Asymptotic Interference Alignment for Optimal Repair of MDS Codes in Distributed Storage

Viveck R. Cadambe, *Member, IEEE*, Syed Ali Jafar, *Senior Member, IEEE*, Hamed Maleki, *Student Member, IEEE*, Kannan Ramchandran, *Fellow, IEEE*, and Changho Suh, *Member, IEEE*

***Abstract*—The high repair bandwidth cost of (n, k) maximum distance separable (MDS) erasure codes has motivated a new class of codes that can reduce repair bandwidth over that of conventional MDS codes. In this paper, we address (n, k, d) exact repair MDS codes, which allow for any single failed node to be repaired exactly with access to any arbitrary set of d survivor nodes. We show the existence of exact repair MDS codes that achieve minimum repair bandwidth (matching the cut-set lower bound) for arbitrary admissible (n, k, d), i.e., $k \leq d \leq n-1$. Moreover, we extend our results to show the optimality of our codes for multiple-node failure scenarios in which an arbitrary set of $r \leq n-k$ failed nodes needs to repaired. Our approach is based on *asymptotic interference alignment* proposed by Cadambe and Jafar. As a byproduct, we also characterize the capacity of a class of multisource nonmulticast networks.**

***Index Terms*—Distributed storage, exact-repair maximum distance separable (MDS) codes, interference alignment, network codes.**

I. Introduction

In distributed storage systems, maximum distance separable (MDS) erasure codes are well-known coding schemes that can offer maximum reliability for a given storage overhead. Consider a scenario where a file of size M is to be stored in n distributed storage nodes. The file is equally split into k parts of size M/k and stored in the first k storage nodes, known as systematic nodes. The remaining $(n-k)$ nodes, known as parity nodes or nonsystematic nodes, store data of the same size, i.e., M/k, adding redundancy to protect from failures of storage nodes. The parity nodes are designed such that a failure of up to $(n-k)$ storage nodes can be tolerated, i.e., any k nodes out of the n nodes can recover the original file. Clearly, for this problem, storing the data using an (n, k) MDS code suffices to achieve the required reconstruction criterion, since an MDS code protects the data from $(n-k)$ erasures.

Consider the case where $r \leq n-k$ nodes fail, and a repair center is introduced to recover the data stored in the failed nodes. The total amount of data to be downloaded by the repair center to regenerate failed nodes will be henceforth referred to as the repair bandwidth. Clearly, a repair bandwidth of M suffices to repair r failed nodes since the repair center can download data of total size M from any k of the remaining $(n-r)$ surviving nodes to reconstruct the entire file, and from it, the data stored in the failed nodes. However, note the inherent inefficiency in the solution: to repair r nodes, each of size M/k, the newcomer downloads data of size M, i.e., $\frac{k}{r}$ times the size of the data to be repaired. In particular, if $r = 1$ node fails, the total data M downloaded by the repair center is k times the amount of data needed to be replaced. A question of interest is whether this inefficiency is fundamental or whether the node can be repaired with downloading data of size less than M. More specifically, we ask the following question: *what is the minimum repair bandwidth required to repair r failed nodes?* This question has been studied previously for the case of a single-node failure from two perspectives [3]–[10]. The first is called functional repair [3] and the second is called exact repair [4]–[10].

The functional repair problem requires that the failed nodes are replaced so that the reconstructed new nodes along with the other nodes satisfy the MDS code property. In other words, the repaired nodes are functionally equivalent to the originally stored data. Note that the data in the repaired nodes need not be identical to the data in the failed nodes: all that is required is that the repaired nodes along with the other nodes form an MDS code. It has been shown in [3] that this problem is equivalent to the well-studied multicast problem. With the help of the well-established results on the multicast problem, Dimakis *et al.* [3] have shown that in the case of any single node failure, when accessing to any arbitrary $d \geq k$[1] of the remaining $(n-1)$ surviving nodes, the minimum repair bandwidth required is

$$M \cdot \frac{d}{k(d+1-k)}. \tag{1}$$

[1]Note that the repair center has to connect to at least k nodes to recover the lost data.

Manuscript received September 29, 2011; revised December 18, 2012; accepted December 20, 2012. Date of publication January 03, 2013; date of current version April 17, 2013. V. Cadambe, S. Jafar, and H. Maleki were supported in part by the Office of Naval Research under Grant N00014-12-10067. K. Ramchandran and C. Suh were supported in part by the Air Force Office of Scientific Research under Grant FA9550-09-1-0120, Defense Threat Reduction Agency under Grant HDTRA1-09-1-0032, and National Science Foundation under Grant CCF-0830788. This paper is based on work done contemporaneously and independently by two groups: (1) V. R. Cadambe, S. A. Jafar, and H. Maleki [1]; (2) C. Suh and K. Ramchandran [2]. The authors decided to consolidate their work into a single manuscript and to list the authors' names in alphabetical order.

V. R. Cadambe is with the Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139 USA, and also with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215 USA (e-mail: viveck@mit.edu).

S. A. Jafar and H. Maleki are with the Department of Electrical Engineering and Computer Science, University of California at Irvine, Irvine, CA 92617 USA (e-mail: syed@uci.edu; hmaleki@uci.edu).

K. Ramchandran is with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94704 USA (e-mail: kannanr@eecs.berkeley.edu).

C. Suh is with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea (e-mail: chsuh@kaist.ac.kr).

Communicated by A. Lozano, Associate Editor for Communications.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2013.2237752

This result implies that functional repair requires a smaller repair bandwidth over that of the naive approach which constructs all the data with access to any k nodes. It gains by a factor of $\frac{k(d+1-k)}{d}$, e.g., for $(n,k,d)=(n,6,30)$ there is a 5× bandwidth reduction.

In this paper, we focus on the exact repair problem where the failed nodes are required to be replaced with identical copies of the failed nodes. An advantage of exact repair is that the storage system can be oblivious to the repair operation, as the storage code coefficients remain unchanged after the repair operation. This is contrast to functional repair where the code is changed in general whenever the failed nodes are repaired. Moreover, exact repair guarantees a desirable systematic structure to the code which enables a client to download the data without any decoding.

Since any solution to the exact repair problem is also a solution to the functional repair problem, the bound (1) can serve as a lower bound to the minimum repair bandwidth for the exact repair problem. However, whether or not this bound is tight has been open. In this paper, we settle the open problem on the minimum repair bandwidth. Specifically, we show that the lower bound of (1) is indeed tight and achievable via linear codes for arbitrary values of (n,k,d). Moreover, as a byproduct, we find that our solution to the exact repair problem leads to the capacity characterization of a class of multisource nonmulticast networks.

A. Related Work

The topic of exact repair of MDS codes has received attention in the recent literature [3]–[8], [11]. It was pioneered by Wu and Dimakis [4] who showed the optimality of the bound (1) for the case of $(n,k)=(4,2)$. Later, Cullina *et al.* [5] showed the optimality of the bound (1) for $(n,k,d)=(5,3,4)$. Progress in construction of exact repair codes beyond these special cases was made by Shah *et al.* [7], [12], [13], Wu [6], and Suh and Ramchandran [8]. Shah *et al.* [7], [13] developed partial exact repair codes for the case of $d\geq 2k-1$, where exact repair is limited to the systematic component of the code. Suh and Ramchandran [8] showed the optimality of (1) for the repair of all nodes (including parity nodes) for $d\geq 2k-1$. However, these results [7], [8], [13] relied on the assumption that all of the surviving systematic nodes participate in repair. Later it was shown in [9] that this constraint can be removed without loss of optimality. In [9], a product matrix-based construction is developed for $d\geq 2k-2$ that do not depend on the assumption. On the other hand, Wu [6] and Rashmi *et al.* [12] presented explicit code constructions for the case of $d=k+1$.

For all other remaining cases, the establishment of fundamental limits of repair bandwidth for exact repair remained open. Our main contribution of this paper is to settle this open problem. Note that the condition of $d\geq 2k-2$ restricts the code rate of $\frac{k}{n}$ to be at most $\frac{1}{2}+\frac{1}{n}$, as $n\geq d$. This means that the unresolved regime of $d<2k-2$ is especially relevant for high code rate, which has been of significant interest in practice. Indeed, many literature in the MDS code design for storage systems have been devoted to systems with two parity nodes, i.e., $n-k=2$ [14], [15]. For this practically relevant low redundancy regime, the only previous insight comes from [7] where it is shown that for $d<2k-3$, scalar linear codes cannot achieve the limit of (1). The question of whether (1) is tight allowing for vector linear and nonlinear codes has been open.

A related line of work is the area of cooperative regenerating codes [16], [17] which deal with a *multiple*-node failure scenario, unlike the previous works intended for a single-node failure case. In the model considered in these references, when $r\geq 1$ nodes fail, r new nodes enter the system with each new node intending to repair one failed destination. These r nodes can cooperate in a limited manner. In this model, the authors aim to minimize the amount of repair bandwidth—a weighted sum of the bandwidth required between each surviving node and a regenerating node, and the bandwidth required among the r regenerating nodes. Hu *et al.* [16] solve this problem for the case of *functional* regeneration of the r failed nodes. In the case of exact repair, for $d=k$, Shum [18] shows that a simple Reed–Solomon code can achieve the functional repair bandwidth lower bound. On the other hand, our study considers the case where the r regenerating nodes can *fully* cooperate each other, and for this case, we show the optimality of (1) for all admissible values of (n,k,d,r).

B. Summary of Contribution

Our first result concerns an (n,k,d) storage system—a storage system where a file of size M is stored using an (n,k) MDS code, and a single-node failure is repaired by a repair center connecting to an arbitrary set of d surviving nodes $(d\geq k)$.

Theorem 1 (Single Node Failure): Consider an (n,k) MDS code used to store a file of size M. Let $x\in\{1,2,\ldots,n\}$ indicate a failure node. Let $Y\subseteq\{1,2,\ldots,n\}-\{x\}$ denote the set of nodes participating in repair. Let $B_{x,Y}$ represent the corresponding repair bandwidth. Then, for any $x\in\{1,2,\ldots,n\}, Y\subseteq\{1,2,\ldots,n\}-\{x\}$

1)
$$\frac{B_{x,Y}}{M}\geq\frac{d}{k(d+1-k)}$$

2) there exists an MDS code such that
$$\lim_{M\to\infty}\frac{B_{x,Y}}{M}=\frac{d}{k(d+1-k)}. \tag{2}$$

Since $B_{x,Y}$ depends only on d, we denote this by B_d. Theorem 2 (to be stated shortly) includes the above theorem as a special case.

Remark 1: Notice that $\frac{B_d}{M}$ approaches (1) as M tends to infinity. This implies that *exact repair is asymptotically equally efficient as functional repair in the limit of large file size*. Unlike [7] and [8] which provide explicit code constructions, our approach shows only the existence of optimal exact repair codes. Instead, our result affords the extension to the multiple-node failure scenario that was not available in the previous literature. This generalization and other notable aspects of our result are listed below. □

1) Our approach can be easily generalized to the multiple-node failure scenario where r nodes fail. We focus on the case of $r\leq\min(k,n-k)$, since for $r>k$, the optimal

repair strategy is straightforward: reconstructing the entire file and then regenerating the failed nodes. Using our approach, we characterize the optimal repair bandwidth analogous to (2), thus developing Theorem 2 that will be stated shortly.

2) An interesting property of our codes is that the code coefficients can be designed regardless of (d, r). In other words, the same code can be used to handle one or multiple node failures, and arbitrary admissible values of d. This is in contrast to previous works [7], [8] where the code coefficients depend on the value of d. Thus, our solution provides more flexibility to the code design.
3) An interesting aspect of our code is that the coding submatrices are diagonal and hence optimal from the perspective of encoding/update complexity [19].

Theorem 2 (Multiple Node Failures): Consider an (n, k) MDS code. Suppose $X \subset \{1, 2, \ldots, n\}$ indicate the set of nodes that fail. The repair center regenerates all the nodes in X by connecting to nodes $Y \subseteq \{1, 2, \ldots, n\} - X$. Let $B_{X,Y}$ denote the corresponding repair bandwidth for the simultaneous repair of nodes in X. Then, for any $X \subset \{1, 2, \ldots, n\}$, $Y \subseteq \{1, 2, 3, \ldots n\} - X$

1)
$$B_{X,Y} \geq \frac{rd}{k(d + r - k)}$$

2) there exists an MDS code such that

$$\lim_{M \to \infty} \frac{B_{X,Y}}{M} = \frac{rd}{k(d + r - k)} \tag{3}$$

where $r = |X| \leq n - k$ and $d = |Y| \leq n - r$.

Since $B_{X,Y}$ depends only on $r = |X|$ and $d = |Y|$, we denote the minimum repair bandwidth by $B_{r,d}$ in the remainder of this paper.

Proof: See Sections III and IV for the achievability proof. For the converse proof, see the proof of the second claim of Theorem 3 in Section V. ■

Notice from (2) and (3) that $B_{r,d} \leq r \cdot B_d$ for sufficiently large file size. Considering the fact that $r \cdot B_d$ is the minimum repair bandwidth under noncooperative one-by-one repair, our multiple node failure scenario requires less repair bandwidth. This is because our setup considers full cooperation: all of the failed nodes are repaired by only one repair center. In fact, our setup is a special case of a partially cooperative repair setup [16], [18] where there are r separate repair centers cooperating each other through some limited communication links. Note that our case can be viewed as the case where there are r separate repair centers fully cooperating each other. Focusing on the full cooperation case, our result shows the optimality of (3) for all admissible values of (n, k, d, r). On the other hand, Shum [18] shows the optimality for the case of $d = k$, although its setup is more general.

II. Role of Interference Alignment in Exact Repair

Making use of the connection made in [8] between the storage repair problem and the wireless interference channel problem, we leverage the scheme in [20] to show the information-theoretic optimality of exact-repair codes for all feasible values of (n, k, d, r). Let us first review a simple example of $(4, 2, 3)$ codes which will illustrate the connection to the wireless interference channel problem through the concept of interference alignment.

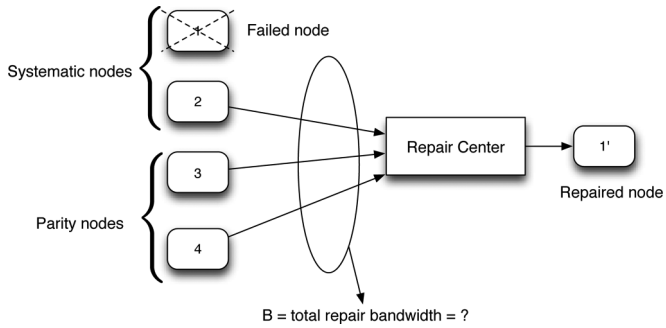

Fig. 1. Pictorial representation of problem definition for $n = 4, k = 2, r = 1, d = 3$.

Review of $(4, 2, 3)$ *Exact-Repair MDS Codes [4]*: We assume that the source file size M is 4 so that each node stores $M/k = 2$. Let $\mathbf{a} = (a_1, a_2)^t$ and $\mathbf{b} = (b_1, b_2)^t$ be two-dimensional vectors where $(\cdot)^t$ indicates a transpose. Systematic nodes 1 and 2 store uncoded information in the form of row vectors, i.e., $\mathbf{a}^t$ and $\mathbf{b}^t$, respectively. Let $\mathbf{A}_i$ and $\mathbf{B}_i$ be 2-by-2 encoding submatrices (i.e., $[\mathbf{A}_i; \mathbf{B}_i]$ constitutes a generator submatrix) for parity node i $(i = 1, 2)$. For example, parity node 1 stores a two-dimensional vector $\mathbf{a}^t\mathbf{A}_1 + \mathbf{b}^t\mathbf{B}_1$. Assuming that $\mathbf{A}_i$, $\mathbf{B}_i$ are chosen so that the code is an MDS code, $\mathbf{a}$ and $\mathbf{b}$ can be decoded from any two nodes.

Suppose that node 1 fails. One can download four linear combinations of (a_1, a_2, b_1, b_2) by downloading all of the information from any two nodes, and thus can recover (a_1, a_2, b_1, b_2). With this naive approach, node 1 can be repaired using a total repair bandwidth of four equations. However, we will show that this repair can be done by downloading only three equations in total, matching the bound of Theorem 1. The idea of achievability is interference alignment.

Here, we use a scalar linear code where each survivor node uses a projection vector $\mathbf{v}_{\alpha i}$ to project its data into a scalar. The example illustrated in Fig. 2 shows exact repair of failed node 1 using interference alignment. By connecting to three nodes, we get: $\mathbf{b}^t\mathbf{v}_{\alpha 1}; \mathbf{a}^t(\mathbf{A}_1\mathbf{v}_{\alpha 2}) + \mathbf{b}^t(\mathbf{B}_1\mathbf{v}_{\alpha 2}); \mathbf{a}^t(\mathbf{A}_2\mathbf{v}_{\alpha 3}) + \mathbf{b}^t(\mathbf{B}_2\mathbf{v}_{\alpha 3})$. Recall that the goal is to decode two desired unknowns (a_1, a_2) out of three equations including four unknowns (a_1, a_2, b_1, b_2). To achieve this goal, we need

$$\mathsf{rank}\left(\begin{bmatrix}(\mathbf{A}_1\mathbf{v}_{\alpha 2})^t \\ (\mathbf{A}_2\mathbf{v}_{\alpha 3})^t\end{bmatrix}\right) = 2; \quad \mathsf{rank}\left(\begin{bmatrix}\mathbf{v}_{\alpha 1}^t \\ (\mathbf{B}_1\mathbf{v}_{\alpha 2})^t \\ (\mathbf{B}_2\mathbf{v}_{\alpha 3})^t\end{bmatrix}\right) = 1. \tag{4}$$

The second condition can be met by setting $\mathbf{v}_{\alpha 2} = \mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1}$ and $\mathbf{v}_{\alpha 3} = \mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}$. This choice forces the interference space to be aligned into a one-dimensional linear subspace. The alignment of the interference into a single dimension ensures that three equations are sufficient to resolve the two desired unknowns. With this setting, the first condition now becomes

$$\mathsf{rank}\left(\begin{bmatrix}\mathbf{A}_1\mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1} & \mathbf{A}_2\mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}\end{bmatrix}\right) = 2. \tag{5}$$

We can satisfy this condition by carefully choosing $\mathbf{A}_i$s and $\mathbf{B}_i$s.

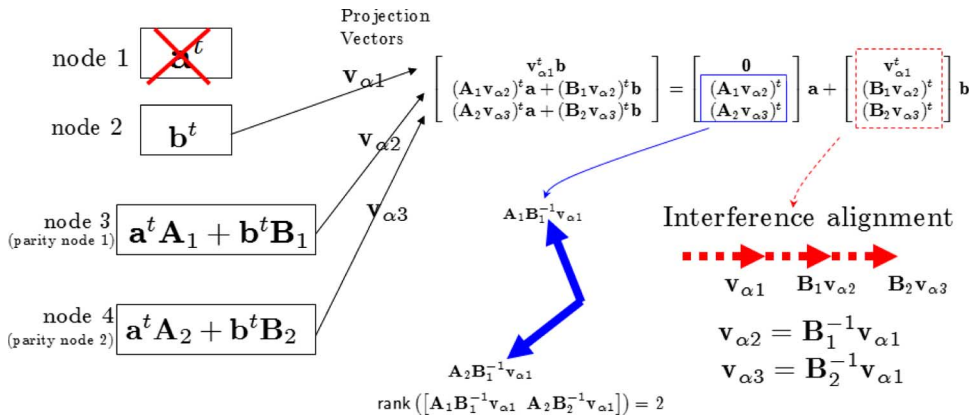

Fig. 2. Interference alignment for a $(4, 2, 3)$ exact-repair MDS code. The choice of $\mathbf{v}_{\alpha 2} = \mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1}$ and $\mathbf{v}_{\alpha 3} = \mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}$ enables achieving interference alignment, thus allowing us to decode the desired signals $\mathbf{a}$.

Connection to the wireless systems: The technique of interference alignment was developed in the context of wireless systems—in particular in the wireless interference and X channels [20]–[22] (See [23] for a tutorial on the same). Broadly speaking, interference alignment is the concept that the interfering signals occupy overlapping dimensions, whereas the desired signal remains separable from the overlapped interference. Since interference occurs naturally in wireless systems due to the broadcast nature of the medium, it has been explored broadly in the context of wireless communication systems.

The applicability of interference alignment in our paper stems from an analogue between the wireless setting and the storage setting. Here, we describe the connection between these two settings. At a high level, in a wireless setting, each receiver gets linear combinations of the transmitted signals including both desired signals and interfering signals. The coefficients as to how the desired and interfering signals are combined together are determined by the channel coefficients. In an (n, k) storage system, with linear coding, each of the parity nodes store linear combinations of the original data ($\mathbf{a}$, $\mathbf{b}$ in the previous example). When a node, say node 1, fails, the goal in this storage setting is to recover the failed node ($\mathbf{a}$ in the example above) from the parity nodes. Note however that the parity nodes store linear combinations of node 1—the desired signal—with the remaining $k-1$ systematic components of the code—the interferers. The coefficients as to how the desired signal is mixed with the interference are determined by the coding coefficients. There is hence a parallel[2] between channel coefficients (in the wireless setting) and the coding coefficients (in the storage setting).

In the wireless setting, interference alignment enables efficiency by reducing the effect of the interference at a receiver and hence freeing up greater number of dimensions for the desired signal. Such interference alignment is enabled by carefully choosing *beamforming vectors* based on the channel gain matrices. In the distributed storage setting, interference alignment reduced the effect of the $(k-1)$ interfering signals at repair center, enabling downloading of a fewer number of linear combinations of the interferers to cancel the interference, hence reducing the repair bandwidth. For instance, in the $(4, 2, 3)$ code example earlier, when node 1 fails, interference alignment enables cancelation of $\mathbf{b}$ through downloading of one scalar $\mathbf{v}_{\alpha 1}^t\mathbf{b}_1$; the naive solution which does not align would download both scalars associated with $\mathbf{b}$. In general, such interference alignment is enabled by carefully choosing the repair vectors based on the code-generator matrices. Therefore, there is a parallel between the repair combining vectors in the distributed storage setting and transmit beamforming vectors in the wireless setting.

To see this connection in a concrete setting, we turn to the $(4, 2, 3)$ code example described earlier. Observe the three equations shown in Fig. 2

$$\underbrace{\begin{bmatrix}\mathbf{0}\\(\mathbf{A}_1\mathbf{v}_{\alpha 2})^t\\(\mathbf{A}_2\mathbf{v}_{\alpha 3})^t\end{bmatrix}\mathbf{a}}_{\text{desired signals}} + \underbrace{\begin{bmatrix}\mathbf{v}_{\alpha 1}^t\\(\mathbf{B}_1\mathbf{v}_{\alpha 2})^t\\(\mathbf{B}_2\mathbf{v}_{\alpha 3})^t\end{bmatrix}\mathbf{b}}_{\text{interference}}.$$

Notice that the goal of repair is to reconstruct $\mathbf{a}$. Separating into two parts, we can relate this repair problem to the wireless interference channel problem wherein a subset of the information needs to be decoded in the presence of interference. Notice the following analogy for the terms of $\mathbf{A}_1$ and $\mathbf{v}_{\alpha 2}$:

	Storage Repair	Wireless Problem
$\mathbf{A}_1$:	Encoding Submatix	Wireless Channel
$\mathbf{v}_{\alpha 2}$:	Projection Vector	Beamforming Vector.

The matrix $\mathbf{A}_1$ and vector $\mathbf{v}_{\alpha 2}$ correspond, respectively, to the channel matrix and beamforming vector in the wireless problem.

The connection indeed lies at the heart of our solution. As we will see in the next section, the previous strategy is not generalizable for arbitrary values of (n, k, d, r) because of the need for *simultaneous* interference alignment across many coding submatrices. The technique that is of central importance in solving this problem is the *asymptotic interference alignment* technique introduced in [20]. We next describe our approach, first in the

[2]Note that one significant difference between the wireless and storage settings is that signaling in the former setting happens over real/complex field, whereas in the latter setting, the codes are over finite fields. As we will see later on, for the purpose of this paper, operating over sufficiently large finite fields dissolves this difference.

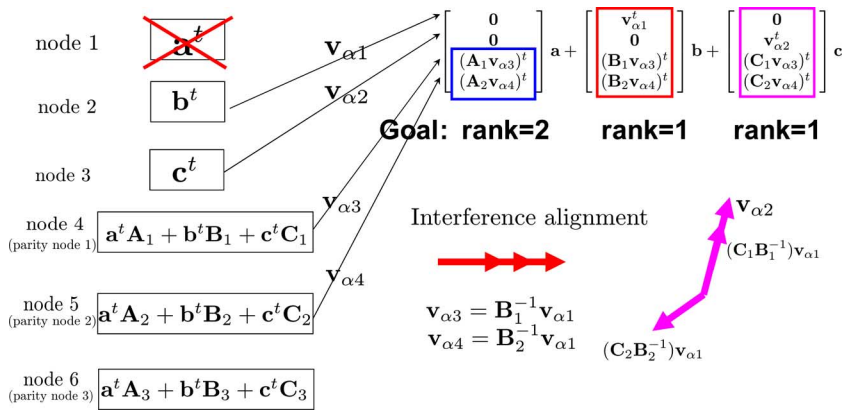

Fig. 3. Difficulty of achieving interference alignment for a scalar linear $(n=6,k=3,d=4,r=1)$ code.

context of a specific example in Section III and later for general values of (n,k,d,r) in Section IV.

Remark 2 (Benefits of the Storage-Code Design Flexibility): Before proceeding, as a side note, we emphasize the great potential to make use of the storage-code design flexibility in achieving interference alignment. In addition to previous work [7], [8] which exploited this connection, [24]–[28] which were published following our study show this potential as well. In particular, Cadambe *et al.* [27] exploited the concept of *subspace interference alignment* [29] as well as the storage-code design flexibility, thereby developing a practical code construction for optimal repair of some storage systems. □

III. Asymptotic Interference Alignment: $(6,3,4)$ Repair MDS Code

The main goal of this paper is to develop a solution framework for optimal repair based on the asymptotic interference alignment scheme of [20]. In general, our solution framework covers all feasible values of (n,k,d,r). This contrasts the scalar-linear code-based framework in [7] and [8] which covers a subset of all feasible values through a deterministic code construction with finite alphabet size. In contrast, here we target only the existence of exact-repair codes without specifying constructions. This allows for a simpler characterization of the solution space for the entire range of admissible repair code parameters. For ease of exposition, we first focus on the specific scenario of $(n=6,\ k=3,\ d=4,\ r=1)$—the simplest case that was not resolved in [8] and [13]. This example scenario will enable us to present all the relevant ideas, and is a representative of the general case. We discuss this special case in detail here and later provide the generalization in Section IV. We will begin by focusing on repair of a failed systematic node. Parity node repair will be dealt with later in the section.

We start by examining the insufficiency of the approach of the previous section. For $n=6,k=3,d=4,r=1$, achieving interference alignment for exact repair turns out to be more complex than the case of $k=2$. Fig. 3 illustrates this difficulty through the example of repairing node 1 for a $(6,3,4)$ code. In accordance with the $(4,2)$ code example in Fig. 2, we choose the total amount of data in the storage system to be $M=6$. Note that each node stores $M/k=2$ equations over a field $\mathbb{F}_q$ where q will be specified soon. Along the lines of the previous section, suppose that we use scalar linear codes, i.e., we download a scalar from each node. We define $\mathbf{a}=(a_1,a_2)^t$, $\mathbf{b}=(b_1,b_2)^t$ and $\mathbf{c}=(c_1,c_2)^t$; 2-by-2 encoding submatrices of $\mathbf{A}_i$, $\mathbf{B}_i$, and $\mathbf{C}_i$ (for $i=1,2,3$); and two-dimensional projection vectors $\mathbf{v}_{\alpha i}$s.

Suppose that survivor nodes $(2,3,4,5)$ participate in exact repair of node 1. We then get the following $d=4$ linear mixtures:

$$\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ (\mathbf{A}_1\mathbf{v}_{\alpha3})^t\\ (\mathbf{A}_2\mathbf{v}_{\alpha4})^t\end{bmatrix}\mathbf{a}+\begin{bmatrix}\mathbf{v}_{\alpha1}^t\\ \mathbf{0}\\ (\mathbf{B}_1\mathbf{v}_{\alpha3})^t\\ (\mathbf{B}_2\mathbf{v}_{\alpha4})^t\end{bmatrix}\mathbf{b}+\begin{bmatrix}\mathbf{0}\\ \mathbf{v}_{\alpha2}^t\\ (\mathbf{C}_1\mathbf{v}_{\alpha3})^t\\ (\mathbf{C}_2\mathbf{v}_{\alpha4})^t\end{bmatrix}\mathbf{c}.$$

In order to successfully recover the two components of $\mathbf{a}$ from the four downloaded equations, the matrices associated with $\mathbf{b}$ and $\mathbf{c}$ should have rank 1, respectively, while the matrix associated with $\mathbf{a}$ should have full rank of 2. In accordance with the $(4,2)$ code example in Fig. 2, if one were to set $\mathbf{v}_{\alpha3}=\mathbf{B}_1^{-1}\mathbf{v}_{\alpha1}$ and $\mathbf{v}_{\alpha4}=\mathbf{B}_2^{-1}\mathbf{v}_{\alpha1}$, then it is possible to achieve interference alignment with respect to $\mathbf{b}$ and reduce the corresponding rank to 1.

However, this choice also specifies the interference space of $\mathbf{c}$. If the $\mathbf{B}_i$s and $\mathbf{C}_i$s are not designed judiciously, interference alignment is not guaranteed for $\mathbf{c}$. Hence, it is not evident how to achieve interference alignment at the same time. This case $(n=6,\ k=4,\ d=3,\ r=1)$ is later solved in [9] through a judicious construction of matrices (referred to in the reference as the product matrix construction). However, the solution of [9] is a scalar code that is not generalizable for all feasible choices of (n,k,d,r). In fact, as demonstrated in [7], scalar linear codes, in general, involve a larger repair bandwidth as compared to the cut-set bound. We will present a *vector* coding approach to resolve the case of $(n=6,\ k=3,\ d=4,\ r=1)$. Later, in Section IV, we will show how our approach is generalizable.

To address a similar simultaneous interference alignment problem in wireless interference channels, Cadambe and Jafar [20] invoked the idea of *symbol extensions*—the notion that multiple symbols can be grouped together and viewed as a vector. By coding jointly over the components of the vector, the reference could achieve simultaneous interference alignment in the wireless context. Here, based on the analogy between the interference channel and the repair context established in the previous section, we invoke the idea of vector coding in the storage context. In vector linear codes, we allow M to

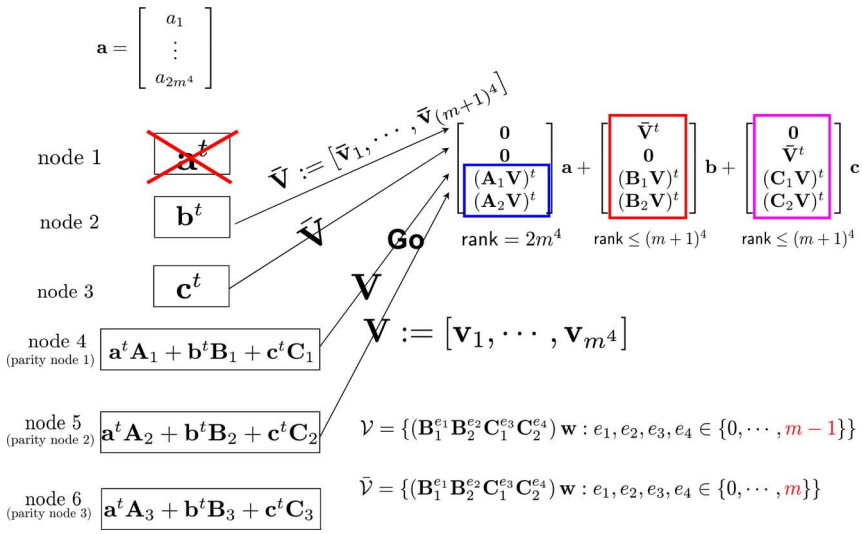

Fig. 4. Illustration of exact repair of systematic node 1.

be a larger parameter of choice,[3] so that each node stores an $M/k = M/3$ dimensional vector over the field $\mathbb{F}_q$. The field size will be (implicitly) specified later in this section. The size of the vector is analogous to the size of the symbol extension used in the interference channel.

Fig. 4 illustrates exact repair of systematic node 1. Drawing parallels from [20], each node stores a $2m^N$-dimensional vector, where m is an arbitrarily large positive integer and the exponent N is carefully chosen depending on code parameters. Specifically

$$N = (k-1)(d-k+1). \tag{6}$$

This choice of N and the form of $2m^N$ are closely related to the scheme to be described in the sequel. In this example, $N = 4$. Note that storage node contains a $2m^4$-dimensional vector, e.g., $\mathbf{a}^t = (a_1, \ldots, a_{2m^4})$, where a_i indicates the ith component of the vector. Now with this vectorization, we show a repair strategy that downloads an m^4-dimensional vector from each of nodes $(4, 5)$ and an $(m+1)^4$-dimensional vector from each of nodes $(2, 3)$ to repair node 1. With this strategy, and noting that $M = 6m^4$, we achieve

$$\lim_{M\to\infty} \frac{B_{1,4}}{M} = \lim_{m\to\infty} \frac{2(m+1)^4 + 2m^4}{6m^4} = \frac{2}{3}$$

as desired according to Theorem 1. Note that since we need $m \to \infty$ or equivalently $M \to \infty$, the cut-set lower bound is achieved in the limit of arbitrarily large file size. Before we describe the repair strategy, let us briefly examine the file-size requirements.

Remark 3 (File-Size Requirements): Consider the case of $m = 1$. In this case, $M = 6$. The repair bandwidth is $2m^4 + 2(m+1)^4 = 34$. This repair bandwidth of 34 is larger than that of the trivial approach of downloading the entire file of $M = 6$.

[3]Note that for sufficiently large file size, M is indeed a parameter of choice. This is because for sufficiently large file size, the file can be split into multiple blocks, each of size M, and coding can be done separately over each of these blocks.

However, as m increases, the repair bandwidth reduces and for $m \geq 6$, the repair bandwidth for our strategy is smaller than that for the trivial approach. In particular, for our strategy, the repair bandwidth reduces with the file size M, as $\frac{2}{3} + O\left(\frac{1}{\sqrt[4]{M}}\right)$. □

Our solution works by achieving the following three objectives (See Fig. 4).

1) *Interference alignment*: The rank of the interference matrix corresponding to $\mathbf{b}$ and the interference matrix corresponding to $\mathbf{c}$ are restricted to $(m+1)^4$. Such simultaneous alignment w.r.t. both $\mathbf{b}$ and $\mathbf{c}$ enables successful interference cancelation by just downloading $(m+1)^4$ linear combinations from each of nodes 2 and 3.
2) *Recovery of desired components*: The matrix corresponding to $\mathbf{a}$ has full rank of $2m^4$, thus enabling reconstruction.
3) *MDS property*: The aforementioned two properties ensure successful reconstruction for a single-node failure, with the desired repair bandwidth. Along with this, we also need to ensure the MDS property that the original information $\mathbf{a}, \mathbf{b}, \mathbf{c}$ can be reconstructed from any three nodes in the system.

Note that downloading a total of $2m^4 + 2(m+1)^4$ equations from the surviving nodes suffices as long as the first two conditions discussed previously are satisfied. We next describe our solution which achieves this repair bandwidth. For the purpose of this section, the field of operation is assumed to be $\mathbb{F}_q$ where q is a prime number which is chosen to be sufficiently large for purposes to be specified later in this construction.

Design of encoding submatrices: The size of encoding submatrices $(\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i)$ is $2m^4$-by-$2m^4$. We consider *diagonal* encoding submatrices. As pointed out in [20], the diagonal matrix structure ensures a *commutative* property which is central to the interference alignment scheme (to be described shortly)

$$\mathbf{A}_i = \begin{bmatrix} \alpha_i^{(1)} & 0 & \cdots & 0 \\ 0 & \alpha_i^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \alpha_i^{(2m^4)} \end{bmatrix}. \tag{7}$$

Repair strategy: Failed node 1 is exactly repaired through the following steps. Assume that survivor nodes $(2,3,4,5)$ participate in exact repair of node 1: $k-1=2$ systematic nodes and $d-k+1=2$ parity nodes. One can alternatively use one systematic node and three parity nodes for repair instead. This does not fundamentally alter the analysis, and will be discussed in Section III-B. For the time being, assume the previous configuration for the connection: $(k-1)$ systematic nodes and $(d-k+1)$ parity nodes. To describe the repair strategy, we denote by $\mathbf{V}$ the $2\ m^4 \times m^4$ projection matrix used by the parity nodes to project their data. Thus, the data obtained by the repair center from the parity nodes can be written as

$$\begin{aligned}&\text{From parity node 1: } \mathbf{a}^t(\mathbf{A}_1\mathbf{V})+\mathbf{b}^t(\mathbf{B}_1\mathbf{V})+\mathbf{c}^t(\mathbf{C}_1\mathbf{V});\\ &\text{From parity node 2: } \mathbf{a}^t(\mathbf{A}_2\mathbf{V})+\mathbf{b}^t(\mathbf{B}_2\mathbf{V})+\mathbf{c}^t(\mathbf{C}_2\mathbf{V}).\end{aligned}$$

The repair center also downloads data from systematic nodes 2 and 3. To cancel the effect of $\mathbf{b}$ and $\mathbf{c}$ from the above data, the repair center has to download at least $\operatorname{rank}([\mathbf{B}_1\mathbf{V},\mathbf{B}_2\mathbf{V}])$ equations from node 2 and at least $\operatorname{rank}([\mathbf{C}_1\mathbf{V},\mathbf{C}_2\mathbf{V}])$ equations from node 3. The goal of the repair strategy is to align $\mathbf{B}_1\mathbf{V}$ with $\mathbf{B}_2\mathbf{V}$ and simultaneously align $\mathbf{C}_1\mathbf{V}$ with $\mathbf{C}_2\mathbf{V}$ so that the number of equations is minimized. In particular, we will design $\mathbf{V}$ so that

$$\begin{aligned}&\mathsf{rank}([\mathbf{B}_1\mathbf{V}\ \ \mathbf{B}_2\mathbf{V}])\le(m+1)^4\\ &\mathsf{rank}([\mathbf{C}_1\mathbf{V}\ \ \mathbf{C}_2\mathbf{V}])\le(m+1)^4\\ &\mathsf{rank}([\mathbf{A}_1\mathbf{V}\ \ \mathbf{A}_2\mathbf{V}])=2m^4.\end{aligned}$$

The first two conditions discussed earlier imply that downloading $(m+1)^4$ scalars from each of systematic nodes 2 and 3 suffices for canceling the effect of $\mathbf{b}$ and $\mathbf{c}$ from the equations downloaded from the parity nodes. The last condition ensures reconstruction of $\mathbf{a}$. This leads to a repair bandwidth of $2m^4+2(m+1)^4$ as required.

Next, we shall describe the repair strategy. We begin with the case of $m=1$, where $\mathbf{A}_i$ is just a 2×1 vector. Each of the two parity survivor matrices participating in repair project their data along a single 2×1 vector $\mathbf{V}=\mathbf{w}=[1\ \ 1]^T$ so that the data received by the repair center is

$$\begin{aligned}&\text{From parity node 1: } \mathbf{a}^t(\mathbf{A}_1\mathbf{w})+\mathbf{b}^t(\mathbf{B}_1\mathbf{w})+\mathbf{c}^t(\mathbf{C}_1\mathbf{w});\\ &\text{From parity node 2: } \mathbf{a}^t(\mathbf{A}_2\mathbf{w})+\mathbf{b}^t(\mathbf{B}_2\mathbf{w})+\mathbf{c}^t(\mathbf{C}_2\mathbf{w})\end{aligned}$$

where $\mathbf{w}$ is equal to $[1\ \ 1]^T$. The receiver hence gets two scalars. In general, note that $\mathbf{B}_1\mathbf{w}$ and $\mathbf{B}_2\mathbf{w}$ can be linearly independent since they lie in at two-dimensional space. Thus, the rank of $[\mathbf{B}_1\mathbf{w}\ \ \mathbf{B}_2\mathbf{w}]$ can be 2 since there is no any overlap (alignment) between $\mathbf{B}_1\mathbf{V}$ and $\mathbf{B}_2\mathbf{V}$. If this is the case, all the data stored in systematic node 2 has to be downloaded for repair. Similarly, for the case of $m=1$, all the data stored in systematic node 3 has to be downloaded for repair. Now, we increase the extent of alignment for $m=2$. In this case $\mathbf{a},\mathbf{b},\mathbf{c}$ are all $2m^4=32$-dimensional vectors. Each of the two parity nodes projects its data into the following $m^4=16$-dimensional vector whose vectors are indexed as follows:

$$\mathbf{V}:=[\mathbf{v}_1,\ldots,\mathbf{v}_{16}]\in\mathbb{F}_q^{32\times 16} \tag{8}$$

where $\mathbf{V}$ is chosen to be have the column vectors from the set

$$\mathcal{V}=\{\mathbf{B}_1^{e_1}\mathbf{B}_2^{e_2}\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4}\mathbf{w}:e_1,e_2,e_3,e_4\in\{0,1\}\}.$$

The previous matrix has 16 columns as required. Now, observe that the columns of $\mathbf{B}_1\mathbf{V}$ can be written as

$$\{\mathbf{B}_1^{e_1}\mathbf{B}_2^{e_2}\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4}\mathbf{w}:e_2,e_3,e_4\in\{0,1\},e_1\in\{1,2\}\}.$$

Similarly, the columns of $\mathbf{B}_2\mathbf{V}$ can be written as

$$\{\mathbf{B}_1^{e_1}\mathbf{B}_2^{e_2}\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4}\mathbf{w}:e_1,e_3,e_4\in\{0,1\},e_2\in\{1,2\}\}.$$

Notice that the following set of column vectors are common to $\mathbf{B}_1\mathbf{V}$ and $\mathbf{B}_2\mathbf{V}$:

$$\{\mathbf{B}_1\mathbf{B}_2\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4}\mathbf{w}:e_3,e_4\in\{0,1\}\}.$$

In other words, we achieve partial alignment between $\mathbf{B}_1\mathbf{V}$ and $\mathbf{B}_2\mathbf{V}$, as 4 of the 16 column vectors are common. A similar overlap occurs between $\mathbf{C}_1\mathbf{V}$ and $\mathbf{C}_2\mathbf{V}$. Finally, note that each of the columns of $\mathbf{B}_1\mathbf{V},\mathbf{B}_2\mathbf{V},\mathbf{C}_1\mathbf{V},\mathbf{C}_2\mathbf{V}$ is contained in the set

$$\overline{\mathcal{V}}=\{\mathbf{B}_1\mathbf{B}_2\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4}\mathbf{w}:(e_1,e_2,e_3,e_4)\in\{0,1,2\}^4\} \tag{9}$$

whose cardinality is $|\mathcal{V}|=(m+1)^4=3^4$. Hence, using the previous set of columns as a projection matrix for nodes 2 and 3, we can achieve a repair bandwidth of $2\times(2+1)^4+2\times 2^4$.

We next generalize the previous approach. We intend to show that our construction gives

$$\begin{aligned}&\operatorname{rank}([\mathbf{B}_1\mathbf{V}\ \ \mathbf{B}_2\mathbf{V}])=O(m^4)=\operatorname{rank}([\mathbf{C}_1\mathbf{V}\ \ \mathbf{C}_2\mathbf{V}])\\ &\operatorname{rank}([\mathbf{A}_1\mathbf{V}\ \ \mathbf{A}_2\mathbf{V}])=2m^4.\end{aligned}$$

To ensure the previous rank constraints, each of the two parity survivor nodes participating in repair projects its data with the following projection matrix:

$$\mathbf{V}:=[\mathbf{v}_1,\ldots,\mathbf{v}_{m^4}]\in\mathbb{F}_q^{2m^4\times m^4} \tag{10}$$

where $\mathbf{v}_i\in\mathcal{V}$. The set $\mathcal{V}$ is defined as

$$\mathcal{V}:=\{(\mathbf{B}_1^{e_1}\mathbf{B}_2^{e_2}\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4})\mathbf{w}:e_1,e_2,e_3,e_4\in\{0,\ldots,m-1\}\} \tag{11}$$

where $\mathbf{w}=[1,\ldots,1]^t$.

Remark 4: Alternatively, the entries of the $\mathbf{w}$ can be chosen randomly and independently from the field. See [30] for example. This choice makes little difference to the proofs in our paper. □

Note that $|\mathcal{V}|\le m^4$. The vector $\mathbf{v}_i$ maps to a different sequence of (e_1,e_2,e_3,e_4). For example, we can map

$$\begin{aligned}&\mathbf{v}_1=\mathbf{w}\\ &\mathbf{v}_2=\mathbf{C}_2\mathbf{w}\\ &\ \ \vdots\\ &\mathbf{v}_{m^4-2}=\mathbf{B}_1^{m-1}\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-3}\mathbf{w}\\ &\mathbf{v}_{m^4-1}=\mathbf{B}_1^{m-1}\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-2}\mathbf{w}\\ &\mathbf{v}_{m^4}=\mathbf{B}_1^{m-1}\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-1}\mathbf{w}.\end{aligned} \tag{12}$$

Consider the equations downloaded from parity nodes 1 and 2 (nodes 4 and 5). Note that $\mathbf{B}_1\mathbf{V}$ contains the following column vectors:

$$\begin{aligned}
&\mathbf{B}_1\mathbf{v}_1 = \mathbf{B}_1\mathbf{w},\ \mathbf{B}_1\mathbf{v}_2 = \mathbf{B}_1\mathbf{C}_2\mathbf{w},\ \mathbf{B}_1\mathbf{v}_3 = \mathbf{B}_1\mathbf{C}_2^2\mathbf{w},\ \cdots,\\
&\mathbf{B}_1\mathbf{v}_{m^4-2} = \mathbf{B}_1^m\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-3}\mathbf{w}\\
&\mathbf{B}_1\mathbf{v}_{m^4-1} = \mathbf{B}_1^m\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-2}\mathbf{w}\\
&\mathbf{B}_1\mathbf{v}_{m^4} = \mathbf{B}_1^m\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-1}\mathbf{w}.
\end{aligned}$$

An important observation is that any column vector $\mathbf{B}_1\mathbf{v}_i$ is an element of $\bar{\mathcal{V}}$ defined as

$$\bar{\mathcal{V}} := \left\{\left(\mathbf{B}_1^{e_1}\mathbf{B}_2^{e_2}\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4}\right)\mathbf{w} : e_1, e_2, e_3, e_4 \in \{0, \ldots, m\}\right\}. \tag{13}$$

Similarly, any column vector in $\mathbf{B}_2\mathbf{V}$, $\mathbf{C}_1\mathbf{V}$ or $\mathbf{C}_2\mathbf{V}$ is an element of $\bar{\mathcal{V}}$. This implies that $[\mathbf{B}_1\mathbf{V}, \mathbf{B}_2\mathbf{V}] \in \mathbb{F}_q^{2m^4\times 2m^4}$ is a rank-deficient matrix, i.e., $\mathsf{rank}[\mathbf{B}_1\mathbf{V}, \mathbf{B}_2\mathbf{V}] \leq |\bar{\mathcal{V}}| = (m+1)^4$. Similarly, $\mathsf{rank}[\mathbf{C}_1\mathbf{V}, \mathbf{C}_2\mathbf{V}] \leq (m+1)^4$. This enables simultaneous interference alignment although the same projection matrix $\mathbf{V}$ is used for $\mathbf{b}$ and $\mathbf{c}$. This observation motivates the systematic survivor nodes to project their data using the following projection matrix:

$$\bar{\mathbf{V}} := [\bar{\mathbf{v}}_1, \cdots, \bar{\mathbf{v}}_{(m+1)^4}] \in \mathbb{F}_q^{2m^4\times(m+1)^4} \tag{14}$$

where $\bar{\mathbf{v}}_i \in \bar{\mathcal{V}}$ and is mapped to a difference sequence of (e_1, e_2, e_3, e_4) as in (12). We can then guarantee that

$$\begin{aligned}
&\mathsf{colspan}[\mathbf{B}_1\mathbf{V}, \mathbf{B}_2\mathbf{V}] \subset \mathsf{colspan}[\bar{\mathbf{V}}]\\
&\mathsf{colspan}[\mathbf{C}_1\mathbf{V}, \mathbf{C}_2\mathbf{V}] \subset \mathsf{colspan}[\bar{\mathbf{V}}].
\end{aligned} \tag{15}$$

Hence, using $\mathbf{b}^t\bar{\mathbf{V}}$ and $\mathbf{c}^t\bar{\mathbf{V}}$ (downloaded from systematic survivor nodes), we can completely remove any interference $(\mathbf{b}^t(\mathbf{B}_1\mathbf{V}), \mathbf{b}^t(\mathbf{B}_2\mathbf{V}), \mathbf{c}^t(\mathbf{C}_1\mathbf{V}), \mathbf{c}^t(\mathbf{C}_2\mathbf{V}))$, thereby obtaining $\mathbf{a}^t[\mathbf{A}_1\mathbf{V}, \mathbf{A}_2\mathbf{V}]$. Put simply, we have satisfied the interference alignment condition which is one of the three objectives stated previously. To successfully reconstruct $\mathbf{a}$, we need

$$\mathsf{rank}[\mathbf{A}_1\mathbf{V}, \mathbf{A}_2\mathbf{V}] = 2m^4. \tag{16}$$

In other words, $[\mathbf{A}_1\mathbf{V}, \mathbf{A}_2\mathbf{V}]$ must have full rank. Finally, to complete the proof, we also need the MDS property—the last objective. The proofs of (16) and the MDS property are based on the Schwartz–Zippel Lemma [31]. Specifically, we show that there exist diagonal encoding submatrices $\mathbf{A}_i, \mathbf{B}_i, \text{and } \mathbf{C}_i$ such that these two properties are satisfied. The argument is as follows.

1) Consider (16). In the matrix on the left-hand side, notice that the design of $\mathbf{V}$ does not depend on $\mathbf{A}_1$ and $\mathbf{A}_2$. Therefore, each entry of the matrix is a different monomial in the diagonal entries of the encoding submatrices $\mathbf{A}_i, \mathbf{B}_i, \text{and } \mathbf{C}_i$. Based on this observation, it can be shown (see [30, Lemma 1]) that if the field size is large enough, the determinant of the matrix in (16) is a nonzero polynomial in the diagonal entries of $\mathbf{A}_i,\ \mathbf{B}_i, \text{and } \mathbf{C}_i,\ i = 1, 2, 3$. Let us denote this polynomial by $g(\cdot)$. Note that for (16) to be satisfied, we need $g(\cdot)$ to evaluate to a nonzero value in the field.
2) The MDS property means that the code must be able to tolerate the failure of any three storage nodes in the system. Equivalently, any set of three nodes in the system, when interpreted as equations in $\mathbf{a}, \mathbf{b}, \text{and } \mathbf{c}$ must have full rank of $M = 6m^4$ and hence the matrix representing these equations, must have a nonzero determinant. Note that there are $\binom{6}{3}$ possible sets of three nodes in the storage system. The MDS property is therefore equivalent to showing that $\binom{6}{3} = 20$ determinants are all nonzero. Note that each determinant is a polynomial in the entries of the encoding submatrices. In the next section, we will show in the more general context of arbitrary n and k that even with diagonal coding submatrices chosen here, all these polynomials are nonzero as long as the field size is sufficiently large. To summarize, we show that as long as the field size is sufficiently large, the MDS property corresponds to 20 nonzero polynomials in the entries of the diagonal elements of $\mathbf{A}_i, \mathbf{B}_i, \text{and } \mathbf{C}_i$ each evaluating to a nonzero value. We will denote these polynomials by $f_1, f_2, \ldots, f_{20}$.

From the above, we only need to show that there exists a realization of diagonal entries for the coding submatrices so that the polynomials $f_1(\cdot), f_2(\cdot), \ldots, f_{20}(\cdot)$ and the polynomial $g(\cdot)$ evaluate to a nonzero value in the field. Showing this will ensure the existence of codes satisfying the final two objectives—the MDS property and the recovery of the desired components—to complete the proof. To do so, we invoke the Schwartz–Zippel Lemma to product polynomial $g \cdot f_1 \cdot f_2 \cdots f_{20}$ which is a nonzero polynomial, by virtue of each of its factors being nonzero polynomials. Over a sufficiently large field, the lemma guarantees, via a probabilistic argument, the existence of diagonal matrices $\mathbf{A}_i, \mathbf{B}_i, \text{and } \mathbf{C}_i$ so that this product polynomial and hence each of its factors evaluate to some nonzero value and hence completes the proof.

A. Parity Node Repair

So far, we have discussed an achievable scheme for repairing a systematic node. The codes $\mathbf{A}_i, \mathbf{B}_i, \text{and } \mathbf{C}_i$ constructed here can also be used to create an optimal repair strategy for a failed parity node in the same manner. The key idea is the following. In an MDS code, any k nodes are information equivalent to the original information in a system and therefore can be interpreted as k systematic nodes. The data stored in the remaining $(n-k)$ nodes are functions of these k nodes and can therefore be interpreted as parity nodes. Hence, through a remapping of the nodes and an appropriate transformation, a parity node of a code can be interpreted as a systematic node of a virtual alternate code—a parity node failure can therefore be interpreted as a systematic node failure under a virtual alternate code. Specifically, for linear MDS codes, by using a change of basis, a parity node in the original code can be virtually interpreted as a systematic node of a virtual alternate code. As long as the alternate code shares properties similar to the original code (diagonal encoding submatrices, etc.), the ideas of systematic node repair can be applied to parity node repair as well. Let us crystallize this idea in the context of an example. Suppose that a parity node, say node 6, fails. We can now remap the nodes so that

this failed node is systematic node $\mathbf{c}'^t$. Therefore, in this alternate virtual code, we have three systematic nodes $\mathbf{a}', \mathbf{b}', \text{and } \mathbf{c}'$:

$$\begin{aligned}&\text{Node 1: } \mathbf{a}'^t = \mathbf{a}^t\\ &\text{Node 2: } \mathbf{b}'^t = \mathbf{b}^t\\ &\text{Node 3: } \mathbf{c}'^t = \mathbf{a}^t\mathbf{A}_3 + \mathbf{b}^t\mathbf{B}_3 + \mathbf{c}^t\mathbf{C}_3.\end{aligned} \tag{17}$$

With the remaping, $\mathbf{c}^t$ is now a parity node. The three parity nodes can be expressed as

$$\begin{aligned}&\text{Node 4: } \mathbf{a}'^t\{\mathbf{A}_1 + \mathbf{A}_3(\mathbf{C}_3)^{-1}\mathbf{C}_1\}\\ &\qquad +\mathbf{b}'^t\{\mathbf{B}_1 + \mathbf{B}_3(\mathbf{C}_3)^{-1}\mathbf{C}_1\} + \mathbf{c}'^t(\mathbf{C}_3)^{-1}\mathbf{C}_1\\ &\text{Node 5: } \mathbf{a}'^t\{\mathbf{A}_2 + \mathbf{A}_3(\mathbf{C}_3)^{-1}\mathbf{C}_2\}\\ &\qquad +\mathbf{b}'^t\{\mathbf{B}_2 + \mathbf{B}_3(\mathbf{C}_3)^{-1}\mathbf{C}_2\} + \mathbf{c}'^t(\mathbf{C}_3)^{-1}\mathbf{C}_2.\\ &\text{Node 6: } \mathbf{a}'^t\mathbf{A}_3(\mathbf{C}_3)^{-1} + \mathbf{b}'^t\mathbf{B}_3(\mathbf{C}_3)^{-1} + \mathbf{c}'^t(\mathbf{C}_3)^{-1}.\end{aligned} \tag{18}$$

Let us denote the ith parity node (i.e., node $i+k = i+3$) as $\mathbf{a}'^t\mathbf{A}'_i + \mathbf{b}'^t\mathbf{B}'_i + \mathbf{c}'^t\mathbf{C}'_i$ so that for example $\mathbf{A}'_1 = \mathbf{A}_1 + \mathbf{A}_3(\mathbf{C}_3)^{-1}\mathbf{C}_1$ and so on. From the previous expressions, all the encoding submatrices $\mathbf{A}'_i, \mathbf{B}'_i, \text{and } \mathbf{C}'_i$ are diagonal. This is because the sum, product, and inverse of two diagonal matrices are diagonal. The diagonal property ensures that the encoding submatrices commute even in this virtual code. This means that by picking the repair vectors in a manner analogous to (11) and (13), we can satisfy a corresponding condition analogous to (15). Using an argument similar to the previous section, it can be shown that the desired components can also be completely recovered. The detailed proof is omitted here to avoid tedious notation.

Remark 5 (Field Size Requirements): Note that a failure of each node is with respect to a different polynomial which has to be shown to evaluate to a nonzero number. Also note that there exists a q_0 such that over field $\mathbb{F}_q$, for $q > q_0$ each of these polynomials is a nonzero polynomial. This is because for sufficiently large q_0, the product of these polynomials is nonzero over the field (due to the Schwartz–Zippel Lemma). To use the lemma, we assume that the field size is large enough to be much greater than the degree of this composite polynomial. It is worth noting that the degree of the product polynomial depends on n, k and the repair bandwidth that we intend to achieve (i.e., m). As m increases, the repair bandwidth approaches optimality, the degree of the product polynomial grows, and hence the field size required for application of the Schwartz–Zippel Lemma also grows.

B. Participation of Arbitrary d Nodes for Exact Repair

We have considered a somewhat restrictive configuration for exact repair: connecting to surviving $(k-1)$ systematic nodes and to other $(d-k+1)$ parity nodes. We now consider more general connection configurations. For example, consider the case when node 1 fails. Suppose we connect to nodes $(2, 4, 5, 6)$ for exact repair of node 1: one systematic node and three parity nodes. We use the idea similar to that of parity node repair. We remap one parity node to make it look like a systematic node. We then virtually connect to two systematic and to two parity nodes. Specifically, we can remap node 6 with $\mathbf{c}'^t$ and perform conversions similar to (17) and (18). Applying the same procedures as before, we can then guarantee the exact repair of $\mathbf{a}$.

IV. Generalization

We will now prove the achievability of Theorem 2 by generalizing the previous setting to the case where (n, k) is arbitrary; $r \leq \min(k, n-k)$ nodes fail; and $d > k$ nodes are contacted for repair. While this setting is more general, most of the ideas follow from the previous section. We therefore only provide a sketch of the main ideas.

Consider a storage system which stores total data M in an (n, k) MDS code-based distributed storage system. The total data are represented by the $M/k \times k$-dimensional matrix $[\mathbf{a}_1 \ \mathbf{a}_2 \ \cdots \ \mathbf{a}_k]$, where $\mathbf{a}_i$ is an $M/k \times 1$-dimensional vector stored by systematic node $i \in \{1, 2, \ldots, k\}$. Node j ($j \in \{k+1, k+2, \ldots, n\}$ being a parity node) stores the $1 \times M/k$ vector $\mathbf{a}_1^t\mathbf{A}_{j,1} + \mathbf{a}_2^t\mathbf{A}_{j,2} + \cdots + \mathbf{a}_k^t\mathbf{A}_{j,k}$ where $\mathbf{A}_{j,i}$ is an $M/k \times M/k$ square matrix for $i \in \{1, 2, \ldots, k\}$. Because of the systematic structure of the code, we assume that for $j \leq k$:

$$\mathbf{A}_{j,i} = \begin{cases}\mathbf{0}, & j \neq i\\ \mathbf{I}, & j = i\end{cases} \forall i \in \{1, 2, \ldots, k\}. \tag{19}$$

The previous assumption implies that the data stored in node $j \in \{1, 2, \ldots, n\}$ are the $M/k \times 1$ vector $\mathbf{D}_j$ shown as follows:

$$\mathbf{D}_j^t = \sum_{i=1}^{k} \mathbf{a}_i^t\mathbf{A}_{j,i}. \tag{20}$$

Note that the encoding submatrices $\mathbf{A}_{j,i}$ for $j = k+1, k+2, \ldots, n$ are a design choice that defines the storage code coefficients. We need to choose these matrices so that the code is an MDS code, i.e., using any subset of k nodes, the entire $M \times 1$ vector of data must be reconstructible. Thus, we need to ensure that

$$\mathsf{rank}\left(\begin{bmatrix}\mathbf{A}_{j_1,1} & \mathbf{A}_{j_1,2} & \cdots & \mathbf{A}_{j_1,k}\\ \mathbf{A}_{j_2,1} & \mathbf{A}_{j_2,2} & \cdots & \mathbf{A}_{j_2,k}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{A}_{j_k,1} & \mathbf{A}_{j_k,2} & \cdots & \mathbf{A}_{j_k,k}\end{bmatrix}\right) = M \tag{21}$$

for any distinct $j_1, j_2, \ldots, j_k \in \{1, 2, \ldots, n\}$.

Now, suppose that $r \leq \min(k, n-k)$ nodes fail. We consider the case where the r failed nodes are systematic nodes. Later, the scenario will be generalized to the case where the failed nodes can be parity nodes as well. Without loss of generality, we assume that the first r systematic nodes fail. We assume that the repair center connects to the surviving $(k-r)$ systematic nodes and the first $(d+r-k)$ parity nodes so that it connects to a total of d nodes. As usual, the goal is to repair the lost data $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r$. In this case, we set $M = k(r+d-k)m^N$ where $N = (d+r-k)(k-r)$. The goal of the solution will be to download $r(m+1)^N$ equations from each of the $(k-r)$ surviving systematic nodes, and rm^N equations from the $(d+r-k)$ parity nodes that the repair center connects to. Note that as $m \to \infty$, the total repair bandwidth is

$$\begin{aligned}\lim_{M\to\infty}\frac{B_{r,d}}{M} &= \lim_{m\to\infty}\frac{r(k-r)(m+1)^N + r(d+r-k)m^N}{k(d+r-k)m^N}\\ &= \frac{rd}{k(d+r-k)}.\end{aligned}$$

Remark 6: Note from the previous expression that bandwidth reduces with file size as $\frac{rd}{k(d+r-k)} + O\left(\frac{1}{\sqrt[N]{M}}\right)$.

Similar to the previous section, to achieve the above repair bandwidth, we aim for the following three objectives.

1) *Interference alignment*: The interference corresponding to each of $\mathbf{a}_{r+1}, \mathbf{a}_{r+2}, \ldots, \mathbf{a}_k$ is aligned simultaneously so that it can be completely canceled.
2) *Recovery of desired components*: The vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r$ can be regenerated at the repair center.
3) *MDS property*: The code is an (n, k) MDS code, i.e., (21) is satisfied.

Our repair strategy is as follows.

1) From each of $(k-r)$ surviving systematic nodes, the repair center downloads vectors $\mathbf{a}_j^t\bar{\mathbf{V}}$ where $r < j \leq k$ and $\bar{\mathbf{V}} \in \mathbb{F}_q^{M/k \times r(m+1)^N}$. These downloaded vectors contain no information associated with the desired data $\mathbf{a}_1, \ldots, \mathbf{a}_r$ and will be used as side information to cancel interference.
2) From each of the $(d+r-k)$ parity nodes, the repair center downloads vectors of the form $\mathbf{D}_j^t\mathbf{V} = \sum_{i=1}^k \mathbf{a}_i^t\mathbf{A}_{j,i}\mathbf{V}$ where $k < j \leq d+r$ and $\mathbf{V} \in \mathbb{F}_q^{M/k \times rm^N}$. These downloaded vectors contain both desired components and interfering components.

The goal of our solution will be to completely cancel the interference from the latter $(d+r-k)$ sets of vectors using the former $(k-r)$ sets of vectors listed earlier, and then to regenerate the $rM/k = r(d+r-k)m^N$ components of $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r$ using the latter $(d+r-k)$ sets of vectors. In order to completely cancel the interference related to $\mathbf{a}_i$, we will need that $\forall j = k+1, k+2, \ldots, d+r$:

$$\mathsf{colspan}(\mathbf{A}_{j,i}\mathbf{V}) \subseteq \mathsf{colspan}(\bar{\mathbf{V}}), \quad i = r+1, r+2, \ldots, k. \tag{22}$$

Note that the above are the desired relations analogous to (15) in the previous section. The previous condition ensures that the entire interference can be canceled. After interference cancelation, each of the $(d+r-k)$ matrices is of the form $\sum_{i=1}^r \mathbf{a}_i^t\mathbf{A}_{j,i}\mathbf{V}$ for $j = k+1, k+2, \ldots, d+r$. For reconstruction of $\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_r$, we need

$$\mathsf{rank}\left(\begin{bmatrix} \mathbf{A}_{k+1,1}\mathbf{V} & \mathbf{A}_{k+2,1}\mathbf{V} & \cdots & \mathbf{A}_{d+r,1}\mathbf{V} \\ \mathbf{A}_{k+1,2}\mathbf{V} & \mathbf{A}_{k+2,2}\mathbf{V} & \cdots & \mathbf{A}_{d+r,2}\mathbf{V} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k+1,r}\mathbf{V} & \mathbf{A}_{k+2,r}\mathbf{V} & \cdots & \mathbf{A}_{d+r,r}\mathbf{V} \end{bmatrix}\right) = r \cdot \frac{M}{k}. \tag{23}$$

The previous condition ensures that the desired lost data can be reconstructed after interference cancelation.

Thus, we need to design $\mathbf{A}_{j,i}$, $\mathbf{V}$, and $\bar{\mathbf{V}}$ for $j \in \{k+1, k+2, \ldots, n\}$ and $i \in \{1, 2, \ldots, k\}$ such that (21), (22), and (23) are satisfied. We now proceed to describe our construction.

Design of encoding submatrices $\mathbf{A}_{j,i}$: As in Section III, we choose the $M/k \times M/k$-dimensional matrices $\mathbf{A}_{j,i}$ $\forall j = k+1, k+2, \ldots, n$ to be diagonal matrices

$$\mathbf{A}_{j,i} = \begin{bmatrix} \alpha_{j,i}^{(1)} & 0 & \cdots & 0 \\ 0 & \alpha_{j,i}^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_{j,i}^{(\frac{M}{k})} \end{bmatrix}. \tag{24}$$

Design of repair matrices $\mathbf{V}$ and $\bar{\mathbf{V}}$: As in Section III, we choose the set of column vectors of $\mathbf{V}$ and $\bar{\mathbf{V}}$, respectively, from the sets $\mathcal{V}$ and $\bar{\mathcal{V}}$ described as follows:

$$\mathcal{V} = \left\{ \left(\prod_{\substack{j=k+1,\cdots,r+d \\ i=r+1,\cdots,k}} \mathbf{A}_{j,i}^{e_{j,i}} \right) \mathbf{w} : e_{k+1,r+1}, \ldots, e_{r+d,k} \in \{0, 1, \ldots, m-1\} \right\} \tag{25}$$

$$\bar{\mathcal{V}} = \left\{ \left(\prod_{\substack{j=k+1,\cdots,r+d \\ i=r+1,\cdots,k}} \mathbf{A}_{j,i}^{e_{j,i}} \right) \mathbf{w} : e_{k+1,r+1}, \ldots, e_{n,k} \in \{0, 1, \ldots, m\} \right\} \tag{26}$$

where $\mathbf{w}$ denotes $[1 \;\; 1 \;\; \cdots \;\; 1]^t$. It can be verified that with the previous choice of column vectors, $\mathbf{A}_{j,i}\mathbf{V} \subset \bar{\mathbf{V}}$ for $i = r+1, r+2, \ldots, k$, $j = k+1, k+2, \ldots, d+r$, and therefore (22) holds.

Proof of (21) and (23): We have chosen encoding submatrices and repair matrices so that (22) is satisfied. In order to show that the matrices of (21) and (23) have full rank, it is enough to show that their determinants are nonzero. Notice that the determinant of the matrix in the left-hand side of (21) is a polynomial in its entries. Note that there are $\binom{n}{k}$ polynomials of this kind, which can be represented, for $l = 1, 2, \ldots, \binom{n}{k}$, as

$$f_l : \mathcal{A} \to \mathbb{F}_q$$

where

$$\mathcal{A} = \left\{ \alpha_{j,i}^{(l)} : j \in \{k+1, k+2, \ldots, n\} \; i \in \{1, 2, \ldots, k\}, \; l \in \{1, 2, \ldots, (d+r-k)m^N\} \right\}$$

denotes the set of all the diagonal entries of the coding matrices. In the appendix, we show that each of these polynomials is a nonzero polynomial.

Let us now show (23). To show that the square matrix on the left-hand side of the equation has full rank of $\frac{rM}{k}$, we need to show that its determinant is nonzero. Since a determinant is a polynomial function of its entries, the determinant expansion above is a polynomial

$$g : \mathcal{A} \to \mathbb{F}_q.$$

An argument similar to [30, Lemma 1] can be used to show that the polynomial formed by this matrix for our solution is a nonzero polynomial. See also [20, Appendix III]. Thus, the product $f_1(\cdot)f_2(\cdot)\cdots f_{\binom{n}{k}}(\cdot)g(\cdot)$ is a nonzero polynomial of $\mathcal{A}$.

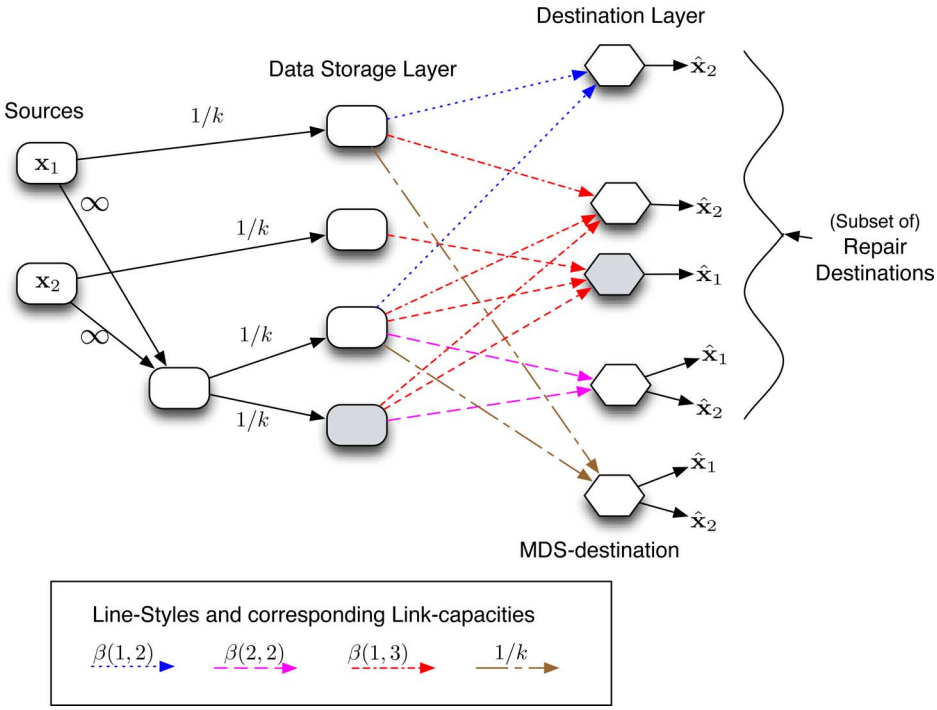

Fig. 5. Multisource nonmulticast network translated from the storage-repair problem.

Using the Schwartz–Zippel Lemma, for sufficiently large q, we have at least one choice of encoding submatrices and repair matrices such that these polynomials evaluate to a nonzero value and therefore a solution exists so that (21) and (23) are satisfied. Thus, we have satisfied all the desired objectives, and complete the proof.

Parity node repair and connecting to arbitrary d nodes: The extension to parity node repair and arbitrary configurations (w.r.t. the connection to survivor nodes for repair) is similar to the case of $(n = 6,\ d = 4,\ k = 3,\ r = 1)$ discussed in Section III. The key idea is to use a change of bases and remap the nodes, thus making these generalized cases identical to the previously handled case. We omit details for the sake of brevity.

Remark 7 (Universality of Our Code): For all possible values of (r, d), note that the encoding submatrices are diagonal. Also because of the nature of the Schwartz–Zippel Lemma, the diagonal entries can be chosen randomly to satisfy the desired properties (alignment, MDS property and recovery of the desired components) with nonzero probability. Therefore, we can choose a sufficiently large value of M and random diagonal encoding submatrices to build a code which can simultaneously perform optimal repair of different failure scenarios. For instance, to build a code which can handle $(r = 1,\ d = n - 1)$, $(r = 1,\ d = n - 2)$, we need to choose M to be the least common multiple of $(n - k)m^N$ and $(n - k - 1)m^N$. Such a code can be interpreted as multiple blocks of size $(n - k)m^N$ for the case of $r = 1$ and $d = n - 1$ so that the strategy described earlier can be used for repair of each block separately. Similarly for the case of $d = n - 2$, the code can be interpreted as multiple blocks of size $(n - k - 1)m^N$. Thus, following this argument, storage codes can be designed, independent of (r, d). This is in contrast to the constructions in [7] and [8] where the storage code is dependent on the repair parameter d. □

V. Capacity of a Class of Multisource Nonmulticast Networks

As in the literature [3], [8], our storage network can be cast into a class of traditional communication networks. Specifically, it can be translated into a multisource nonmulticast network. The main distinction of our translation w.r.t. the previous work is to include the multiple-node failure scenario, thus leading to having more destinations with specific communication demands in the network. In this study, we exploit this connection to establish the capacity of the translated communication network.

Let us start by illustrating the network translation. For completeness, we will describe details on the translation, although they have significant overlaps with those in [3] and [8]. The translated network consists of three layers: a source layer, a storage layer, and a destination layer. See Fig. 5. The source layer has k source nodes, each having a uniformly distributed message $\mathbf{a}_i$ independent of all other messages. We assume that the number of network uses is M and each message has rate R_i. So, $\mathbf{a}_i$ is a $2^{MR_i} \times 1$ vector. The network has an intermediate node which has incoming edges from each source via an infinite-capacity link. The storage layer comprises n nodes. The top k nodes are connected with a corresponding source node via a $1/k$-capacity link, representing systematic nodes in the storage network. The bottom $(n - k)$ nodes are linked with the intermediate node through a $1/k$-capacity link, representing parity nodes. Note that link capacity w.r.t. a storage node is analogous to the storage cost. The destination layer consists of two types of nodes—repair destination nodes and MDS-destination nodes. Each repair scenario is represented by a repair destination node. In this network, we consider systematic-node repair scenarios only.

We partition the repair destination nodes into sets $\mathcal{D}_{r,d}$ based on

1) the number d of storage nodes that a repair destination node connects to;
2) the number r of failed systematic storage nodes,

where $\mathcal{D}_{r,d}$ denote the set of all repair destination nodes that wish to decode r messages with access to d storage nodes. On a failure of r nodes, we have $(n-r)$ storage nodes that survive, so $k \leq d \leq n-r$. Since there are $\binom{n-r}{d}$ repair scenarios and $\binom{k}{r}$ systematic-node failure scenarios, the total number of destination nodes in $\mathcal{D}_{r,d}$ is $\binom{k}{r}\binom{n-r}{d}$. We assume a capacity of $\beta(r,d)$ for the link between a repair destination node in $\mathcal{D}_{r,d}$ and each of its corresponding d storage nodes. To ensure the MDS property, we have a set of MDS-destination nodes which intend to decode all of the k messages with access to the top k storage nodes. Note that there are $\binom{n}{k}$ such MDS-destination nodes. We assume a capacity of $\frac{1}{k}$ for an incoming link.

A rate tuple $(R_1, R_2, \ldots, R_k)$ is said to be achievable if there exists a code such that every destination node can decode its desired messages with a probability of error that vanishes as M tends to infinity. The capacity region of the network is the convex closure of the set of all achievable rate tuples.

Theorem 3: 1) If $\beta(r,d) = \frac{r}{k(d+r-k)}$, then the capacity region is

$$\left\{(R_1, R_2, \ldots, R_k) : R_k \leq \frac{1}{k}\right\}.$$

2) If the rate tuple $\left(\frac{1}{k}, \frac{1}{k}, \ldots, \frac{1}{k}\right)$ is achievable, then

$$\beta(r,d) \geq \frac{r}{k(d+r-k)}$$

for all $r \leq \min(k, n-k)$ and $k \leq d \leq n-r$.

Remark 8: Note that the quantity $Md\beta(r,d)$ is analogous to the total repair bandwidth in the storage setup, given that the rate tuple $R_i = \frac{1}{k},\ i = 1, 2, \ldots, k$ is achievable. Since the second claim in this theorem shows that $\beta(r,d)$ cannot be less than $\frac{r}{k(d+r-k)}$, it also shows that the repair bandwidths in Theorems 1 and 2 are optimal.

Proof: **Part 1**: The achievability proof is straightforward. It simply follows from that of Theorem 1. Specifically, the link from the intermediate node to the storage node i carries an $M/k \times 1$ vectors of the form (20). Also, the repair strategy in the storage setup determines the vector associated with the link between a storage and a repair destination node. The converse is simply due to the MDS property. An MDS destination node must be able to decode all of the k messages. Since each storage node has an incoming link of rate $\frac{1}{k}$, we have $R_i \leq \frac{1}{k}$.

Part 2: We employ a cut-set bound argument. Consider a repair destination node which wants to decode $(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r)$ with access to some d storage nodes, say $(l_1, l_2, \ldots, l_d)$, among nodes $(r+1, r+2, \ldots, n)$. Note that this repair destination node should have the information contained in the first r storage nodes. The MDS property implies that the repair destination node combined with any $(k-r)$ of the storage nodes other than nodes $(1, 2, \ldots, r)$ must be able to reconstruct all the original messages.

We now construct a cut in the network as follows. The destination side of the cut consists of the repair destination node and part of the connected storage nodes $(l_1, l_2, \ldots, l_{k-r})$. Note that $k - r < d$. All of the other nodes in the system belong to the source side of the cut. For example, in Fig. 5, the shaded nodes indicate the destination side of the cut for the bound on $\beta(1,3)$. Note that the flow across the cut should be at least 1—the total rate of all the messages.

The total flow across this cut is equal to the sum of the two: 1) total flow into storage nodes $(l_1, l_2, \ldots, l_{k-r})$; and 2) the total flow into the repair destination node from the remaining connected storage nodes $(l_{k-r+1}, \ldots, l_d)$. Therefore, we need

$$(k-r) \cdot \frac{1}{k} + \{d - (k-r)\} \cdot \beta(r,d) \geq 1.$$

This implies that

$$\beta(r,d) \geq \frac{r}{k(d+r-k)}.$$

This completes the proof.

VI. Conclusion

We explored the exact repair problem in distributed storage systems to characterize the minimum repair bandwidth for a multiple-node failure in MDS codes. We showed that in contrast to the result of [32], the repair bandwidth for exact repair is asymptotically the same as that of functional repair.

As a byproduct, we also established the capacity region of a class of multisource nonmulticast networks. An interesting technical aspect of the result involves the achievable scheme that inherits the asymptotic interference alignment developed in the context of wireless interference channels.

Our study spawns some interesting research directions. The first is w.r.t. translating our theoretical insights into practice. Note that our codes suffer from the following limitations.

1) Our result is w.r.t. the existence of optimal codes.
2) Our codes achieve the minimum repair bandwidth only in the limit of a large file size, and over a sufficiently large field size.

This requires significant efforts in developing explicit MDS codes with a finite field size. In fact, subsequent to our study, several works [26]–[28], [33]–[35] have addressed many of the previous issues through explicit MDS code constructions. Despite these efforts, however, explicit code constructions are far from being closed, especially for handling multiple node failures, and for all possible values of d.

The second research direction is to apply interference alignment to nonmulticast multihop networks. In this regard, there has been recent interest in [36]–[40] where interference-alignment-based network coding schemes are developed for multiple unicast networks. We believe that our study provides different insights and therefore helps making significant progress on the networks.

Appendix
Proof of (21)

We intend to show that the determinant of the matrix in (21) is a nonzero polynomial in its entries. Assume without loss of generality that $j_1, j_2, \ldots, j_k$ are in ascending order. Let $j_1, j_2, \ldots, j_{k-m} \in \{1, 2, \ldots, k\}$ and

$j_{k-m+1}, j_{k-m+2}, \ldots, j_k \in \{k+1, k+2, \ldots, n\}$. We want to show that the determinant of the following matrix is a nonzero polynomial of its entries:

$$\begin{bmatrix} \mathbf{A}_{j_1,1} & \mathbf{A}_{1,2} & \cdots & \mathbf{A}_{1,k} \\ \mathbf{A}_{j_2,1} & \mathbf{A}_{2,2} & \cdots & \mathbf{A}_{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{j_{k-m},1} & \mathbf{A}_{j_{k-m},2} & \cdots & \mathbf{A}_{j_{k-m},k} \\ \mathbf{A}_{j_{k-m+1},1} & \mathbf{A}_{j_{k-m+1},2} & \cdots & \mathbf{A}_{j_{k-m+1},k} \\ \mathbf{A}_{j_{k-m+2},1} & \mathbf{A}_{j_{k-m+2},2} & \cdots & \mathbf{A}_{j_{k-m+2},k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{j_k,1} & \mathbf{A}_{j_k,2} & \cdots & \mathbf{A}_{j_k,k} \end{bmatrix}.$$

Since

$$\mathbf{A}_{j,i} = \begin{bmatrix} \mathbf{0} & j \neq i \\ \mathbf{I} & j = i \end{bmatrix}, \ \forall i \in \{1, 2, \cdots, k\}$$

the previous matrix is equal to

$$\begin{bmatrix} \mathbf{I} & \ldots & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathbf{0} & \ldots & \mathbf{I} & \ldots & \mathbf{0} \\ \mathbf{A}_{j_{k-m+1},1} & \ldots & \mathbf{A}_{j_{k-m+1},k-m} & \ldots & \mathbf{A}_{j_{k-m+1},k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{j_k,1} & \ldots & \mathbf{A}_{j_k,k-m} & \ldots & \mathbf{A}_{j_k,k} \end{bmatrix}.$$

It suffices to show that the determinant formed by the previous matrix is a nonzero polynomial. To show this, it suffices if there is a choice of coding coefficients (i.e., diagonal entries of $\mathbf{A}_{i,j},\ i = k+1, \ldots, n,\ j = 1, \ldots, k$) for which the determinant is nonzero scalar (since the zero polynomial evaluates to 0 for all possible coding co-efficients). Suppose we choose our coding coefficients as follows:

$$\mathbf{A}_{j,i} = \begin{cases} \mathbf{0}, & \text{if } (j,i) \notin \{(j_t, t) : t = k-m+1, \cdots, k\} \\ \mathbf{I}, & \text{o.w.} \end{cases}$$

Then, the previous matrix is the identity matrix which clearly has a nonzero determinant. This implies that the determinant a nonzero polynomial in the diagonal entries of $\mathbf{A}_{j,i}$.

REFERENCES

[1] V. R. Cadambe, S. A. Jafar, and H. Maleki, "Distributed data storage with minimum storage regenerating codes—Exact and functional repair are asymptotically equally efficient," Apr. 2010 [Online]. Available: arXiv:1004.4229

[2] C. Suh and K. Ramchandran, "On the existence of optimal exact-repair MDS codes for distributed storage," Apr. 2010 [Online]. Available: arXiv:1004.4663

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

[4] Y. Wu and A. G. Dimakis, "Reducing repair traffic for erasure coding-based storage via interference alignment," in *Proc. IEEE Int. Symp. Inf. Theory*, Seoul, Korea, Jul. 2009, pp. 2276–2280.

[5] D. Cullina, A. G. Dimakis, and T. Ho, "Searching for minimum storage regenerating codes," presented at the Allerton Conf. Control, Comput. Commun., Sep. 2009.

[6] Y. Wu, "A construction of systematic MDS codes with minimum repair bandwidth," Oct. 2009 [Online]. Available: arXiv:0910.2486

[7] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Explicit codes minimizing repair bandwidth for distributed storage," in *IEEE Inf. Theory Workshop*, Cairo, Egypt, Jan. 2010.

[8] C. Suh and K. Ramchandran, "Exact-repair MDS code construction using interference alignment," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1425–1442, Mar. 2011.

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

[10] K. Rashmi, N. Shah, and P. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," presented at the Inf. Theory Appl. Workshop, Feb. 2012.

[11] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[12] K. Rashmi, N. Shah, P. Kumar, and K. Ramchandran, "Explicit construction of optimal exact regenerating codes for distributed storage," in *Proc. IEEE 47th Annu. Allerton Conf. Commun., Control, Comput.*, 2009, pp. 1243–1249.

[13] N. Shah, K. Rashmi, P. Kumar, and K. Ramchandran, "Interference alignment in regenerating codes for distributed storage: Necessity and code constructions," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2134–2158, Apr. 2012.

[14] M. Blaum, J. Brady, J. Bruck, and J. Menon, "Evenodd: An optimal scheme for tolerating double disk failures in raid architectures," in *Proc. 21st Annu. Int. Symp. Comput. Architect.*, Apr. 1994, pp. 245–254.

[15] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, "Row-diagonal parity for double disk failure correction," in *Proc. 3rd USENIX Symp. File Storage Technol.*, 2004, pp. 1–14.

[16] 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.

[17] K. W. Shum and Y. Hu, "Exact minimum-repair-bandwidth cooperative regenerating codes for distributed storage systems," in *IEEE Int. Symp. Inf. Theory*, Saint Petersburg, Russia, Jul. 2011.

[18] K. W. Shum, "Cooperative regenerating codes for distributed storage systems," in *IEEE Int. Conf. Commun.*, Kyoto, Japan, Jun. 2011.

[19] A. Rawat, S. Vishwanath, A. Bhowmick, and E. Soljanin, "Update efficient codes for distributed storage," in *IEEE Int. Symp. Inf. Theory*, St. Petersburg, Russia, Jul. 2011.

[20] V. R. Cadambe and S. A. Jafar, "Interference alignment and the degrees of freedom for the K user interference channel," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3425–3441, Aug. 2008.

[21] M. A. Maddah-Ali, S. A. Motahari, and A. K. Khandani, "Communication over MIMO X channels: Interference alignment, decomposition, and performance analysis," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3457–3470, Aug. 2008.

[22] S. A. Jafar and S. Shamai, "Degrees of freedom region for the MIMO X channel," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 151–170, Jan. 2008.

[23] "Interference alignment—A new look at signal dimensions in a communication network," *Found. Trends Commun. Inf. Theory*, vol. 7, no. 1, pp. 1–134, 2010.

[24] V. R. Cadambe, C. Huang, and J. Li, "Permutation code: Optimal exact-repair of a single failed node in MDS code based distributed storage systems," in *Proc. IEEE Symp. Inf. Theory*, Jul. 2011, pp. 1225–1229.

[25] I. Tamo, Z. Wang, and J. Bruck, "MDS array codes with optimal rebuilding," in *Proc. IEEE Symp. Inf. Theory*, Jul. 2011, pp. 1240–1244.

[26] D. S. Papailiopoulos and A. G. Dimakis, "Distributed storage codes through Hadamard designs," in *Proc. IEEE Symp. Inf. Theory*, Jul. 2011, pp. 1230–1234.

[27] V. R. Cadambe, C. Huang, S. A. Jafar, and J. Li, "Optimal repair of MDS codes in distributed storage via subspace interference alignment," Jun. 2011 [Online]. Available: arXiv:1106.1250

[28] I. Tamo, Z. Wang, and J. Bruck, Zigzag codes: MDS array codes with optimal rebuilding CoRR, 2011 [Online]. Available: http://arxiv.org/abs/1112.0371

[29] C. Suh and D. Tse, "Interference alignment for cellular networks," in *Proc. Allerton Conf. Control, Comput. Commun.*, Sep. 2008, pp. 1037–1044.

[30] V. R. Cadambe and S. A. Jafar, "Interference alignment and the degrees of freedom of wireless X networks," *IEEE Trans. Inf. Theory*, vol. 55, no. 9, pp. 3893–3908, Sep. 2009.

[31] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 1995.
[32] N. Shah, K. Rashmi, P. Kumar, and K. Ramchandran, "Distributed storage codes with repair-by-transfer and non-achievability of interior points on the storage-bandwidth tradeoff," *IEEE Trans. Inf. Theory.*, vol. 58, no. 3, pp. 1837–1852, Mar. 2012.
[33] I. Tamo, Z. Wang, and J. Bruck, MDS array codes with optimal rebuilding CoRR, 2011 [Online]. Available: http://arxiv.org/abs/1103.3737
[34] D. Papailiopoulos, A. Dimakis, and V. Cadambe, "Repair optimal erasure codes through hadamard designs," in *Proc. 49th Annu. Allerton Conf. Commun., Control, and Computing*, 2011, pp. 1382–1389.
[35] V. Cadambe, C. Huang, J. Li, and S. Mehrotra, "Polynomial length MDS codes with optimal repair in distributed storage," in *Proc. Conf. Rec. 45th Asilomar Conf. Signals, Syst. Comput.*, 2011, pp. 1382–1389.
[36] A. Das, S. Vishwanath, S. Jafar, and A. Markopoulou, "Network coding for multiple unicasts: An interference alignment approach," in *Proc. IEEE Int. Symp. Inf. Theory*, 2010, pp. 1878–1882.
[37] A. Ramakrishnan, A. Das, H. Maleki, A. Markopoulou, S. Jafar, and S. Vishwanath, "Network coding for three unicast sessions: Interference alignment approaches," in *Proc. 48th Annu. Allerton Conf. Commun., Control, Computing*, 2010, pp. 1054–1061.
[38] H. Maleki, V. Cadambe, and S. Jafar, "Index coding-An interference alignment perspective," 2012 [Online]. Available: arXiv:1205.1483
[39] C. Meng, A. Ramakrishnan, A. Markopoulou, and S. Jafar, "On the feasibility of precoding-based network alignment for three unicast sessions," 2012 [Online]. Available: arXiv:1202.3405
[40] S. Kannan and P. Viswanath, "Capacity of multiple unicast in wireless networks: A polymatroidal approach," 2011 [Online]. Available: http://arxiv.org/abs/1111.4768

Viveck R. Cadambe (S'06–M'11) holds a joint position as a postdoctoral fellow at the Research Lab of Electronics (RLE) at MIT, and as a postdoctoral researcher at the ECE department in Boston University. He received his Ph.D. from the University of California, Irvine in the Department of Electrical Engineering and Computer Science in 2011. He received my B.S. and M.S. degrees in Electrical Engineering from the Indian Institute of Technology Madras, Chennai, in 2006. His research interests include information theory, coding theory, wireless communications, communication and storage networks.

Dr. Cadambe is a recipient of the 2009 IEEE Information Theory Society Paper Award and the UCI Electrical Engineering and Computer Science Department Best Paper Award for 2008-09. His dissertation received the 2011 CPCC Best Dissertation Award at UCI. He is also a recipient of the University of California, Irvine CPCC graduate fellowship for the year 2007-08. He was a summer intern at the Communications, Collaboration and Systems Group at Microsoft Research, Redmond WA during June-September of 2010.

Syed Ali Jafar (S'99–M'04–SM'09) received his B. Tech. from IIT Delhi, India, in 1997, M.S. from Caltech, Pasadena, USA, in 1999, and Ph.D. from Stanford University, USA, in 2003, all in Electrical Engineering. His industry experience includes positions at Lucent Bell Labs, Qualcomm Inc. and Hughes Software Systems. He is currently an Associate Professor in the Department of Electrical Engineering and Computer Science at the University of California Irvine, Irvine, CA, USA. His research interests include multiuser information theory and wireless communications.

Dr. Jafar received the NSF CAREER award in 2006, the ONR Young Investigator Award in 2008, the Information Theory Society paper award in 2009, the Maseeh Outstanding Research Award in 2010, and an IEEE GLOBECOM Best Paper Award in 2012. Dr. Jafar received the UC Irvine EECS Professor of the Year award four times, in 2006, 2009, 2011 and 2012, from the Engineering Students Council and the Teaching Excellence Award in 2012 from the School of Engineering. He was a University of Canterbury Erskine Fellow in 2010 and is an IEEE Communications Society Distinguished Lecturer for 2013–2014. Dr. Jafar was the inaugural instructor for the First Canadian School of Information Theory in 2011, a plenary speaker for various conferences and workshops including SPCOM 2010, CTW 2010 and SPAWC 2012. He served as Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS 2004–2009, for the IEEE COMMUNICATIONS LETTERS 2008–2009 and for the IEEE TRANSACTIONS ON INFORMATION THEORY 2009–2012.

Hamed Maleki (S'08) received the B.Sc. degree in electrical engineering from Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, and the M.Sc. degree in electrical engineering from University of Tehran in 2006 and 2008, respectively. He is currently working toward the Ph.D. degree at the University of California, Irvine. His current research interests include multiuser information theory and wireless communications. Mr. Maleki was a recipient of UC Irvine Graduate Fellowship for the year 2008–2009. He was also a recipient of the University of California, Irvine CPCC graduate fellowship for the year 2009–2010.

Kannan Ramchandran (F'05) is a Professor of Electrical Engineering and Computer Science at the University of California at Berkeley, where he has been since 1999. Prior to that, he was with the University of Illinois at Urbana-Champaign from 1993 to 1999, and was at AT&T Bell Laboratories from 1984 to 1990. His current research interests include distributed signal processing algorithms for wireless sensor and ad hoc networks, multimedia and peer-to-peer networking, multiuser information and communication theory, and wavelets and multiresolution signal and image processing. Prof. Ramchandran is a Fellow of the IEEE. His research awards include the Elaihu Jury award for the best doctoral thesis in the systems area at Columbia University, the NSF CAREER award, the ONR and ARO Young Investigator Awards, two Best Paper awards from the IEEE Signal Processing Society, a Hank Magnuski Scholar award for excellence in junior faculty at the University of Illinois, and an Okawa Foundation Prize for excellence in research at Berkeley. He has published extensively in his field, holds 8 patents, serves as an active consultant to industry, and has held various editorial and Technical Program Committee positions.

Changho Suh (S'10–M'12) is an Assistant Professor in the Department of Electrical Engineering at Korea Advanced Institute of Science and Technology (KAIST) since 2012. He received the B.S. and M.S. degrees in Electrical Engineering from KAIST in 2000 and 2002 respectively, and the Ph.D. degree in Electrical Engineering and Computer Sciences from UC-Berkeley in 2011. From 2011 to 2012, he was a postdoctoral associate at the Research Laboratory of Electronics in MIT. From 2002 to 2006, he had been with the Telecommunication R&D Center, Samsung Electronics.

Dr. Suh received the David J. Sakrison Memorial Prize for outstanding doctoral research from the UC-Berkeley EECS Department in 2011, the Best Student Paper Award of the IEEE International Symposium on Information Theory in 2009 and the Outstanding Graduate Student Instructor Award in 2010. He was awarded several fellowships, including the Vodafone U.S. Foundation Fellowship in 2006 and 2007; the Kwanjeong Educational Foundation Fellowship in 2009; and the Korea Government Fellowship from 1996 to 2002.