzhou 510006, China (e-mail: zhangyj43@mail2.sysu.edu.cn; liushq27@mail.sysu.edu.cn).

Li Chen is with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China (e-mail: chenli55@mail.sysu.edu.cn).

Digital Object Identifier 10.1109/LCOMM.2020.2992550

[1]Though the code satisfying this property may not be a maximum distance separable (MDS) code, i.e., does not satisfy the Singleton bound, it may be called the MDS property in some literatures for brevity.

an important corollary will be given, which can be extended from Lemma 2 in [9].

Let $C = (C_1, C_2, \ldots, C_n)$ denote a codeword of an (n, k, l) MDS array code $\mathcal{C}$ over finite field $\mathbb{F}_q$, and $C_i = (c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})^T \in \mathbb{F}_q^l$ is its ith coordinate. For simplicity, we also let C_i denote the ith node that stores the l-dimensional column vector.

Corollary 1: Let $\mathbb{F}_q$ denote a finite field with cardinality $q \geq n + s + 1$. Let $\kappa_{1,0}$, $\kappa_{1,1}$, $\kappa_{2,0}$, $\kappa_{2,1}$, $\ldots$, $\kappa_{s,0}$, $\kappa_{s,1}$, κ_{s+1}, κ_{s+2}, $\ldots$, κ_n be $n + s$ distinct elements of $\mathbb{F}_q$. Define an $(n, k, 2)$ MDS array code $\mathcal{C}$ over $\mathbb{F}_q$ by the following $2r$ parity-check equations

$$\kappa_{1,u}^p c_{1,u} + \kappa_{2,u}^p c_{2,u} + \cdots + \kappa_{s,u}^p c_{s,u} + \sum_{i=s+1}^{n} \kappa_i^p c_{i,u} = 0, \quad (1)$$

where $u = 0, 1$ and $p = 0, 1, \ldots, r-1$. Let $\sigma_i = c_{i,0} + c_{i,1}$. Then for any subset $R \subseteq \{s+1, s+2, \ldots, n\}$ with cardinality $|R| = d = k + s$, the values of $c_{1,0}, c_{1,1}, c_{2,0}, c_{2,1}, \ldots, c_{s,0}, c_{s,1}, \sigma_{s+1}, \ldots, \sigma_n$ can be calculated from $\{\sigma_i, i \in R\}$.

The construction of partially collaborative regenerating code repairing the first t nodes is as follows.

Denote $[n]$ as the set of integers $\{1, 2, \ldots, n\}$, $b + [n]$ as the set $\{b+1, b+2, \ldots, b+n\}$. Let $\kappa_{1,0}$, $\kappa_{1,1}$, $\kappa_{2,0}$, $\kappa_{2,1}$, $\ldots$, $\kappa_{t,0}$, $\kappa_{t,1}$, κ_{t+1}, κ_{t+2}, $\ldots$, κ_n be $n+t$ distinct elements of $\mathbb{F}$. We propose an (n, k, l) MDS array code over $\mathbb{F}_q$, where $q \geq n + t + 1$, $l = t - s + 2$. The code $\mathcal{C}$ is defined by the following parity-check equations

$$\sum_{i=1}^{t} \kappa_{i,0}^p c_{i,0} + \sum_{i=t+1}^{n} \kappa_i^p c_{i,0} = 0, \quad (2)$$

$$\sum_{i=a}^{a+s-1} \kappa_{i,1}^p c_{i,a} + \sum_{\substack{i \in [t] \backslash \\ \{a-1+[s]\}}} \kappa_{i,0}^p c_{i,a} + \sum_{i=t+1}^{n} \kappa_i^p c_{i,a} = 0, \quad (3)$$

where $p = 0, 1, \ldots, r-1$ and $a \in [t - s + 1]$. For $i \in [n]$ and $j \in [t - s + 1]$, let us define $\sigma_{i,j} = c_{i,0} + c_{i,j}$.

The repair process is as follows. In the download process, the failed node C_j ($j \in [t-s+1]$) downloads $\sigma_{i,j}$ from helper nodes C_i, where $i \in R, R \subseteq [n] \backslash [t]$ and $|R| = d = k + s$, while the failed nodes $C_{t-s+2}, \ldots, C_t$ download $\sigma_{i,t-s+1}$, where $i \in R$. Note that the equations of (2) and (3) have the same structure as those of (1). Therefore, after the download process, the failed node C_j ($j \in [t - s + 1]$) knows the values of $c_{j,0}, c_{j,j}, c_{j+1,0}, c_{j+1,j}, \ldots, c_{j+s-1,0}, c_{j+s-1,j}$ and $\sigma_{i,j}$, where $i \in [t] \backslash j - 1 + [s]$. Nodes $C_{t-s+2}, \ldots, C_t$ obtain the same information as node C_{t-s+1}. In the collaboration process, the failed node C_j ($j \in [t - s + 1]$) downloads $\sigma_{j,i}$ or $c_{j,i}$ from nodes C_i, where $i \in [t - s + 1] \backslash \{j\}$, and nodes $C_{t-s+2}, \ldots, C_t$ download $\sigma_{j,i}$ or $c_{j,i}$ from nodes C_i, where $i \in [t - s]$. After the collaboration process, the failed node C_j ($j \in [t]$) knows the values of $c_{j,0}, \ldots, c_{j,t-s+1}$. Therefore, node C_j ($j \in [t]$) can recover all its coordinates. In the download process, each failed node downloads $k + s$ symbols. While in the collaboration process, each failed node downloads $t - s$ symbols. Therefore, the total repair bandwidth is $(k + s) + (t - s) = k + t$.

III. Repair of Any t Nodes

In this section, we give an explicit code construction repairing any t failed nodes. According to the indices of the failed nodes, we partition the coordinates of a node into groups. The repair process of each group is the same as the case of repairing the first t failed nodes. When the repair of each group is completed, the total repair process is also completed. Specifically, we propose an (n, k, l) MDS array code over finite field $\mathbb{F}$, where $l = (t + 2 - s)^\omega$ and $\omega = \binom{n}{t}$. Note that the size of column vector will be large, which is difficult to implement in practice. Reducing the size of l is part of our future work. However, the construction gives a possible solution for the system with $d > k$.

Definition 1: Let $\kappa_{i,j}$, where $i \in [n]$ and $j \in \{0, 1\}$, be $2n$ distinct elements of $\mathbb{F}_q$, where $q \geq 2n + 1$. The code $\mathcal{C}$ is defined by the following rl parity-check equations, such that for $p = 0, 1, \ldots, r-1$ and $a = 0, 1, \ldots, l-1$,

$$\sum_{i=1}^{n} \kappa_{i,\upsilon(i,a)}^p c_{i,a} = 0,$$

where

$$\upsilon : [n] \times \{0, 1, \cdots, l-1\} \to \{0, 1\}$$
$$(i, a)$$
$$\mapsto \left(\sum_{\substack{\Omega \subseteq [n], \\ |\Omega| = t, \Omega \ni i}} \begin{cases} \mathbb{1}\{a_{\mu(\Omega)} = 1\} + \cdots + \\ \mathbb{1}\{a_{\mu(\Omega)} = \hat{\zeta}(\Omega, i)\}, & \zeta(\Omega, i) \leq s \\ \\ \mathbb{1}\{a_{\mu(\Omega)} = \zeta(\Omega, i) - s + 1\} + \cdots + \\ \mathbb{1}\{a_{\mu(\Omega)} = \hat{\zeta}(\Omega, i)\}, & \zeta(\Omega, i) > s \end{cases} \right) \pmod 2.$$

$\mathbb{1}$ is the indicator function. For a given $a = 0, 1, \ldots, l-1$, $a_\omega, a_{\omega-1}, \ldots, a_1$ be the digits of its expansion in the base of $t + 2 - s$. μ be a bijection between the set of t-subsets $\{\Omega : \Omega \subseteq [n], |\Omega| = t\}$ and the numbers $\{1, 2, \ldots, \omega\}$ as follows,

$$\mu(\{i_t, i_{t-1}, \ldots, i_1\}) = \sum_{j=0}^{t-1} \binom{i_{t-j} - 1}{t - j},$$

where $n \geq i_t > i_{t-1} > \cdots > i_1 \geq 1$, and we use the convention that $\binom{i_1}{i_2} = 0$ if $i_1 < i_2$. For a set $\Omega \subseteq [n]$ and an element $i \in \Omega$, let $\zeta(\Omega, i) = |\{j : j \in \Omega, j \leq i\}|$ be the number of elements in Ω which are not greater than i and $\hat{\zeta}(\Omega, i) = \min\{\zeta(\Omega, i), t - s + 1\}$.

For a given $u = 0, 1, \ldots, t - s + 1$, let $a(i, u) := (a_\omega, \ldots, a_{i+1}, u, a_{i-1}, \ldots, a_1)$. For $i \in [n]$, $u \in [t - s + 1]$, and any $a \in \{0, 1, \ldots, l-1\}$, we define $\sigma_{i,a(\mu(\Omega),u)} = c_{i,a(\mu(\Omega),0)} + c_{i,a(\mu(\Omega),u)}$. The following Lemma will show the obtained information from the download step, which helps state the repair process.

Lemma 1: Let $\Omega = \{i_j\}_{j \in [t]}$ be the set of failed nodes. For any set of helper nodes $R \subseteq [n] \backslash \Omega$, where $|R| = k + s$, $u \in [t - s + 1]$ and $a \in \{0, 1, \ldots, l-1\}$, the values of $c_{i_u,a(\mu(\Omega),0)}$, $c_{i_u,a(\mu(\Omega),u)}$, $c_{i_{u+1},a(\mu(\Omega),0)}$, $c_{i_{u+1},a(\mu(\Omega),u)}$, $\ldots$, $c_{i_{u+s-1},a(\mu(\Omega),0)}$, $c_{i_{u+s-1},a(\mu(\Omega),u)}$, and $\{\sigma_{i,a(\mu(\Omega),u)} : i \in \Omega \backslash \{i_u, i_{u+1}, \ldots, i_{u+s-1}\}\}$ are uniquely determined by $\{\sigma_{i,a(\mu(\Omega),u)} : i \in R\}$.

Proof: The parity-check equations corresponding to the row labeled by $a(\mu(\Omega),0), a(\mu(\Omega),1), \ldots, a(\mu(\Omega),t-s+1)$ can be rewritten as

$$\sum_{i=1}^{n} \kappa_{i,\upsilon(i,a(\mu(\Omega),u))}^{p} c_{i,a(\mu(\Omega),u)} = 0,$$

where $p = 0, 1, \ldots, r-1$, $u = 0, 1, \ldots, t-s+1$.

According to the definition of function υ, if $i \notin \Omega$, the value of $\upsilon(i,a)$ does not depend on the digit of a at the position $\mu(\Omega)$. Therefore, for any $a \in \{0,1,\ldots,l-1\}$,

$$\begin{aligned}\upsilon(i,a(\mu(\Omega),0)) &= \upsilon(i,a(\mu(\Omega),1)) = \cdots \\ &= \upsilon(i,a(\mu(\Omega),t-s+1)), \quad i \in [n]\backslash\Omega.\end{aligned}$$

Likewise, for $i_j \in \Omega$ and $j \in [t]$, we have

$$\begin{aligned}\upsilon(i_j,a(\mu(\Omega),0)) &= \upsilon(i_j,a(\mu(\Omega),u)), \\ &\quad u \in [t-s+1], \quad j \in [t]\backslash u-1+[s], \\ \upsilon(i_j,a(\mu(\Omega),0)) &\neq \upsilon(i_j,a(\mu(\Omega),u)), \\ &\quad u \in [t-s+1], \quad j \in u-1+[s].\end{aligned}$$

Here, we define

$$\begin{aligned}\kappa_i &:= \kappa_{i,\upsilon(i,a(\mu(\Omega),0))} = \cdots = \kappa_{i,\upsilon(i,a(\mu(\Omega),t-s+1))}, \\ &\quad i \in [n]\backslash\Omega, \\ \kappa'_{i_j,0} &:= \kappa_{i_j,\upsilon(i_j,a(\mu(\Omega),0))} = \kappa_{i_j,\upsilon(i_j,a(\mu(\Omega),u))}, \\ &\quad u \in [t-s+1], \quad j \in [t]\backslash u-1+[s], \\ \kappa'_{i_j,1} &:= \kappa_{i_j,\upsilon(i_j,a(\mu(\Omega),u))}, \\ &\quad u \in [t-s+1], \quad j \in u-1+[s].\end{aligned}$$

Since the elements $\kappa'_{i_1,0}, \kappa'_{i_2,0}, \ldots, \kappa'_{i_t,0}, \kappa'_{i_1,1}, \kappa'_{i_2,1}, \ldots, \kappa'_{i_t,1}$ and κ_i are all distinct,

$$\begin{aligned}&\sum_{j=1}^{t} (\kappa'_{i_j,0})^p c_{i_j,a(\mu(\Omega),0)} + \sum_{i\in[n]\backslash\Omega} \kappa_i^p c_{i,a(\mu(\Omega),0)} = 0, \\ &\sum_{j=u}^{u+s-1} (\kappa'_{i_j,1})^p c_{i_j,a(\mu(\Omega),u)} + \sum_{\substack{j\in[t]\backslash \\ \{u-1+[s]\}}} (\kappa'_{i_j,0})^p c_{i_j,a(\mu(\Omega),u)} \\ &\quad + \sum_{i\in[n]\backslash\Omega} \kappa_i^p c_{i,a(\mu(\Omega),u)} = 0,\end{aligned}$$

where $p = 0,1,\ldots,r-1$ and $u = 1,2,\ldots,t-s+1$. Since the above equations have the same form as those defined in (2) and (3), this Lemma follows immediately from Corollary 1. ∎

The repair process involves two steps, download and partial collaboration, which are described below.

Download: Each failed node C_{i_j}, where $j \in [t-s+1]$, downloads $\{\sigma_{i,a(\mu(\Omega),j)} : a_{\mu(\Omega)} = 0\}$ from helper nodes C_i, where $i \in R$, $R \subseteq [n] \setminus \Omega$ and $|R| = d = k+s$, while the other failed nodes $C_{i_{t-s+2}}, \ldots, C_{i_t}$ download $\{\sigma_{i,a(\mu(\Omega),t-s+1)} : a_{\mu(\Omega)} = 0, i \in R\}$. In the download process, each failed node downloads $(k+s)\frac{l}{t-s+2}$ symbols. Based on Lemma 1, failed node C_{i_j} where $j \in [t-s+1]$, knows the values of $\{c_{i_j,a}, \ldots, c_{i_{j+s-1},a} : a_{\mu(\Omega)} = 0\}, \{c_{i_j,a(\mu(\Omega),j)}, \ldots, c_{i_{j+s-1},a(\mu(\Omega),j)} : a_{\mu(\Omega)} = 0\}$ and $\{\sigma_{i_{j'},a(\mu(\Omega),j)} : a_{\mu(\Omega)} = 0, j' \in [t]\backslash j-1+[s]\}$. Nodes $C_{i_{t-s+2}}, \ldots, C_{i_t}$ obtain the same information as node $C_{i_{t-s+1}}$.

Partial Collaboration: The failed nodes C_{i_j}, where $j \in [t-s+1]$, download $\{\sigma_{i_j,a(\mu(\Omega),j')} : a_{\mu(\Omega)} = 0\}$ from the nodes $C_{i_{j'}}$, where $j' \in [t-s+1]\backslash\{j\}$, and the failed nodes C_{i_j}, where $j \in \{t-s+2,\ldots,t\}$, download $\{\sigma_{i_j,a(\mu(\Omega),j')} : a_{\mu(\Omega)} = 0\}$ from the nodes $C_{i_{j'}}$, where $j' \in [t-s]$. In the collaboration process, each failed node C_{i_j}, where $j \in [t]$, downloads $(t-s)\frac{l}{t-s+2}$ symbols, and knows the values of $\{c_{i_j,a(\mu(\Omega),0)}, c_{i_j,a(\mu(\Omega),1)}, \ldots, c_{i_j,a(\mu(\Omega),t-s+1)} : a_{\mu(\Omega)} = 0\}$. Therefore, C_{i_j}, where $j \in [t]$, can recover all its coordinates.

The following example illustrates the above code construction and its repair process.

Example 1: Consider an MDS array code with parameters: $n = 7, t = 3, s = 2, \omega = 35$ and $l = (t-s+2)^\omega = 3^\omega$. Let $\Omega = \{5,4,1\}$ denote the set of failed nodes, then $\mu(\{5,4,1\}) = 8$. We partition the coordinates of a node into $l/3$ groups of size 3, where each group is formed by the coordinates with indices $a(8,0), a(8,1)$ and $a(8,2)$, respectively.

For $p = 0,1,2,\ldots,r-1$ and $u = 0,1,2$, the parity-check equations become

$$\sum_{i=1}^{n} \kappa_{i,\upsilon(i,a(8,u))}^{p} c_{i,a(8,u)} = 0.$$

According to the definition of function υ, if $i \notin \Omega$, the value of $\upsilon(i,a)$ does not depend on the digit of a in position 8. Thus, for $i \in [n]\backslash\{5,4,1\}$, we have $\upsilon(i,a(8,0)) = \upsilon(i,a(8,1)) = \upsilon(i,a(8,2))$. For simplicity, let $\kappa_i := \kappa_{i,\upsilon(i,a(8,0))} = \kappa_{i,\upsilon(i,a(8,1))} = \kappa_{i,\upsilon(i,a(8,2))}$.

For $i_j \in \Omega$, let $\Psi_j = \{\Omega' : i_j \in \Omega', |\Omega'| = t, \Omega' \neq \Omega\}$. For $\upsilon(i,a)$, when $i = 1$, we can have

$$\begin{aligned}&\upsilon(1,a) \\ &= \left(\mathbb{1}\{a_8 = 1\} + \sum_{\Omega'\in\Psi_1} \begin{cases} \mathbb{1}\{a_{\mu(\Omega')} = 1\} + \cdots + \\ \mathbb{1}\{a_{\mu(\Omega')} = \hat{\zeta}(\Omega',i)\}, \\ \qquad\qquad \zeta(\Omega',i) \le s \\ \\ \mathbb{1}\{a_{\mu(\Omega')} = \zeta(\Omega',i) - s + 1\} \\ + \cdots + \mathbb{1}\{a_{\mu(\Omega')} = \hat{\zeta}(\Omega',i)\}, \\ \qquad\qquad \zeta(\Omega',i) > s \end{cases} \right) \\ &\qquad \pmod 2.\end{aligned}$$

For each group, $a(8,0), a(8,1)$ and $a(8,2)$ only have different digits at position 8. The difference of the value of υ in a group depends on whether a_8 is equal to 1 and we have

$$\upsilon(1,a(8,0)) = \upsilon(1,a(8,2)), \quad \upsilon(1,a(8,0)) \neq \upsilon(1,a(8,1)).$$

For simplicity, let

$$\kappa'_{1,0} := \kappa_{1,\upsilon(1,a(8,0))} = \kappa_{1,\upsilon(1,a(8,2))}, \quad \kappa'_{1,1} := \kappa_{1,\upsilon(1,a(8,1))}.$$

Similarly, when $i = 4$ and $i = 5$, we can have

$$\begin{aligned}&\kappa'_{4,0} := \kappa_{4,\upsilon(4,a(8,0))}, \quad \kappa'_{4,1} := \kappa_{4,\upsilon(4,a(8,1))} = \kappa_{4,\upsilon(4,a(8,2))}, \\ &\kappa'_{5,0} := \kappa_{5,\upsilon(5,a(8,0))} = \kappa_{5,\upsilon(5,a(8,1))}, \quad \kappa'_{5,1} := \kappa_{5,\upsilon(5,a(8,2))}.\end{aligned}$$

TABLE I

THE REPAIR PROCESS

Nodes	Download ($i \in R$)	Download Obtained	Collaboration
1	$\{\sigma_{i,a(8,1)} : a_8 = 0\}$	$\{c_{1,a(8,0)}, c_{1,a(8,1)}, c_{4,a(8,0)}, c_{4,a(8,1)}, \sigma_{5,a(8,1)} : a_8 = 0\}$	Node 4: $\{\sigma_{1,a(8,2)} : a_8 = 0\}$
4	$\{\sigma_{i,a(8,2)} : a_8 = 0\}$	$\{c_{4,a(8,0)}, c_{4,a(8,2)}, c_{5,a(8,0)}, c_{5,a(8,2)}, \sigma_{1,a(8,2)} : a_8 = 0\}$	Node 1: $\{\sigma_{4,a(8,1)} : a_8 = 0\}$
5	$\{\sigma_{i,a(8,2)} : a_8 = 0\}$	$\{c_{4,a(8,0)}, c_{4,a(8,2)}, c_{5,a(8,0)}, c_{5,a(8,2)}, \sigma_{1,a(8,2)} : a_8 = 0\}$	Node 1: $\{\sigma_{5,a(8,1)} : a_8 = 0\}$

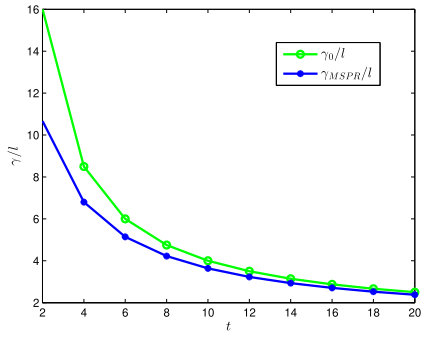

Fig. 1. Comparison of the repair bandwidth between code proposed in Section III and MSPR point for $s = 2$. When t grows, the repair bandwidth of the constructed code appoaches the MSPR point asymptotically.

Therefore, for $p = 0, 1, \ldots, r-1$, the parity-check equations become

$$(\kappa'_{1,0})^p c_{1,a(8,0)} + (\kappa'_{4,0})^p c_{4,a(8,0)} + (\kappa'_{5,0})^p c_{5,a(8,0)} + \sum_{i\in[n]\setminus\Omega} \kappa_i^p c_{i,a(8,0)} = 0,$$

$$(\kappa'_{1,1})^p c_{1,a(8,1)} + (\kappa'_{4,1})^p c_{4,a(8,1)} + (\kappa'_{5,0})^p c_{5,a(8,1)} + \sum_{i\in[n]\setminus\Omega} \kappa_i^p c_{i,a(8,1)} = 0,$$

$$(\kappa'_{1,0})^p c_{2,a(8,2)} + (\kappa'_{4,1})^p c_{3,a(8,2)} + (\kappa'_{5,1})^p c_{5,a(8,2)} + \sum_{i\in[n]\setminus\Omega} \kappa_i^p c_{i,a(8,2)} = 0.$$

Table I shows the downloaded information of each failed node, the obtained information after the download process and the exchanged information in the partial collaboration process.

IV. ACHIEVABILITY OF THE MINIMUM STORAGE

The code construction of Section III has $\alpha = l$ and $d = k + s$. In the repair process, each failed node downloads $\beta = \frac{l}{t-s+2}$ symbols from each of the d nodes, and then downloads $\beta' = \frac{l}{t-s+2}$ symbols from each of the other $t - s$ repairing nodes. Therefore, the total repair bandwidth is

$$\gamma_0 = d\beta + (t-s)\beta' = (k+t)\frac{l}{t-s+2}.$$

Note that at the MSPR point, where $\alpha_{\text{MSPR}} = l$ and $d = k + s$, the repair bandwidth γ_{MSPR} should satisfy

$$\gamma_{\text{MSPR}} = \frac{(d+t-s)l}{d-k+t-s+1} = (k+t)\frac{l}{t+1}.$$

Therefore, the repair bandwidth of the constructed code is

$$\gamma_0 = (1 + \frac{s-1}{t-s+2})\gamma_{\text{MSPR}}.$$

It can be seen that for a fixed s ($1 < s < t$), when t grows, the repair bandwidth γ_0 approaches the repair bandwidth γ_{MSPR} at the MSPR point asymptotically, which is demonstrated by Fig. 1.

V. CONCLUSION

This letter has proposed a code construction allowing partial collaboration based on MDS array codes for $d > k$. As the number of failures grows, the repair bandwidth of our construction approaches the MSPR point asymptotically. One of the remaining open problems is to construct codes allowing partial collaboration for $d > k$ to achieve the MSPR point. Existing work for caching with partial failure repair, e.g. [14], may help solve the challenge. This will be part of our future work. It should also be pointed out that the construction has a large node size l, which still distances this proposal from the current art of practical implementation. However, this construction provides a possible solution for systems with $d > k$, while further reducing the node size complements our future work.

REFERENCES

[1] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. 19th ACM Symp. Operating Syst. Princ.*, Oct. 2003, pp. 19–43.

[2] J. Kubiatowicz *et al.*, "OceanStore: An architecture for global-scale persistent storage," in *Proc. ACM ASPLOS*, 2000, pp. 1–12.

[3] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G. Voelker, "Total recall: System support for automated availability management," in *Proc. 1st Conf. Netw. Sys. Design Implem. (NSDI)*, 2004, p. 25.

[4] A. G. Dimakis, P. B. Godfrey, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," in *Proc. IEEE INFOCOM 26th IEEE Int. Conf. Comput. Commun.*, May 2007, pp. 2000–2008.

[5] S. Goparaju, A. Fazeli, and A. Vardy, "Minimum storage regenerating codes for all parameters," *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6318–6328, Oct. 2017.

[6] Y. Hu, Y. Xu, X. Wang, C. Zhan, and P. Li, "Cooperative recovery of distributed storage systems from multiple losses with network coding," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 2, pp. 268–276, Feb. 2010.

[7] A.-M. Kermarrec, N. Le Scouarnec, and G. Straub, "Repairing multiple failures with coordinated and adaptive regenerating codes," in *Proc. Int. Symp. Netw. Coding*, Jul. 2011, pp. 1–6.

[8] K. W. Shum and Y. Hu, "Cooperative regenerating codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7229–7258, Nov. 2013.

[9] M. Ye and A. Barg, "Cooperative repair: Constructions of optimal MDS codes for all admissible parameters," *IEEE Trans. Inf. Theory*, vol. 65, no. 3, pp. 1639–1656, Mar. 2019.

[10] S. Liu and F. Oggier, "On storage codes allowing partially collaborative repairs," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2014, pp. 2440–2444.

[11] S. Liu and F. Oggier, "Two storage code constructions allowing partially collaborative repairs," in *Proc. IEEE Int. Symp. Inf. Theory*, Oct. 2014, pp. 378–382.

[12] S. Liu and F. Oggier, "On applications of orbit codes to storage," *Adv. Math. Commun.*, vol. 10, no. 1, pp. 113–130, Mar. 2016.

[13] M. Blaum, P. G. Farell, and H. van Tilborg, "Array codes," in *Handbook Coding Theory*, vol. 2, V. Pless and W. C. Huffman Eds. Amsterdam, The Netherlands: Elsevier, 1998, ch. 22, pp. 1855–1909.

[14] N. Mital, K. Kralevska, C. Ling, and D. Gunduz, "Storage-repair bandwidth trade-off for wireless caching with partial failure and broadcast repair," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Nov. 2018, pp. 1–5.