Asymptotic Interference Alignment for Optimal Repair of MDS codes in Distributed Data Storage

Viveck R. Cadambe*, Syed A. Jafar*, Hamed Maleki*,
Kannan Ramchandran† and Changho Suh†

Abstract

The high repair bandwidth cost of (n, k) Maximum Distance Separable (MDS) erasure codes has motivated the search for a new class of codes that can significantly reduce repair bandwidth cost over that of conventional MDS codes. In this paper, we address (n, k, d) MDS codes, which allow for any single failed node to be repaired exactly with access to any arbitrary set of d survivor nodes, where $k \leq d \leq n-1$. We show the existence of MDS codes that achieve minimum repair bandwidth (matching the cutset lower bound) for arbitrary admissible (n, k, d), i.e., $k < n$ and $k \leq d \leq n-1$. Further, we show that our code constructions are repair-bandwidth-optimal for *any* recoverable failure scenario, i.e., the failure of an arbitrary set of $r \leq n-k$ nodes can also be optimally repaired. Our approach is based on the asymptotic interference alignment scheme proposed by Cadambe and Jafar for the wireless interference channel. Interestingly, our results lead to the capacity characterization of a class of multi-source (non-multicast) wired networks.

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 split into k equal parts of size M/k and stored in the first k storage nodes, also known as systematic nodes. The remaining $(n-k)$ nodes, known as parity nodes or non-systematic nodes, store data of the same size, i.e., M/k, adding redundancy to protect from failure of storage nodes. The parity nodes are designed so that a failure of up to $(n-k)$ storage nodes can be tolerated, i.e., the original file can be completely recovered from the data stored at any k nodes out of the original n nodes. 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. Now, consider the case where only $r \leq n-k$ 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

This journal paper is based on work done contemporaneously and independently by two groups: (1) Viveck R. Cadambe, Syed A. Jafar and Hamed Maleki [1] from UC-Irvine (2) Changho Suh and Kannan Ramchandran [2] from UC-Berkeley. The authors decided to consolidate their work into a single manuscript and list the authors' names in alphabetical order.

*Viveck R. Cadambe, Syed A. Jafar and Hamed Maleki are with the Department of Electrical Engineering and Computer Science, University of California at Irvine, CA 92697. {vcadambe,syed,hmaleki}@uci.edu

†Kannan Ramchandran and Changho Suh are with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, CA 94704. {kannanr,chsuh}@eecs.berkeley.edu

The research of C. Suh and K. Ramchandran was funded in part by an AFOSR grant (FA9550-09-1-0120), a DTRA grant (HDTRA1-09-1-0032) and an NSF grant (CCF-0830788).

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 reconstruct 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 downloaded by the repair center, M, 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 the repair center downloading data of size less than M. More specifically, the question of interest of this paper is *what is the minimum repair bandwidth required to repair r failed nodes?* The question of minimum repair bandwidth has been studied previously for the case of a single failed node from two perspectives [3], [4], [5], [6], [7], [8]. The first is called functional regeneration [3], [4], [5] and the second is called exact (or systematic) regeneration [6], [7], [8], [9], [10].

The functional regeneration problem requires the repair center to replace the failed nodes by a function of the data, so that the reconstructed new nodes, along with the other nodes satisfy the property of being an (n,k) MDS code. In other words, the repaired nodes are information equivalent to the originally stored data. Note that in the functional regeneration problem, the data stored by the repaired nodes need not be identical to the data stored by the failed nodes; all that is required is that the repaired nodes along with the other nodes form an MDS code. This problem has been shown to be equivalent to finding the capacity of a particular wired single-source multi-cast network for the case of a single failed node. Since random linear network coding achieves the cut-set bound in a single-source multi-cast network, the functional regeneration problem has been solved. It is shown in [4] that in the case of any single node failure, if we allowed the repair center to connect to any arbitrary $d \geq k$[1] of the remaining $n-1$ surviving nodes, the minimum bandwidth required is $B_f = \frac{Md}{k(d+1-k)}$. Note that since $\frac{Md}{k(d+1-k)}$ is smaller than M (for $k<d$), the solution implies that (functional) repair of a single failed node requires a smaller repair bandwidth than the naive approach of downloading k nodes completely, by a factor of $\frac{d}{k(d+1-k)}$.

The focus of this paper is on the exact regeneration problem, where the repair center is required to replace the failed nodes by replicas, i.e., identical copies of the failed nodes. The advantage of exact repair is that the storage system can be oblivious to the repair operation. The storage code structure after the repair is exactly the same as before the failure. This is not the case with functional repair, where, in general, the code is changed every time repairs are made. Exact repair is also especially useful for maintaining the systematic structure of a systematic code after repair. There is a practical advantage of preservation of a systematic structure which ensures direct access to the data for a client, since the client can simply download it from the k systematic nodes without any decoding. The constraints for exact regeneration, however, are more strict than functional regeneration. Since any solution for the exact regeneration problem is also a solution to functional regeneration problem, $B_f = \frac{Md}{k(d+1-k)}$ serves as a lower bound to the minimum repair bandwidth for the exact repair of a single failed node. Prior to this work, it was not known whether this bound is tight. The exact regeneration problem is especially challenging because it is related to the multi-source network capacity problem with arbitrary demands (i.e., non-multicast) whose general case remains a challenging open problem in network information theory. In general, for the network capacity problem, linear codes are insufficient [11]. This

[1]Note that the repair center has to connect to at least k nodes to recover the lost data. Therefore, we have $d \geq k$.

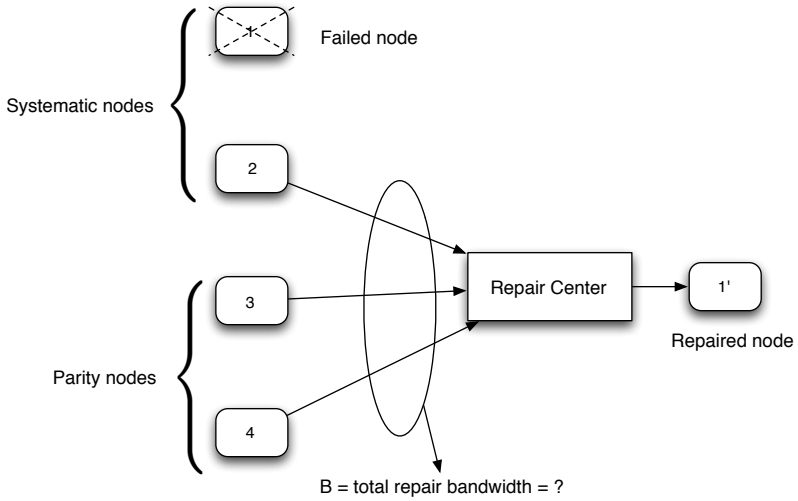

Fig. 1. Pictorial Representation of Problem Definition for $n = 4, k = 2, r = 1, d = 3$

is in contrast with the functional regeneration problem which is related to the simpler single-source *multicast* network capacity problem where random linear network coding is optimal [12], [13], [14]. In this work, we settle the open problem of the minimum repair bandwidth required for exact regeneration and show that the lower bound of B_f is indeed tight and achievable via linear codes for arbitrary feasible values of (n, k). Further, interestingly, our solution to the exact regeneration problem leads to the capacity characterization of a class of multi-source (non-multicast) networks. Linear codes suffice for this class of networks. Before we proceed, it is worth mentioning that certain previous works ([15] and references therein) also consider cases where systematic nodes store more data than of size M/k, to further reduce repair bandwidth. In this work, the focus is only on the setting where each of the n nodes store the minimum possible data of size exactly M/k (also termed as the minimum storage repair (MSR) codes problem in [4]).

A. Related Work

The topic of exact repair codes has received attention in the recent literature [6], [16], [9], [5], [7]. Wu-Dimakis [6] and Cullina-Dimakis-Ho [9] showed that the lower bound of B_f can be attained for the cases of (a) $k = 2$ and $k = n - 1$; (b) $(n, k, d) = (5, 3, 4)$ respectively. Shah-Rashmi-Kumar-Ramchandran [16] and Wu [5] developed partial exact repair codes for the cases of $\frac{k}{n} \leq \frac{1}{2} + \frac{2}{n}$ [16] and $k = d + 1$ [5], where exact repair is limited to the systematic component of the code. Suh and Ramchandran [7] showed the optimality of the functional regeneration lower bound B_f with *scalar linear* codes[2] for the case of $\frac{k}{n} \leq \frac{1}{2}, d \geq 2k - 1$. Surprisingly, there is no cost for exact regeneration over functional regeneration of a failed node, and a repair bandwidth of $\frac{Md}{k(d+1-k)}$ suffices even for exact regeneration for all the above mentioned cases. The solutions for the cases mentioned stem from drawing parallels between the exact regeneration problem and the *wireless* interference channel [7]. Such parallels enable

[2]In scalar linear codes, symbols downloaded for repair by the repair center are scalars.

the use of the interference management technique of interference alignment [17], [18], [20] for the exact regeneration problem. However, for *arbitrary* feasible values of (n, k, d) including the practically relevant low redundancy case of $k > \max(n/2, 3)$, the minimum repair bandwidth remained open. The only insight in previous literature for this case comes from [16], where it was shown that scalar linear codes cannot achieve the lower bound of B_f if $\frac{k}{n} > \frac{1}{2} + \frac{2}{n}$. Allowing for non-linear or *vector* linear codes, the tightness of B_f has remained open. See [15], [19] for a detailed history. The key contribution of this paper which resolves this open problem is described next.

B. Summary of Contribution

Consider a (n, k, d) distributed storage system which stores a file of size M using an (n, k) MDS code, where each storage node stores a file of size M/k. Let B_d represent the minimum repair bandwidth required to repair a single failed node, when the repair center is constrained to connect to an arbitrary set of d surviving nodes, where $d \geq k$. Then, the main result of the paper can be formally stated as follows.

Theorem 1: The minimum repair bandwidth B_d for exact regeneration in a (n, k) distributed storage system satisfies

$$\lim_{M\to\infty} \frac{B_d}{M} = \frac{d}{k(d+1-k)}.$$

Equivalently, we can write

$$B_d = \frac{Md}{k(d+1-k)} + o(M)$$

The above result characterizes the minimum repair bandwidth for an (n, k) code, where the repair center connects to $d \geq k$ storage nodes for repair. Simply put, the result states that *exact repair is asymptotically equally efficient as functional repair, in the limit of large file sizes.* While the lower bound has been previously established in the context of functional repair, our main contribution is an asymptotic MDS code and a repair strategy which is constructed by drawing inspiration from the asymptotic interference alignment solution for the K user wireless interference channel in [20]. Unlike previous related literature which provides explicit code constructions, our approach shows the existence of codes which satisfy the lower bound (asymptotically). At the cost of the asymptotic nature of our code construction, our result affords generalizations that are not available through the explicit constructions in previous literature. These generalizations and other notable aspects of our results are listed below.

- Our approach can be used to repair multiple node failures as well, where the repair is conducted by a repair center connecting to any feasible subset of the surviving nodes. On noting that an MDS code can tolerate up to $(n-k)$ node failures, consider the case where r nodes fail, where $1 \leq r \leq n-k$. Now, if $r > k$, then, clearly, a repair bandwidth of M is optimal because the entire original file has to be reconstructed; on reconstruction of the original file, the failed nodes can be regenerated. Therefore, we focus on the case where $r < \min(k, n-k)$, where the repair center can connect to d of the surviving $n-r$ nodes, where $k \leq d \leq n-r$. For this case, the minimum repair bandwidth B can be shown to satisfy $\lim_{M\to\infty} \frac{B}{M} = \frac{rd}{k(d+r-k)}$. The achievability of this bandwidth is based on asymptotic interference alignment. The outer bound comes from generalizing the outer bounds of [4] described in Section V.

- Because the exact repair problem is related to a class of multi-source network coding problems with arbitrary demands (i.e., non-multicast), our class of codes leads to a characterization of the capacity region of a non-trivial class of networks. The outer bound for this class of networks, much like [4] uses the cut-set outer bound, and establishes an outer bound of $\frac{Mrd}{k(d+r-k)}$ on the repair bandwidth for the case of r failed nodes. This class of networks is discussed later in Section V.
- Our codes have an interesting property that the code generation, i.e., that the storage code can be designed to be independent of the number of nodes to be connected to, i.e., d and the failure scenario to be handled, r. In other words, the *same* code can be used to handle one or multiple node failures, and arbitrary admissible values of d. Note that this is unlike previous works, where the code generated depends on the value of d. From a design perspective, this means that with the design of [6], [7], [8], the number of nodes to be contacted must be decided ahead of time. Our solution is more flexible in that this parameters can be decided after the failure. The price we pay for this flexibility is the asymptotic nature of the code.
- An interesting aspect of our code is that the coding submatrices are diagonal, and hence optimal from the perspective of encoding/update complexity [21].

Before we proceed, we formally state the first of the above generalizations in a theorem. Consider an (n, k, d) distributed storage system which stores a file of size M using an (n, k) MDS code, where each storage node stores a file of size M/k. Let $B_{r,d}$ represent the minimum repair bandwidth required to repair any r failed nodes, when the repair center is constrained to connect to an arbitrary set of d surviving nodes, where $r \leq \min(k, n-k)$ and $k \leq d \leq n-r$.

Theorem 2: The minimum repair bandwidth $B_{r,d}$ satisfies

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

The achievability is described in Section III and Section IV. The outer bound is discussed using the network capacity interpretation of the problem in Section V.

II. The Role of Interference Alignment in Exact Regeneration : $n = 4, k = 2, d = 3, r = 1$

In this paper, making use of the connection described in [7] 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 intimate connection to the wireless interference channel problem through the concept of interference alignment.

Review of $(4, 2, 3)$ Exact-Repair MDS Codes [6]: We assume that the source file size M is 4 units, so that each node stores $M/k = 2$ units each - where a unit is equivalent to an element of the field $\mathbb{F}_q$. Let $\mathbf{a} = (a_1, a_2)^t$ and $\mathbf{b} = (b_1, b_2)^t$ be $\alpha(= 2)$-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., $[\mathbf{A}_i; \mathbf{B}_i]$ corresponds to generator submatrices) for parity node i $(i = 1, 2)$. For example, parity node 1 stores 2 units of information via a 2 dimensional vector $\mathbf{a}^t\mathbf{A}_1 + \mathbf{b}^t\mathbf{B}_1$. Now, 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 2 nodes. In other words, if, for instance node 1 fails, one can download 4 linear combinations of (a_1, a_2, b_1, b_2) by downloading two nodes completely, and resolve these linear combinations to recover (a_1, a_2, b_1, b_2). Thus, node 1 can be repaired using a total repair bandwidth of 4 units.

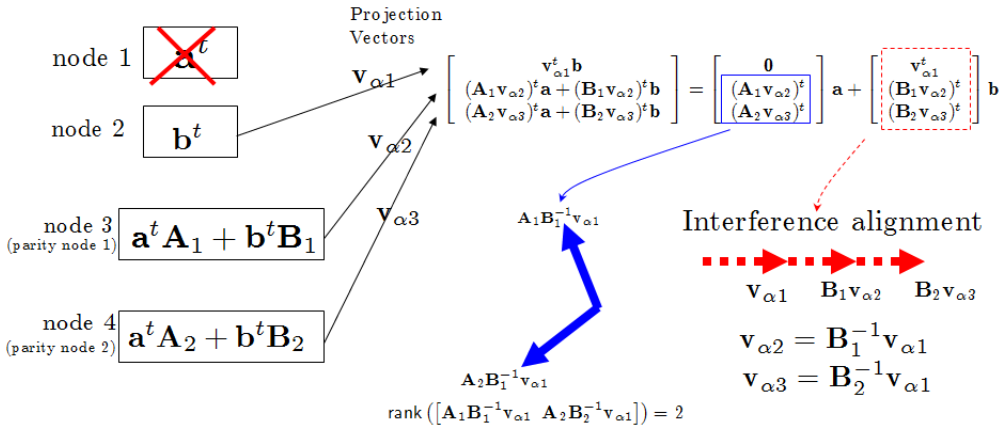

Fig. 2. Interference alignment for a $(4, 2, 3)$ Exact-Repair MDS code [6]. The blue solid-line and red dashed-line vectors indicate linear subspaces with respect to "$\mathbf{a}$" and "$\mathbf{b}$", respectively. The choice of $\mathbf{v}_{\alpha 2} = \mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1}$ and $\mathbf{v}_{\alpha 3} = \mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}$ enables to achieve interference alignment, thus allowing to decode the desired signals $\mathbf{a}$.

However, for the bound of Theorem 1 to be tight, the failed node needs to be repaired more efficiently. Specifically, the failed node needs to be repaired using a repair bandwidth of 3 units, i.e., by downloading 1 scalar from every surviving node. The cut-set bound of 3 units is tight, and can be achieved using interference alignment.

In the optimal repair scheme, 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:

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

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 scalars. With this setting, the first condition now becomes

$$\text{rank}\left(\left[\mathbf{A}_1\mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1} \;\; \mathbf{A}_2\mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}\right]\right) = 2. \tag{2}$$

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

Connection To the Wireless Interference Channel Problem [7]: Observe the three equations shown in Fig. 2:

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

Separating into two parts, we can view this repair problem as 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.

There are, however, significant differences. In the wireless problem, the channel matrices are provided by nature and therefore not controllable. The transmission strategy alone (beamforming vector) can be controlled for achieving interference alignment. On the other hand, in the storage repair problems, both matrices and vectors are controllable, i.e., projection vectors and encoding submatrices can be designed, resulting in more flexibility. This difference was exploited in [7] where optimal exact-repair codes are developed for the case of $\frac{k}{n} \leq \frac{1}{2}, d \geq 2k-1$. However, the code construction in [7] is not extendible to the other regime, i.e., $\frac{k}{n} > \frac{1}{2}$ or $d \leq 2k-1$. We shall examine the challenge behind these, previously unsolved, cases next.

III. Asymptotic Alignment for $n = 6, k = 3, d = 4, r = 1$

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], [8] which covers only a subset of all feasible values through a deterministic code construction with small alphabet size and guaranteed zero error. 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. In order to convey the concepts in a clear and concise manner, we first focus on the simplest example which does not belong to the framework in [7]: $(n = 6, k = 3, d = 4, r = 1)$. This example is sufficient to demonstrate all the relevant ideas, and is a representative of the general case. We discuss this special case in detail here and later provide an overview of the generalization in Section IV. We will begin by focusing on repair of a failed systematic node here. Parity node repair will be dealt with later in the section.

We begin 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 $M = 6$ units - this means that for Theorem 1 to be tight, we need to download 1 unit from each of the $d = 4$ nodes. Note that each node stores $M/k = 2$ units, i.e., a 2 scalars over a field $\mathbb{F}_q$. 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 2-dimensional projection vectors $\mathbf{v}_{\alpha i}$'s.

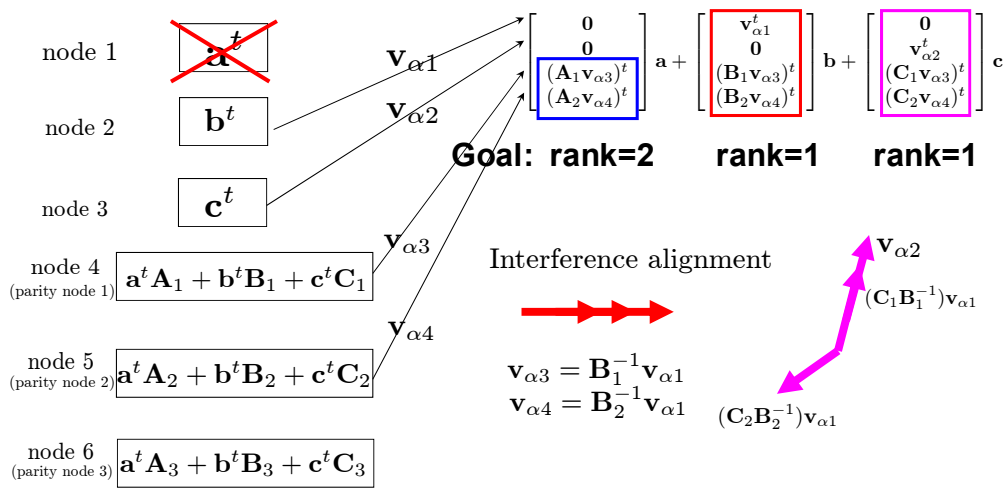

Fig. 3. Difficulty of achieving interference alignment for a scalar linear $(6, 3, 4)$ code. In accordance with the $(4, 2)$ code example in Fig. 2, if one were to set $\mathbf{v}_{\alpha 3} = \mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1}$ and $\mathbf{v}_{\alpha 4} = \mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}$, then it is possible to achieve interference alignment with respect to $\mathbf{b}$. 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.

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

$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ (\mathbf{A}_1\mathbf{v}_{\alpha 3})^t \\ (\mathbf{A}_2\mathbf{v}_{\alpha 4})^t \end{bmatrix} \mathbf{a} + \begin{bmatrix} \mathbf{v}_{\alpha 1}^t \\ \mathbf{0} \\ (\mathbf{B}_1\mathbf{v}_{\alpha 3})^t \\ (\mathbf{B}_2\mathbf{v}_{\alpha 4})^t \end{bmatrix} \mathbf{b} + \begin{bmatrix} \mathbf{0} \\ \mathbf{v}_{\alpha 2}^t \\ (\mathbf{C}_1\mathbf{v}_{\alpha 3})^t \\ (\mathbf{C}_2\mathbf{v}_{\alpha 4})^t \end{bmatrix} \mathbf{c}.$$

In order to successfully recover the two desired signal components of "$\mathbf{a}$" from the 4 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 $\mathbf{v}_{\alpha 3} = \mathbf{B}_1^{-1}\mathbf{v}_{\alpha 1}$ and $\mathbf{v}_{\alpha 4} = \mathbf{B}_2^{-1}\mathbf{v}_{\alpha 1}$, 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. In fact, as demonstrated in [8], scalar linear codes, in general, involve a greater repair bandwidth as compared to the cut-set bound.

To address a similar simultaneous interference alignment problem in wireless interference channels, reference [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 achieved 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,

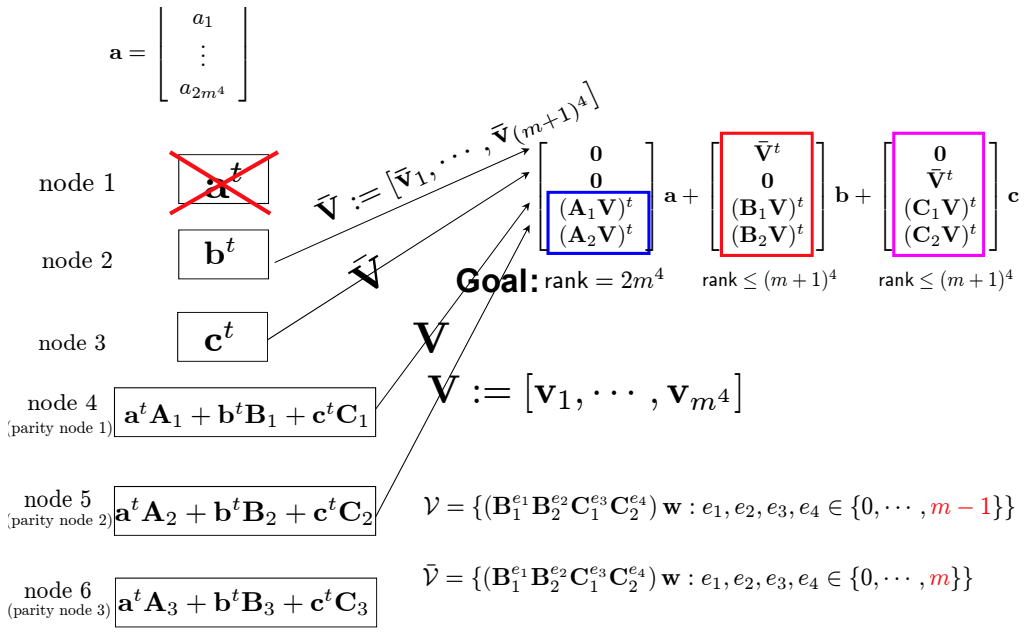

Fig. 4. Illustration of exact repair of systematic node 1. The data stored in each node is a $2m^n \times 1$ vector, where m is an arbitrarily large positive integer and the exponent N is equal to 4 and is carefully chosen depending on code parameters, specifically $N = (k-1)(d-k+1) = 4$. This corresponds to the total number of encoding submatrices involved in the connection except for those associated with desired signals. A failed node is exactly repaired by having systematic and parity survivor nodes project their data onto linear subspaces spanned by column vectors of $\bar{\mathbf{V}} := [\bar{\mathbf{v}}_1, \cdots, \bar{\mathbf{v}}_{(m+1)^4}]$ and $\mathbf{V} := [\mathbf{v}_1, \cdots, \mathbf{v}_{m^4}]$, respectively. Here $\bar{\mathbf{v}}_i \in \bar{\mathcal{V}}$ and $\mathbf{v}_i \in \mathcal{V}$. Notice that $\mathbf{B}_1\mathbf{v}_i, \mathbf{B}_2\mathbf{v}_i, \mathbf{C}_1\mathbf{v}_i, \mathbf{C}_2\mathbf{v}_i \in \bar{\mathcal{V}}, \forall i = 1, \cdots, m^4$. Hence, the matrix associated with interference $\mathbf{b}$ has rank of at most $(m+1)^4$ instead of $2m^4$. Similarly the matrix associated with interference $\mathbf{c}$ has rank of at most $(m+1)^4$. This enables simultaneous interference alignment as $m \to \infty$. On the other hand, $\mathsf{rank}[\mathbf{A}_1\mathbf{V}, \mathbf{A}_2\mathbf{V}] = 2m^4$ with probability 1, providing a probabilistic guarantee of decodability of desired signals. Finally, notice that the total repair bandwidth $\gamma = 2\frac{(m+1)^4}{6m^4} + 2m^4/6m^4 \overset{m\to\infty}{\longrightarrow} 2/3$ approaches the cutset bound as m goes to infinity (which corresponds to the amount of data stored going to infinity).

we allow M to be a larger parameter of choice[3], so that each node stores a $\alpha = M/k = M/3$ dimensional vector over the field of size $\mathbb{F}_q$. The size of the vector here, α, is analogous to the size of the symbol extension used in interference channels.

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

$$N = (k-1)(d-k+1). \tag{3}$$

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., $\mathbf{a}^t = (a_1, \cdots, a_{2m^4})$, where a_i indicates the ith component of the vector. Now with this

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

vectorization, we show a repair strategy that downloads a m^4 dimensional vector from each of nodes 4,5 and a $(m+1)^4$ dimensional vector from each of nodes $2, 3$ to repair node 1. With this strategy, and noting that $M = 6m^4$, we have

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

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

1) *Interference Alignment:* The rank of the interference corresponding to $\mathbf{b}$, and the interference corresponding to $\mathbf{c}$ are each, simultaneously, restricted to $(m+1)^4$. Such simultaneous alignment w.r.t. both $\mathbf{b}$ and $\mathbf{c}$ enables successful interference cancellation by just downloading $(m+1)^4$ linear combinations from each of nodes 2 and 3.
2) *Reconstruction of Desired Signal:* The desired signals which corresponds to $\mathbf{a}$ has a full rank of $2m^4$ enabling reconstruction.
3) *MDS Property:* The two properties above ensure successful reconstruction for a single node failure, with the desired repair bandwidth. Along with this, we also need to ensure the MDS property, i.e., to ensure that the original information $\mathbf{a}, \mathbf{b}, \mathbf{c}$ can be reconstructed from any 3 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 above are satisfied. Next, we describe our solution which achieves this repair bandwidth.

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):

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

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, i.e., $k-1 = 2$ systematic nodes and $d-k+1 = 2$ parity nodes. One can alternatively use 1 systematic node and 3 parity nodes for repair instead. This does not fundamentally alter the analysis, and will be covered in Section III-B. For the time being, assume the above configuration for the connection: $(k-1)$ systematic nodes and $(d-k+1)$ parity nodes. Each of the two parity survivor nodes participating in repair project their data using the following *projection matrix*:

$$\mathbf{V} := [\mathbf{v}_1, \cdots, \mathbf{v}_{m^4}] \in \mathbb{F}_q^{2m^4 \times m^4}, \tag{5}$$

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

$$\mathcal{V} := \left\{ (\mathbf{B}_1^{e_1} \mathbf{B}_2^{e_2} \mathbf{C}_1^{e_3} \mathbf{C}_2^{e_4})\, \mathbf{w} : e_1, e_2, e_3, e_4 \in \{0, \cdots, m-1\} \right\}, \tag{6}$$

where $\mathbf{w} = [1, \cdots, 1]^t$. Note that $|\mathcal{V}| \leq m^4$. The vector $\mathbf{v}_i$ maps to a different sequence of (e_1, e_2, e_3, e_4). For example, we can map:

$$\begin{aligned}
&\mathbf{v}_1 = \mathbf{w},\ \mathbf{v}_2 = \mathbf{C}_2\mathbf{w},\ \mathbf{v}_3 = \mathbf{C}_2^2\mathbf{w},\ \cdots, \\
&\mathbf{v}_{m^4-2} = \mathbf{B}_1^{m-1}\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-3}\mathbf{w}, \\
&\mathbf{v}_{m^4-1} = \mathbf{B}_1^{m-1}\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-2}\mathbf{w}, \\
&\mathbf{v}_{m^4} = \mathbf{B}_1^{m-1}\mathbf{B}_2^{m-1}\mathbf{C}_1^{m-1}\mathbf{C}_2^{m-1}\mathbf{w}.
\end{aligned} \tag{7}$$

Consider the equations downloaded from parity nodes 1 and 2 (nodes 4 and 5):

$$\begin{aligned}
&\text{From parity node 1: } \mathbf{a}^t(\mathbf{A}_1\mathbf{V}) + \mathbf{b}^t(\mathbf{B}_1\mathbf{V}) + \mathbf{c}^t(\mathbf{C}_1\mathbf{V}); \\
&\text{From parity node 2: } \mathbf{a}^t(\mathbf{A}_2\mathbf{V}) + \mathbf{b}^t(\mathbf{B}_2\mathbf{V}) + \mathbf{c}^t(\mathbf{C}_2\mathbf{V}).
\end{aligned} \tag{8}$$

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:

$$\bar{\mathcal{V}} := \left\{ (\mathbf{B}_1^{e_1}\mathbf{B}_2^{e_2}\mathbf{C}_1^{e_3}\mathbf{C}_2^{e_4})\,\mathbf{w} : e_1, e_2, e_3, e_4 \in \{0, \cdots, m\} \right\}. \tag{9}$$

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 allows for *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:

$$\bar{\mathbf{V}} := [\bar{\mathbf{v}}_1, \cdots, \bar{\mathbf{v}}_{(m+1)^4}] \in \mathbb{F}_q^{2m^4 \times (m+1)^4}, \tag{10}$$

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 (7). We can then guarantee that:

$$\begin{aligned}
&\mathsf{colspan}[\mathbf{B}_1\mathbf{V}, \mathbf{B}_2\mathbf{V}] \subset \mathsf{colspan}[\bar{\mathbf{V}}] \\
&\mathsf{colspan}[\mathbf{C}_1\mathbf{V}, \mathbf{C}_2\mathbf{V}] \subset \mathsf{colspan}[\bar{\mathbf{V}}].
\end{aligned} \tag{11}$$

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 $(\mathbf{b}^t(\mathbf{B}_1\mathbf{V}), \mathbf{b}^t(\mathbf{B}_2\mathbf{V}), \mathbf{c}^t(\mathbf{C}_1\mathbf{V}), \mathbf{c}^t(\mathbf{C}_2\mathbf{V}))$ from (8), 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 before. To successfully reconstruct the desired signal $\mathbf{a}$ and satisfy the second objective, we need:

$$\mathsf{rank}[\mathbf{A}_1\mathbf{V}, \mathbf{A}_2\mathbf{V}] = 2m^4. \tag{12}$$

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 third objective. The proofs of equation (12) and the MDS property are existence proofs stemming from the Schwartz-Zippel Lemma [22]. Specifically, we show that

there exist diagonal encoding submatrices $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$ so that these two properties are satisfied. The argument is as follows.

1) Consider equation (12). In the matrix on the left hand side, notice that the design of $\mathbf{V}$ in (6) does not depend on $\mathbf{A}_1$ and $\mathbf{A}_2$. Therefore, it can be noted that each entry of the matrix is a *different* monomial in the diagonal entries of the (diagonal) encoding submatrices $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$. Based on this observation, it can shown (see Lemma 1 in [23]) that the determinant of the matrix in (12) is a non-zero polynomial in the (diagonal) entries of $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i, i = 1, 2, 3$. Let us denote this polynomial by $g(.)$. Note that for (12) to be satisfied, we need $g(.)$ to evaluate to a non-zero value in the field.
2) The MDS property means that the code must be able to tolerate the failure of any 3 storage nodes in the system. Equivalently, any set of three nodes in the system, when interpreted as equations in $\mathbf{a}, \mathbf{b}, \mathbf{c}$ must have a full rank of $M = 6m^4$, and hence, the matrix representing these equations must have a non-zero determinant. Note that there are $\binom{6}{3}$ possible sets of 3 nodes in the storage system. The MDS property is therefore equivalent to showing that $\binom{6}{3} = 20$ determinants are all non-zero. 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, k that even with diagonal coding submatrices chosen here, all these polynomials are non-zero. To summarize, we show that the MDS property corresponds to 20 *non-zero* polynomials in the entries of the diagonal elements of $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$, each evaluating to a non-zero 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 $f_1(.), f_2(.), \ldots, f_{20}(.)$, and the polynomial $g(.)$ each evaluate to a non-zero value in the field. Showing this will ensure the existence of codes satisfying the final two objectives - the MDS property and the reconstruction of the desired signal - to complete the proof. To do so, we invoke the Schwartz-Zippel Lemma to product polynomial $g \cdot f_1 \cdot f_2 \ldots f_{20}$ which is a non-zero polynomial, by virtue of each of its factors being non-zero polynomials. Over a sufficiently large field, the lemma guarantees, via a probabilistic argument, the existence of (diagonal) matrices $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$ so that this product polynomial, and hence each of its factors evaluate to some non-zero value and hence completes the proof.

A. Parity Node Repair

So far, we have discussed an achievable scheme for regenerating a systematic node. The code $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$ 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. Therefore, 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. Now, we can 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 $\mathbf{a}', \mathbf{b}', \mathbf{c}'$. with

$$\begin{aligned} &\text{Node 1: } \mathbf{a}'^t = \mathbf{a}^t \\ &\text{Node 2: } \mathbf{b}'^t = \mathbf{b}^t \\ &\text{Node 3: } \mathbf{c}'^t = \mathbf{a}^t\mathbf{A}_3 + \mathbf{b}^t\mathbf{B}_3 + \mathbf{c}^t\mathbf{C}_3. \end{aligned} \tag{13}$$

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

$$\begin{aligned} &\text{Node 4: } \mathbf{a}'^t\{\mathbf{A}_1 + \mathbf{A}_3(\mathbf{C}_3)^{-1}\mathbf{C}_1\} + \mathbf{b}'^t\{\mathbf{B}_1 + \mathbf{B}_3(\mathbf{C}_3)^{-1}\mathbf{C}_1\} + \mathbf{c}'^t(\mathbf{C}_3)^{-1}\mathbf{C}_1 \\ &\text{Node 5: } \mathbf{a}'^t\{\mathbf{A}_2 + \mathbf{A}_3(\mathbf{C}_3)^{-1}\mathbf{C}_2\} + \mathbf{b}'^t\{\mathbf{B}_2 + \mathbf{B}_3(\mathbf{C}_3)^{-1}\mathbf{C}_2\} + \mathbf{c}'^t(\mathbf{C}_3)^{-1}\mathbf{C}_2. \\ &\text{Node 6: } \mathbf{c}^t = \mathbf{a}'^t\mathbf{A}_3(\mathbf{C}_3)^{-1} + \mathbf{b}'^t\mathbf{B}_3(\mathbf{C}_3)^{-1} + \mathbf{c}'^t(\mathbf{C}_3)^{-1}. \end{aligned} \tag{14}$$

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 above expressions, all the encoding submatrices $\mathbf{A}'_i, \mathbf{B}'_i, \mathbf{C}'_i$ are diagonal. This is because the sum, product and inverse of two diagonal matrices are diagonal. The diagonal property ensures that, even in this virtual code, the encoding submatrices commute. This means, by picking the repair vectors in a manner analogous to (6),(9) aligns interference so that an equation analogous to (11) is satisfied. Using an argument similar to the previous section, it can be shown that the desired signal can also be completely recovered as well. (The detailed proof is omitted here to avoid tedious notation.)

B. Participation of Arbitrary d Nodes for Exact Repair

We have so far considered a somewhat restrictive connection configuration for exact repair: namely 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: 1 systematic node and 3 parity nodes. The idea, similar to the idea of parity node repair repair, is to remap one parity node to make it look like a systematic node. We then virtually connect to 2 systematic and to 2 parity nodes. Specifically, we can remap node 6 with $\mathbf{c}'^t$ and perform conversions similar to (13),(14). Then, applying the same procedures as before we can guarantee the exact repair of $\mathbf{a}$.

IV. Generalization

In this section, we show Theorem 2 by generalizing the setting of the previous section for the case where (n, k) is arbitrary, $r \leq \min(k, n-k)$ nodes fail and $d > k$ nodes are used for repair. While the setting is more general, most of the ideas follow from the previous section. We therefore only provide a sketch of the main ideas for brevity. Consider a storage system which stores a total data M in an (n, k) MDS code based distributed storage system. The total data is represented by the $M/k \times k$ dimensional matrix $[\mathbf{a}_1 \;\; \mathbf{a}_2 \;\; \ldots \;\; \mathbf{a}_k]$, where $\mathbf{a}_i$ is an $M/k \times 1$ dimensional vector stored by systematic node $i \in \{1, 2, \ldots, k\}$. Node j, where $j \in \{k+1, k+2, \ldots, n\}$ being a parity node stores the $1 \times M/k$ vector $\mathbf{a}_1^T\mathbf{A}_{j,1} + \mathbf{a}_2^T\mathbf{A}_{j,2} + \ldots + \mathbf{a}_k^T\mathbf{A}_{j,k}$, where $\mathbf{A}_{j,i}$ is a $M/k \times M/k$ square matrix for $i \in \{1, 2, \ldots, k\}$. Because of the systematic structure of the code, we assume that for $j \leq k$,

$$\mathbf{A}_{j,i} = \begin{cases} \mathbf{0} & j \neq i \\ \mathbf{I} & j = i \end{cases}, \forall i \in \{1, 2, \ldots, k\}. \tag{15}$$

The above assumption implies that the data stored in node $j \in \{1, 2, \ldots, n\}$ is the $M/k \times 1$ vector $\mathbf{D}_j$ shown below.

$$\mathbf{D}_j^T = \sum_{i=1}^{k} \mathbf{a}_i^T \mathbf{A}_{j,i}. \tag{16}$$

Note that the encoding submatrices $\mathbf{A}_{j,i}$ for $j = k+1, k+2, \ldots, n$ are a design choice that define the code. 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 reconstructable. Thus, we need to ensure that

$$\text{rank}\left(\begin{bmatrix} \mathbf{A}_{j_1,1} & \mathbf{A}_{j_1,2} & \ldots & \mathbf{A}_{j_1,k} \\ \mathbf{A}_{j_2,1} & \mathbf{A}_{j_2,2} & \ldots & \mathbf{A}_{j_2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{j_k,1} & \mathbf{A}_{j_k,2} & \ldots & \mathbf{A}_{j_k,k} \end{bmatrix}\right) = M \tag{17}$$

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

Now, let $r \leq \min(k, n-k)$ 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. The goal of the repair center, as usual, is to regenerate 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

$$\lim_{M\to\infty} \frac{B}{M} = \lim_{m\to\infty} \frac{r(k-r)(m+1)^N + r(d+r-k)m^N}{k(d+r-k)m^N} = \frac{rd}{k(d+r-k)}$$

.

The above repair bandwidth is achieved by satisfying the following three objectives analogous to the previous section.

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) *Reconstruction of Desired Signal:* The desired signals $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r$ can be regenerated at the repair center.
3) *MDS Property:* The code is a (n, k) MDS code, i.e., equation (17) is satisfied.

Specifically, we follow the following repair strategy.

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

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 above, 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, $\forall j = k+1, k+2, \ldots, d+r$,

$$\text{colspan}(\mathbf{A}_{j,i}\mathbf{V}) \quad \subseteq \quad \text{colspan}(\bar{\mathbf{V}}), i = r+1, r+2, \ldots, k \tag{18}$$

Note that the above are the desired interference alignment relations analogous to (11) in the previous section. The above condition ensures that the entire interference can be cancelled. After interference cancellation, 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 linear reconstruction of $\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_r$, from

$$\begin{pmatrix} \mathbf{a}_1^T & \mathbf{a}_2^T & \ldots & \mathbf{a}_r^T \end{pmatrix} \left(\begin{bmatrix} \mathbf{A}_{k+1,1}\mathbf{V} & \mathbf{A}_{k+2,1}\mathbf{V} & \ldots & \mathbf{A}_{d+r,1}\mathbf{V} \\ \mathbf{A}_{k+1,2}\mathbf{V} & \mathbf{A}_{k+2,2}\mathbf{V} & \ldots & \mathbf{A}_{d+r,2}\mathbf{V} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k+1,r}\mathbf{V} & \mathbf{A}_{k+2,r}\mathbf{V} & \ldots & \mathbf{A}_{d+r,r}\mathbf{V} \end{bmatrix} \right),$$

we need

$$\text{colspan} \left(\begin{bmatrix} \mathbf{A}_{k+1,1}\mathbf{V} & \mathbf{A}_{k+2,1}\mathbf{V} & \ldots & \mathbf{A}_{d+r,1}\mathbf{V} \\ \mathbf{A}_{k+1,2}\mathbf{V} & \mathbf{A}_{k+2,2}\mathbf{V} & \ldots & \mathbf{A}_{d+r,2}\mathbf{V} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k+1,r}\mathbf{V} & \mathbf{A}_{k+2,r}\mathbf{V} & \ldots & \mathbf{A}_{d+r,r}\mathbf{V} \end{bmatrix} \right) = \frac{r.M}{k} \tag{19}$$

The above condition ensures that the desired (lost) data can be reconstructed (after interference cancellation).

Thus, essentially we need to construct $\mathbf{A}_{j,i}$ and $\mathbf{V}, \bar{\mathbf{V}}$ for $j \in \{k+1, k+2, \ldots, n\}, i \in \{1, 2, \ldots, k\}$ so that (17), (18) and (19), and hence, the three desired objectives, are satisfied. We now proceed to describe briefly our construction.

Design of Encoding Submatrices, $\mathbf{A}_{j,i}$*:* As described previously 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 as follows.

$$\mathbf{A}_{i,j} = \begin{bmatrix} \alpha_{i,j}^{(1)} & 0 & \ldots & 0 \\ 0 & \alpha_{i,j}^{(2)} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \alpha_{i,j}^{(\frac{M}{k})} \end{bmatrix} \tag{20}$$

Design of Repair Vectors, $\mathbf{V}, \bar{\mathbf{V}}$*:* In a manner similar to Section III, we choose the set of column vectors of $\mathbf{V}$ and $\bar{\mathbf{V}}$ respectively from the sets $\mathcal{V}, \bar{\mathcal{V}}$ described as follows.

$$\mathcal{V} = \left\{ \left(\prod_{\substack{j=k+1,\ldots,r+d \\ i=r+1,\ldots,k}} \mathbf{A}_{j,i}^{e_{j,i}} \right) \mathbf{w} : e_{k+1,r+1}, \ldots, e_{r+d,k} \in \{0, 1, \ldots, m-1\} \right\} \tag{21}$$

$$\bar{\mathcal{V}} = \left\{ \left(\prod_{\substack{j=k+1,\ldots,r+d \\ i=r+1,\ldots,k}} \mathbf{A}_{j,i}^{e_{j,i}} \right) \mathbf{w} : e_{k+1,r+1}, \ldots, e_{n,k} \in \{0, 1, 2, \ldots, m\} \right\} \tag{22}$$

where the entries of the $M/k \times 1$ column vector $\mathbf{w}$ is chosen to be $\begin{bmatrix} 1 & 1 & \ldots & 1 \end{bmatrix}^T$. It can be verified with the above choice of column vectors that $\mathbf{A}_{j,i}\mathbf{V} \subset \bar{\mathbf{V}}$ for $i = r+1, r+2, \ldots, k, j = k+1, k+2, \ldots, d+r$, and therefore (18) holds.

Proof of (17), (19): We have now chosen coding matrices and repair vectors so that the alignment constraints (18) are satisfied. We now need to show (17) and (19). In order to show that the matrices of (17) and (19) are full rank, it is enough to show that their determinants are non-zero. Notice that the determinant of the matrix of (17), i.e.,

$$\begin{bmatrix} \mathbf{A}_{j_1,1} & \mathbf{A}_{j_1,2} & \dots & \mathbf{A}_{j_1,k} \\ \mathbf{A}_{j_2,1} & \mathbf{A}_{j_2,2} & \dots & \mathbf{A}_{j_2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{j_k,1} & \mathbf{A}_{j_k,2} & \dots & \mathbf{A}_{j_k,k} \end{bmatrix} \tag{23}$$

is a polynomial in its entries. Note that there are $\binom{n}{k}$ polynomials of this kind, which can be represented, for $l = 1, 2, \dots, \binom{n}{k}$, as

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

where

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

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

Similarly, we need to show (19). To show that the square matrix on the left hand side of the equation has a full rank of $\frac{rM}{k}$, we need to show that its determinant is non-zero. 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 very similar to Lemma 1 of [23] can be used to show that the polynomial formed by this matrix for our solution is a non-zero polynomial (See also Appendix III in [20]). Thus, the product $f_1(.)f_2(.)...f_{\binom{n}{k}}(.)g(.)$ is non-zero polynomial of $\mathcal{A}$. Using Schwartz-Zippel Lemma, for large enough q, we have at least one choice of coding matrices and repair vectors such that these polynomials evaluate to a non-zero value, and therefore a solution exists so that (17),(19) are satisfied. Thus we have satisfied all the desired objectives, and the proof is complete.

Parity Node Repair, and Connecting to arbitrary d nodes: The extensions to parity node repair and the case where repair can be conducted by connecting to an arbitrary set of d sources are similar to the case of $(n = 6, d = 4, k = 3, r = 1)$ discussed previously in Section III. The key idea to handle both cases is that, a change of bases and a re-ordering of the nodes can be used to make these cases identical to the case we have handled above. We omit the details here for the sake of brevity.

Remark: For all possible values of r and d, note that the coding matrices 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 reconstruction of desired signal) with non-zero probability. Therefore, we can choose a sufficiently large value of M and random diagonal coding submatrices to build a code which can simultaneously perform optimal repair of various different failure scenarios (i.e., various different values of r, d). 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 an 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 where $r = 1, d = n-1$ so that the strategy described above can be used for repair of each block separately. Similarly, for

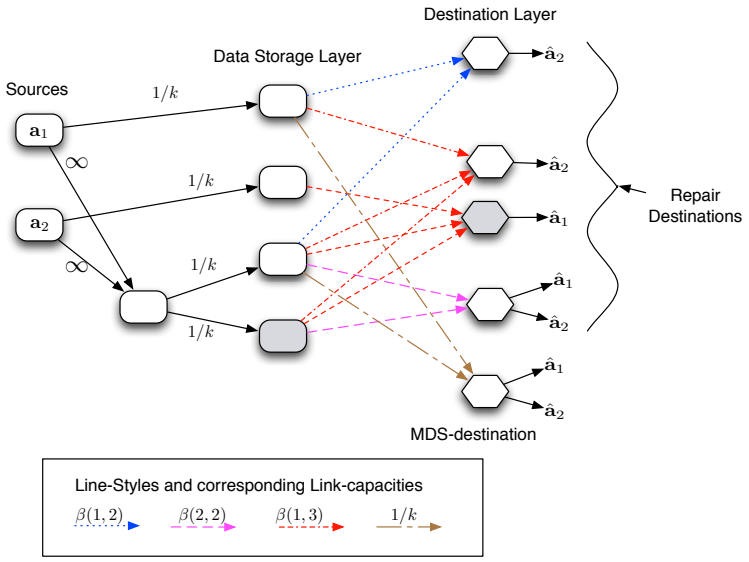

Fig. 5. The Data Storage Multi-Source Multi-Cast Network (with only a sub-set of the destination nodes shown). The shaded nodes indicate the destination side of the cut in the proof of Theorem 3

the case of $d = n - 2$, the code can be interpreted as multiple blocks of size $(n - k - 1)m^N$. Thus, following through this argument, it can be observed that the storage code design can be independent of r, d, and these parameters are determined when failure occurs and repair has to be performed. This is unlike previous explicit constructions where the storage code is dependent on the repair parameter d [8], [7].

V. Capacity of a class of Multi-Source Multicast Networks

In this section, we show that our results for the data storage problem lead to the capacity of a class of multi-source multicast wireless networks. Consider a 3-layer multi-source multicast network with a source layer, a data storage layer and a destination layer (Figure 5). The source layer has k independent uniformly distributed source nodes, with source i generating message $\mathbf{a}_i$ independent of all other messages, for $i \in \{1, 2, \ldots, k\}$. We assume that the number of channel uses of the network is equal to M, and that each source has rate R_i, so that $\mathbf{a}_i$ is $2^{MR_i} \times 1$ vector. The network then has an *intermediate node* which has incoming edges from each source node via a link of infinite capacity. There are n data storage nodes in the data storage layer. For $i \leq k$, there is directed link from data source i into data storage node i of capacity $1/k$. In other words, this link carries a $M/k \times 1$ dimensional vector. For $k < i \leq n$, there is a link of capacity $1/k$ from the intermediate node to data storage node i. Note that the capacity of the incoming link at storage node i in this network is analogous to the storage capacity of the storage node in the distributed storage set up. The destination layer consists of two types of nodes - repair destination nodes and MDS-destination nodes. Specifically, each constraints on the code corresponding to the MDS property is modeled by a destination. Similarly, each repair scenario in the storage set up corresponds to a destination node in the repair layer in the network. The multicast network here models the recovery of systematic nodes alone, and not of parity nodes. We partition the repair nodes into sets based on

- The *number* of data storage nodes d a node connects to, and
- The *number* of failed systematic storage nodes $r \leq k$ recovered by, or equivalently, the number of source messages decoded by a repair node.

Let $\mathcal{D}_{r,d}$ denote the set of all repair nodes that require to decode r sources (or equivalently, reconstruct r failed systematic nodes) on connecting to d data storage nodes, where $1 \leq r \leq \min(k, n-k)$ and $k \leq d \leq n$. On a failure of r nodes, there are $n-r$ healthy storage nodes. Therefore, d lies in the range $k \leq d < n-r$. Given a source message set $\{\mathbf{a}_{i_1}, \mathbf{a}_{i_2}, \ldots, \mathbf{a}_{i_r}\}$, there exist $\binom{n-r}{d}$ repair scenarios for each $k \leq d \leq n-r$, with a repair node corresponding to every choice of d nodes among the data storage nodes in $\{1, 2, \ldots, n\} - \{i_1, i_2, \ldots, i_r\}$. Note that this is because the repair center intends to repair failures of nodes $i_1, i_2, \ldots, i_r$, which are hence, unavailable for repair purposes. Since, given r there are $\binom{k}{r}$ possible systematic storage node failure scenarios, the total number of destination nodes in $\mathcal{D}_{r,d}$ is $\binom{k}{r}\binom{n-r}{d}$. The total number of repair destinations in the network is $\sum_{r=1}^{\min(k,n-k)} \sum_{d=k}^{n-r} |\mathcal{D}_{r,d}|$. We assume that a repair node in $\mathcal{D}_{r,d}$ is connected to each of its d corresponding data storage nodes via an incoming link of capacity $\beta(r,d)$. Finally, to ensure the MDS property we have a set of MDS-destination nodes connecting to every set of k data storage nodes intending to decode all the k source messages. Note that there are $\binom{n}{k}$ such MDS-destinations. For these MDS-destinations, we assume that the incoming links have a capacity of $\frac{1}{k}$, so that they can download the entire code in the k data storage nodes connected to such a destination.

For a given bandwidth allocation β, a rate-tuple $(R_1, R_2, \ldots, R_k)$ is said to be achievable if there exists a sequence of coding schemes indexed by M such that every destination node is able to decode all its desired messages with an error probability that vanishes asymptotically in M. The capacity region of the network is the convex closure of the set of all achievable rate-tuples.

The main result of the section is presented below.

Theorem 3: Suppose that $\beta(r,d) = \frac{r}{k(d+r-k)}$, then the capacity region of this network is

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

Further, suppose that the rate-tuple $(1/k, 1/k, \ldots, 1/k)$ is achievable, then $\beta(r,d) \geq \frac{r}{k(d+r-k)}$ for all $r \leq \min(k, n-k)$ and $k \leq d \leq n-r$.

Proof: The achievability part of the theorem follows from the achievable scheme of Theorem 1. This is because any scheme that is achievable in the data storage set up can be used as an achievable scheme in the data storage multicast network. Specifically, the link from the intermediate node to the data storage node i carries an $M/k \times 1$ vectors of the form (16). Similarly, the repair strategy in the distributed data storage set up determines the vector carried from the link from data storage layer to a repair node in the destination layer, thus ensuring achievability. The converse easily follows from the MDS property, i.e., the MDS destination that connects to data storage nodes $1, 2, \ldots, k$ must be able to decode all the k messages. Since each data storage node has an incoming link of rate $1/k$, we have $R_i \leq 1/k$.

For the second part that claims that $\beta(r,d)$ cannot be reduced, the argument comes from a cut-set bound argument similar to the converse in functional regeneration of [4]. Specifically, consider, for example, a repair node which is connected to some d data storage nodes among nodes $r+1, r+2, \ldots, n$, and decodes $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r$. Let us assume that the repair node is connected to data storage nodes $l_1, l_2, \ldots, l_d$. Note that in any achievable scheme, this repair node has all the information contained in the first r data storage nodes (which respectively store the first r sources). The MDS property of the original code implies that the repair node combined

with any $k-r$ of the data storage nodes other than nodes $1, 2, \ldots, r$ must be able to reconstruct all the original messages. Now, we construct a cut in the network as follows. The destination side of the cut consists of the repair node, and $l_1, l_2, \ldots, l_{k-r}$ - note here that $k - r < d$. All 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 a bound on $\beta(1, 3)$. Note that the flow across the cut should be at least equal to 1 - the total rate of all the source messages - because of the aforementioned MDS property. The total flow across this cut is equal to the total flow into storage nodes $l_1, l_2, \ldots, l_{k-r}$ plus the total flow into the repair node from storage nodes $l_{k-r+1}, \ldots, l_d$ which is equal to $(k-r)\frac{1}{k} + (d-(k-r))\beta(r,d)$. Therefore, we need

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

as required. ■

Note that the above theorem implies that the solution to the data storage set up leads to the capacity of a class of multi-source multicast wireless networks. Finally, note here that the quantity $Md\beta(r,d)$ is analogous to the total repair bandwidth in the distributed storage set up, given that the rate-tuple $R_i = 1/k, i = 1, 2, \ldots, k$ is achievable. Since the above theorem shows that $\beta(r,d)$ cannot be less than $\frac{r}{k(d+r-k)}$, the theorem also shows that the repair bandwidths found in Theorem 1 and Theorem 2 are optimal.

VI. Conclusion

We have shown that, surprisingly, there is no loss of exact regeneration over functional regeneration in terms of the amount of repair bandwidth per bit of repaired data, in the limit of large file sizes, regardless of the desired redundancy level. While previous work [8] has shown there is an efficiency loss for exact regeneration over functional regeneration for low redundancy levels when using scalar codes, we show asymptotic equivalence using vector codes. Also unlike previous work in [7], [8] we do not provide explicit codes or specify the minimum field size, since our arguments are based on properties of random matrices. Exploring whether the limit of large file sizes is *necessary* or whether finite file sizes are sufficient to achieve the minimum repair bandwidth is a direction of future work. Another interesting direction of future work includes interference alignment solutions for exact repair for cases beyond the minimum storage (MSR) point where the storage nodes store more than the minimum possible data.

Appendix

We intend to show that the determinant of the matrix in (17) is a non-zero polynomial in its entries. Assuming, 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 $j_{k-m+1}, j_{k-m+2}, \ldots, j_k \in \{k+1, k+2, \ldots, n\}$. Therefore, we need to show that the determinant of the following matrix is a non-zero polynomial of its entries.

$$\begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} & \ldots & \mathbf{A}_{1,k} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} & \ldots & \mathbf{A}_{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k-m,1} & \mathbf{A}_{k-m,2} & \ldots & \mathbf{A}_{k-m,k} \\ \mathbf{A}_{k+1,1} & \mathbf{A}_{k+1,2} & \ldots & \mathbf{A}_{k+1,k} \\ \mathbf{A}_{k+2,1} & \mathbf{A}_{k+2,2} & \ldots & \mathbf{A}_{k+2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k+m,1} & \mathbf{A}_{k+m,2} & \ldots & \mathbf{A}_{k+m,k} \end{bmatrix}$$

Since

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

we want the following matrix to be full rank.

$$\begin{bmatrix} \mathbf{I}_{\frac{M}{k}\times\frac{M}{k}} & \ldots & \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} & \ldots & \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} \\ \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} & \ddots & \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} & \ldots & \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} \\ \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} & \ldots & \mathbf{I}_{\frac{M}{k}\times\frac{M}{k}} & \ldots & \mathbf{0}_{\frac{M}{k}\times\frac{M}{k}} \\ \mathbf{A}_{k+1,1} & \ldots & \mathbf{A}_{k+1,k-m} & \ldots & \mathbf{A}_{k+1,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k+m,1} & \ldots & \mathbf{A}_{k+m,k-m} & \ldots & \mathbf{A}_{k+m,k} \end{bmatrix} \tag{24}$$

Therefore, we essentially need to show that the determinant formed by the above matrix is non-zero. Since the first $(k-m)\frac{M}{k} \times (k-m)\frac{M}{k}$ matrix is the identity matrix, expanding the determinant along the first $(k-m)\frac{M}{k}$ rows, the determinant can be shown to be equal to the determinant of the following matrix.

$$\mathbf{P} = \begin{bmatrix} \mathbf{A}_{k+1,(k-m)+1} & \mathbf{A}_{k+1,(k-m)+2} & \ldots & \mathbf{A}_{k+1,k} \\ \mathbf{A}_{k+2,(k-m)+1} & \mathbf{A}_{k+2,(k-m)+2} & \ldots & \mathbf{A}_{k+2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k+m,(k-m)+1} & \mathbf{A}_{k+m,(k-m)+2} & \ldots & \mathbf{A}_{k+m,k} \end{bmatrix} \tag{25}$$

We need to show that the determinant of the $m\frac{M}{k} \times m\frac{M}{k}$ matrix $\mathbf{P}$ is non-zero. Note that we have

$$\mathbf{A}_{j,i} = \begin{bmatrix} \alpha_{j,i}^{(1)} & 0 & \ldots & 0 \\ 0 & \alpha_{j,i}^{(2)} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \alpha_{j,i}^{(\frac{M}{k})} \end{bmatrix} \tag{26}$$

where each $a_{j,i}^{l}$ is independent of $a_{j',i'}^{l'}$ for $i \neq i'$ or $j \neq j'$ or $l \neq l'$. Since interchanging the rows or columns of a matrix does not change its determinant except for its sign, we make the row and column exchange operations to simplify $\mathbf{P}$. Let the rows of $\mathbf{P}$ be $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{mM/k}$. Now, we only need to show that the determinant of $\mathbf{P}'$ is non-zero where

$$\mathbf{P}' = \begin{bmatrix} \mathbf{r}_{\pi_1(1)} \\ \mathbf{r}_{\pi_1(2)} \\ \vdots \\ \mathbf{r}_{\pi_1(mM/k)} \end{bmatrix}$$

where $\pi_1 : \{1, 2, \ldots, mM/k\} \rightarrow \{1, 2, \ldots, mM/k\}$ is a permutation. Now, further, let $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{mM/k}$ be the columns of $\mathbf{P}'$. We then perform column exchange operations of $\mathbf{P}'$ to get the matrix $\mathbf{P}'' = [\mathbf{c}_{\pi_2(1)} \quad \mathbf{c}_{\pi_2(2)} \quad \ldots \quad \mathbf{c}_{\pi_2(mM/k)}]$, where, $\pi_2 : \{1, 2, \ldots, mM/k\} \rightarrow \{1, 2, \ldots, mM/k\}$ is also a permutation. Now, that the determinant of $\mathbf{P}$ is non-zero is equivalent to showing that the determinant of $\mathbf{P}''$ is non-zero. Choosing the permutations π_1, π_2 as

$$\pi_1(i) = \pi_2(i) \quad = \quad 1 + \left\lfloor \frac{i-1}{M/k} \right\rfloor + \left(i - \left\lfloor \frac{i-1}{M/k} \right\rfloor \frac{M}{k} - 1 \right) m$$

it can be verified that the $m\frac{M}{k} \times m\frac{M}{k}$ matrix $\mathbf{P}^{''}$ has a block diagonal structure, with $\frac{M}{k}$ blocks of size $m \times m$. The ith block of $\mathbf{P}^{'}$ is

$$\begin{bmatrix} \alpha^{(i)}_{k+1,(k-m)+1} & \alpha^{(i)}_{k+1,(k-m)+2} & \cdots & \alpha^{(i)}_{k+1,k} \\ \alpha^{(i)}_{k+2,(k-m)+1} & \alpha^{(i)}_{k+2,(k-m)+2} & \cdots & \alpha^{(i)}_{k+2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{(i)}_{k+m,(k-m)+1} & \alpha^{(i)}_{k+m,(k-m)+2} & \cdots & \alpha^{(i)}_{k+m,k} \end{bmatrix}. \tag{27}$$

Since the determinant of a block diagonal matrix is a product of the determinant of each of its blocks, and the determinant of the square matrix formed by the above block is a non-zero polynomial of its entries, the determinant of the matrix in (17) is a non-zero polynomial of its entries, as required.

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," *arXiv:1004.4229*, Apr. 2010.

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

[3] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *Proceedings of the IEEE INFOCOM, Anchorage, Alaska*, May 2007.

[4] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, pp. 4539–4551, Sept. 2010.

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

[6] Y. Wu and A. G. Dimakis, "Reducing repair traffic for erasure coding-based storage via interference alignment," *Proceedings of the IEEE International Symposium on Information Theory, Seoul, Korea*, July 2009.

[7] C. Suh and K. Ramchandran, "Exact-repair MDS code construction using interference alignment," *IEEE Transactions on Information Theory*, vol. 57, pp. 1425–1442, Mar. 2011.

[8] N. B. Shah, R. K. V., P. V. Kumar, and K. Ramachandran, "Explicit codes minimizing repair bandwidth for distributed storage," *CoRR*, vol. abs/0908.2984, 2009. http://arxiv.org/abs/0908.2984.

[9] D. Cullina, A. G. Dimakis, and T. Ho, "Searching for minimum storage regenerating codes," *Allerton Conference on Control, Computing and Communication*, Sept. 2009.

[10] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *submitted to the IEEE Transaction on Information Theory (arXiv:1005.4178)*, May 2010.

[11] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear network codes," *IEEE Transactions on Information Theory*, vol. 51, pp. 2745–2759, Aug. 2005.

[12] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.

[13] T. Ho, R. Koetter, M. Médard, M. Effros, J. Shi, and D. Karger, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, pp. 4413–4430, Oct. 2006.

[14] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 782–795, Oct. 2003.

[15] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, pp. 476–489, Mar. 2011.

[16] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Explicit codes minimizing repair bandwidth for distributed storage," *Proceedings of the IEEE Information Theory Workshop, Cairo, Egypt*, Jan. 2010.

[17] M. A. Maddah-Ali, S. A. Motahari, and A. K. Khandani, "Communication over MIMO X channels: Interference alignment, decomposition, and performance analysis," *IEEE Transactions on Information Theory*, vol. 54, pp. 3457–3470, Aug. 2008.

[18] S. A. Jafar and S. Shamai, "Degrees of freedom region for the MIMO X channel," *IEEE Transactions on Information Theory*, vol. 54, pp. 151–170, Jan. 2008.

[19] *The Coding for Distributed Storage Wiki*. Online available at http://tinyurl.com/storagecoding.

[20] V. R. Cadambe and S. A. Jafar, "Interference Alignment and the Degrees of Freedom for the K User Interference Channel," *IEEE Transactions on Information Theory*, vol. 54, pp. 3425–3441, Aug. 2008.

[21] A. Rawat, S. Vishwanath, A. Bhowmick, and E. Soljanin, "Update efficient codes for distributed storage," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT), St. Petersburg, Russia*, July 2011.

[22] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge University Press, 1995.

[23] V. R. Cadambe and S. A. Jafar, "Interference alignment and the degrees of freedom of wireless X networks," *IEEE Transactions on Information Theory*, vol. 55, pp. 3893 –3908, sept. 2009.