Locally Repairable Regenerating Code Constructions

Imad Ahmad, Student Member, IEEE, and Chih-Chun Wang, Senior Member, IEEE

Abstract—In this work, we give locally repairable regenerating code (LRRC) constructions that can protect the file size promised by the graph analysis of the modified family helper selection (MFHS) scheme at the minimum-bandwidth-regenerating (MBR) point.

I. INTRODUCTION

The aim of this work is to provide code constructions of locally repairable regenerating codes (LRRCs) that achieve the minimum-bandwidth-regenerating (MBR) point of the modified family helper selection (MFHS) scheme. It is worth mentioning that related existing works to LRRCs can be found in [7], [10] and also see the references in our previous work [1].

The goal of this section is to prove the existence and construction of linear codes that can protect a file of size $q$ against any $(n-k)$ simultaneous failures.

We start first by describing the notation that will be used in this section. We represent the original file by an $n \times k$ matrix $\mathbf{H}$ defined over finite field $\mathbb{F}(q)$ denoted by $\mathbb{F}$, where $q$ is a fixed finite field size satisfying $q > ndM[\mathcal{H}]$ and $\mathcal{H}$ is a finite set that will be defined shortly. The file is viewed in the following as containing $\mathcal{M}$ packets, where the packet is thought of as the smallest unit of data. Recall that we are considering the MBR point in this section. Therefore, the file size is $\mathcal{M}$ packets, the storage-per-node $\alpha = d$ packets and the repair-bandwidth per-helper $\beta = 1$ packet. Now, storage node $i$ stores $\alpha = d$ packets $\mathbf{X}^T \mathbf{Q}_i$, where $\mathbf{Q}_i$ is an $\mathcal{M} \times d$ matrix.

II. RANDOM LINEAR CODE CONSTRUCTION FOR A CLASS OF $(n,k,d,r)$ PARAMETERS

In this section, we prove the existence and construction of linear locally repairable regenerating codes (LRRCs) that achieve the MBR point of the MFHS scheme for $(n,k,d,r)$ values that satisfy that $n-d-r = 2$ and $n \mod (n-d-r) = 0$, i.e., the family size in the MFHS scheme is 2 and there are no incomplete families. The code existence proof idea and construction are inspired by the work in [1].

By this definition, matrices $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_n$ are sufficient to completely specify a code.

As in [11] and [8], the existence proof is based on applying the Schwartz-Zippel theorem.

Lemma 1: Let $Q(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_2]$ be a multivariate polynomial of total degree $d_0$ (the total degree is the maximum degree of the additive terms and the degree of a term is the sum of exponents of the variables). Fix any finite set $\mathcal{S} \subseteq \mathbb{F}$, and let $r_1, \ldots, r_n$ be chosen independently and uniformly at random from $\mathcal{S}$. Then if $Q(x_1, \ldots, x_n)$ is not equal to a zero polynomial,

$$\Pr[Q(r_1, \ldots, r_n) = 0] \leq \frac{d_0}{|\mathcal{S}|}$$

As in [11] and [8], the core idea of the proof relies on using induction to represent the dynamics of the storage problem where the induction invariant is the condition that guarantees the existence of the code. Then, the induction invariant is formulated as a product of multivariate polynomials where each polynomial is shown to be non-zero to complete the proof by invoking the Schwartz-Zippel theorem. The existence proof itself implies that we can construct linear codes randomly.

Before we give the induction invariant and show how the induction process works, we define an important set of vectors $\mathcal{H}$. We abuse the above notation and we let $\pi$ be a permutation of the storage node vector $(1, 2, \ldots, n)$ and $\pi(i)$ be its $i$-th coordinate value. Define vector $\mathbf{b}(\pi)$ such that its $i$-th coordinate value $\mathbf{b}(\pi)(i) = (d - z_i(\pi))^+$, where $z_i(\cdot)$ is as defined in the proof of [11] Proposition 12. Moreover, we define vector $\mathbf{c}(\pi)$ as the truncated version of $\mathbf{b}(\pi)$, where by truncation we mean the following: (ii) we find the smallest $m$ such that $\sum_{i=1}^{m} b_i(\pi) \geq \mathcal{M}$; (ii) we set $c_i(\pi) = b_i(\pi)$ for $i = 1$ to $(m-1)$. We then set $c_m(\pi) = \mathcal{M} - \sum_{i=1}^{m-1} b_i(\pi)$ and $c_m(\pi) = 0$ for $i = (m+1)$ to $n$. To illustrate the constructions of $\mathbf{b}(\pi)$ and $\mathbf{c}(\pi)$, consider $(n, k, d, r) = (6, 4, 3, 1)$ and $\pi = (2, 3, 4, 1, 5, 6)$. We have by [1] that $\mathcal{M} = 7$ and we can get that $\mathbf{b}(\pi) = (3, 2, 0, 0, 0, 0)$ by the definition of vector $\mathbf{b}(\pi)$. Truncating $\mathbf{b}(\pi)$ as described above, we get $\mathbf{c}(\pi) = (3, 2, 2, 0, 0, 0)$. Now, we will give an important property of $\mathbf{b}(\pi)$ and $\mathbf{c}(\pi)$ in the following lemma.

Lemma 2: For $n-d-r = 2$ and $n \mod (n-d-r) = 0$, we have that for any node permutation $\pi$, $\mathbf{b}(\pi)$ and $\mathbf{c}(\pi)$ satisfy that $b_i(\pi) \geq b_{i+1}(\pi)$ and $c_i(\pi) \geq c_{i+1}(\pi)$.

Proof: The proof is divided into two cases.

Case 1: $F(I(\pi(i)) = F(I(\pi(i+1)))$. Since $(d - z_i(\pi)) = (d - z_{i+1}(\pi))$, we have that $b_i(\pi) = b_{i+1}(\pi)$.

Case 2: $F(I(\pi(i)) \neq F(I(\pi(i+1)))$. The value $(d - z_i(\pi))$ is smallest when all nodes $\pi(1)$ to $\pi(i-1)$ are not in the

This work was supported in parts by NSF grants CCF-0845968, CNS-0905331, CCF-1422997, and ECCS-1407604.
family $FI(\pi(i))$. On the other hand, the largest $(d - z_{i+1}(\pi))$ value is when the other node in family $FI(\pi(i + 1))$ is in the nodes $\pi(1)$ to $\pi(i - 1)$. Notice, however, that we have that $FI(\pi(i)) \neq FI(\pi(i + 1))$. This fact will decrease $(d - z_{i+1}(\pi))$ by a value of 1. Therefore, the smallest $(d - z_{i+1}(\pi))$ still satisfies $(d - z_1(\pi)) \geq (d - z_{i+1}(\pi))$. Recall that $b_i(\pi) = (d - z_1(\pi))^+ = b_{i+1}(\pi) = (d - z_{i+1}(\pi))^+$ by definition. Therefore, $b_i(\pi) \geq b_{i+1}(\pi)$.

By the above two cases, we have proved that $b_i(\pi) \geq b_{i+1}(\pi)$ for any node permutation $\pi$. Since $c(\pi)$ is a truncated version of $b(\pi)$, $c(\pi)$ also satisfies that $c_i(\pi) \geq c_{i+1}(\pi)$ for any permutation $\pi$. Hence, the proof of this lemma is complete.

We are now ready to define set $H$ that will be used to write the induction invariant. Suppose we have two $n$-dimensional vectors $a$ and $b$. We say that a majorizes $b$, denoted by $a \geq b$, if $\sum_{i=1}^m a_i \geq \sum_{i=1}^m b_i$ for all $m = 1$ to $n$, where $a_i$ and $b_i$ are the $i$-th largest values in $a$ and $b$, respectively.

Now, we define set $H$ as follows. A vector $\mathbf{h} \in H$ if it satisfies the following two conditions: (i) it is an $n$-dimensional integer vector such that $0 \leq h_i \leq d$ for all $i = 1$ to $n$ and (ii) there exists a node permutation $\pi$ such that $h_{\pi(i)} \geq h_{\pi(i+1)}$ for $i = 1$ to $(n - 1)$ and $c(\pi) \geq \mathbf{h}$. Using set $H$, we write the induction invariant as

$$\prod_{\mathbf{h} \in H} \det([Q_1E_{h_1}, \ldots, Q_nE_{h_n}]) \neq 0, \quad (3)$$

where $E_x$ denotes a selection matrix such that $Q_iE_x$ holds the first $x$ columns of $Q_i$, and recall that $h_i$ is the $i$-th coordinate of vector $\mathbf{h}$.

In order to prove that there exists a code that can protect a file of size $M$, it is sufficient to prove that initially we can get matrices $Q_i$ such that the induction invariant (3) is satisfied over $\mathbb{F}$ and then show that the induction invariant can be maintained after any node arbitrary failure/repair that can happen. This is sufficient because the reconstruction (MDS) property or condition is weaker than the condition in (3). Specifically, since all possible $c(\pi)$ vectors belong to set $H$, we have that every subset of $k$ matrices from $\{Q_1, \ldots, Q_n\}$ is full rank or equivalently can reconstruct the file.

The first step thus is to prove that (3) is satisfied initially by $Q_1, \ldots, Q_n$ over $\mathbb{F}$. Notice, that the left-hand side (LHS) of (3) can be thought of as a multivariate polynomial, where the variables are the entries of $Q_1, \ldots, Q_n$. Since the size of set $H$ is finite, the degree of the polynomial in (3) is at most $d_1 = ndM|H|$ and by Schwartz-Zippel theorem we can find entries of the matrices $Q_1, \ldots, Q_n$ such that (3) is since $|\mathbb{F}| = q > d_1$.

Now, we suppose that the induction invariant in (3) is satisfied until stage $(t - 1)$ over $\mathbb{F}$. Note that in order to be able to invoke Schwartz-Zippel theorem, the LHS of (3) has to be a non-zero polynomial. It is not hard to see that for each vector $\mathbf{h} \in H$, the determinant term in (3) is a non-zero polynomial by using the column vectors of the $M \times M$ identity matrix and placing them on the first columns of $Q_1, \ldots, Q_n$. Specifically, $Q_1$ takes the first $h_1$ columns of the identity matrix, $Q_2$ takes the first $h_2$ columns of the remaining columns and so on and so forth. Since the product of non-zero polynomials is a non-zero polynomial, the LHS of (3) is a non-zero polynomial.

Without loss of generality, suppose node 1 fails and a newcomer replacing node 1 communicates with helpers $\{x_1, \ldots, x_d\} \subset D(1)$ for repair. By the nature of the repair problem and the fact that we are considering linear codes, we can write the coding matrix on the newcomer as

$$Q'_1 = [Q_{x_1}b_1, \ldots, Q_{x_d}b_d]Z, \quad (4)$$

where $b_i$ is a $d \times 1$ column vector and $Z$ is a $d \times d$ matrix representing the possible linear transformation a newcomer can apply to the $d$ received repair packets.

In this step, we want to prove that it is possible to find $b_1, \ldots, b_d$ vectors and a matrix $Z$ over $\mathbb{F}$ that satisfy (3).

In total we have $2d^2$ variables. Therefore, the degree of the polynomial on the LHS of the new condition

$$\prod_{\mathbf{h} \in H} \det([Q'_1E_{h_1}, Q_2E_{h_2}, \ldots, Q_nE_{h_n}]) \neq 0, \quad (5)$$

is at most $2d^2|H|$. Notice that $2d^2|H| \leq ndM|H|$ since $n \geq 2$ and $M \geq d$. Therefore, since $q > ndM|H|$, by Schwartz-Zippel theorem, we have that we can find $b_1, \ldots, b_n$ vectors and matrix $Z$ over $\mathbb{F}$ such that (5) is true. Assuming that the LHS of (5) is non-zero. It turns out that proving this fact is non-trivial. The remainder of this section is dedicated to showing that the LHS of (5) is a non-zero polynomial.

The proof idea for proving that the LHS of (5) is a non-zero polynomial is as follows. We still suppose that node 1 fails without loss of generality and we suppose that node 1 will repair from helpers $\{x_1, \ldots, x_d\} \subset D(1)$. Recall that it is sufficient to prove that the polynomial of the determinant for each $\mathbf{h} \in H$ is non-zero since the product of non-zero polynomials is a non-zero polynomial. Using this fact, we will consider any $\mathbf{h} \in H$ first. Then, we will prove that we can always find a vector $\mathbf{h}' \in H$ such that $h'_{i} = 0$, and there exists a subset of nodes in $\{x_1, \ldots, x_d\}$ of size $b_1$, $\{s_1, \ldots, s_{b_1}\}$, such that $h'_{s_1}, \ldots, h'_{s_{b_1}}$ satisfy $h'_{s_i} = h_{s_i} + 1$ for $i = 1$ to $b_1$.

Since $\mathbf{h}' \in H$, we have that

$$\det([Q_2E_{h'_2}, \ldots, Q_nE_{h'_n}]) \neq 0. \quad (6)$$

Now, we choose the vectors $b_1, \ldots, b_d$ such that they select the $h'_{s_i}$-th column vector from each of $Q_{s_1}, \ldots, Q_{s_{b_1}}$ and we choose $Z$ to be the $d \times d$ identity matrix. By (6), we have that

$$\det([Q_1E_{h'_1}, Q_2E_{h'_2}, \ldots, Q_nE_{h'_n}]) \neq 0. \quad (7)$$

Since we have found one combination of the entries in the vectors $b_1, \ldots, b_d$ and the matrix $Z$ such that for the considered $\mathbf{h} \in H$ the determinant is non-zero, then this polynomial is non-zero. Notice that we stated that we can find such $\mathbf{h}' \in H$ for any $\mathbf{h} \in H$. Therefore, the LHS of (5) is a non-zero polynomial. Now, we are left with proving the fact that we can always find such $\mathbf{h}' \in H$. In order to do that, we will need the following claim.

Claim 1: Consider any vector $\mathbf{h} \in H$ where $n - d - r = 2$ and $n \mod (n - d - r) = 0$. Consider a node permutation $\pi$ such that $h_{\pi(i)} \geq h_{\pi(i+1)}$. If $c(\pi) \geq \mathbf{h}$ and $h_{\pi(i)} = h_{\pi(i+1)}$ for some integer $i$, then $c(\pi') \geq \mathbf{h}$ where $\pi'$ is a node permutation
such that $\pi'(i) = \pi(i + 1), \pi'(i + 1) = \pi(i)$, and $\pi'(j) = \pi(j)$ for $j \in \{1, \ldots, i - 1, i + 2, \ldots, n\}$.

The proof of this claim is relegated to Appendix A.

We now describe procedure CONNECT that takes the vector $h \in \mathcal{H}$ as input and outputs a vector $h' \in \mathcal{H}$ that satisfies the properties discussed above. After we describe procedure CONNECT, we will prove the correctness of the procedure.

Now, we will define node classes that divide the nodes according to their values in $h$. We have $(d + 1)$ classes defined as $A_g = \{m : 1 \leq m \leq n, h_m = g\}$ for $g = 0$ to $d$. Procedure CONNECT is as follows:

1. Pick node permutation $\pi^{(0)}$ such that (i) $h_{\pi^{(0)}(i)} \geq h_{\pi^{(0)}(i+1)}$ and (ii) if $n_1 \in A_g \cap \{x_1, \ldots, x_d\}$ for some $g$ and $n_2 \in A_g \{x_1, \ldots, x_d\}$, then $p^{(0)}(n_1) < p^{(0)}(n_2)$, where $p^{(0)}$ satisfies $\pi^{(0)}(\pi^{(0)}(i)) = i$.
2. Let $h^{(0)} = h$ and let $D^{(0)} = \{x_1, \ldots, x_d\}$. Let $A_g^{(0)} = A_g$ for $g = 0$ to $d$. Note that jointly Claim 1 and the fact that $h \in \mathcal{H}$ imply that $c(\pi^{(0)}) \geq h^{(0)}$.
3. Sequentially, do the following for $t = 1$ to $h_1$:
   a) Find $x \in D^{(t-1)}$ such that (i) $h^{(t-1)}(x) \leq h^{(t-1)}(y)$ for all $y \in D^{(t-1)}$ and (ii) $p^{(t-1)}(x) < p^{(t-1)}(y)$ for all $y \in A^{(t-1)} \cap D^{(t-1)} \{x\}$.
   b) Let $h^{(t)}_i = h^{(t-1)}_i + 1$, $h^{(t)}_i = h^{(t-1)}_i - 1$, and $h^{(t)}_i = h^{(t-1)}_i$ for all $i \in \{x_1, \ldots, x_d\}$. Let $D^{(t)} = D^{(t-1)} \{x\} \cup A^{(t-1)} \{x\}$ and $A_g^{(t)} = \{m : 1 \leq m \leq n, h_m = g\}$ for $g = 0$ to $d$. Let $\pi^{(t)}$ such that (i) $\pi^{(t)}(\pi^{(t-1)}(i)) < \pi^{(t)}(\pi^{(t-1)}(j))$ whenever $i < j$ and both $\pi^{(t-1)}(i)$ and $\pi^{(t-1)}(j)$ are not equal to 1, i.e., the order of the nodes in $\pi^{(t-1)}$ is preserved for all nodes except node 1, which can move to later position in the order (have larger $p$ value), and (ii) $\pi^{(t)}$ satisfies that $\pi^{(t)}(\pi^{(t)}(i)) \geq \pi^{(t)}(\pi^{(t+1)}(i))$. This is always possible by the construction of $\pi^{(t)}$.
4. Return the output $h' = h^{(h_1)}$. Proposition 1: Procedure CONNECT is correct.

Proof: First, we need to show that the output of CONNECT $h'$ satisfies that $h'^{i} = 0$ and $\{h^{i}_i : h^{i} = h + 1, i \in \{x_1, \ldots, x_d\}\} = h_1$. We can see that by Step 3 of CONNECT, $h_1$ always decreases by 1 with every iteration and with every iteration we are always choosing a helper in $\{x_1, \ldots, x_d\}$ and adding 1 to its $h$ value. Since Step 3 is done exactly $h_1$ times, $h'$ satisfies the properties. The other important property of $h'$ that we need to show is that $h' \in \mathcal{H}$. In order to do that, we need to prove two points (i) $0 \leq h'^{i} < d$ (ii) there exists a node permutation $\pi$ such that $h'^{(i)}(\pi) \geq h'^{(i+1)}(\pi)$ and $c(\pi) \geq h'$. Since $0 \leq h'_1 \leq d$, then $h'_1 > d$ is only possible for $i \in \{x_1, \ldots, x_d\}$. Since in CONNECT, only $h_1$ (and not all $d$ nodes) of the helpers $x_1, \ldots, x_d$ will have their $h$ value incremented by 1, we need to have that at least $h_1$ of the helpers satisfy that their $h$ value is strictly less than $d$.

We will first argue that having more than two nodes of $\{x_1, \ldots, x_d\}$ with an $h$ value equal to $d$ is not possible. The reason is as follows. Suppose we have more than two nodes like that. Let these nodes be $x_1, \ldots, x_{m}$. Then, construct a node permutation $\pi$ such that the first $m$ coordinates are $x_1, \ldots, x_m$ and $h^{(i)}(i) \geq h^{(i+1)}(i)$ for $i = 2$ to $(n - 1)$. Since $n - d - r = 2$, the family size is 2 and $c_3(\pi) < d$. Thus, $c(\pi) \geq h'$ and this implies that $h' \notin \mathcal{H}$ by Claim 1. Therefore, by contradiction we have proved that we cannot have more than two nodes of $\{x_1, \ldots, x_d\}$ with $h$ values equal to $d$. Now, if $h_1 = d$, it is not possible to have any node of $\{x_1, \ldots, x_d\}$ with $h$ value $d$ since that again means that $c(\pi) \geq h'$, where $\pi$ is the same as the $\pi$ we constructed above.

If $h_1 = d - 1$, it is not possible to have more than 1 node of $\{x_1, \ldots, x_d\}$ with $h$ value $d$ for the same reason. In both cases, $0 \leq h_i' \leq d$ for $i = 1$ to $n$ since there are $h_1$ helpers with $h$ values strictly less than $d$. For $h_1 \leq d - 2$, since it is not possible to have more than 2 nodes of $\{x_1, \ldots, x_d\}$ with $h$ values equal to $d$, then there are always $h_1$ helpers with $h$ values strictly less than $d$. Therefore, in all cases $h'$ satisfies $0 \leq h_i' \leq d$ for all $i = 1$ to $n$.

At this point, we are left with proving that there exists a permutation $\pi$ such that $c(\pi) \geq h'$. We will show shortly that $\pi(h_1) \pi(h_2) \ldots \pi(h_n)$ is indeed this permutation. Stronger than that, we will prove that at the end of each iteration $i$ of CONNECT, $c(\pi^{(i)}) \geq h^{(i)}$. The proof is by contradiction. Suppose that $c(\pi^{(i)}) < h^{(i)}$. Let $l$ be the smallest $l$ such that

\[ h^{(i)}_{\pi^{(i)}(1)} + \cdots + h^{(i)}_{\pi^{(i)}(l)} > c_1(h^{(i)}) + \cdots + c_l(h^{(i)}). \]  

Notice that $l < k$. Let $l'$ be the largest integer such that $h^{(i-1)}_{\pi^{(i-1)}}(l') = h^{(i-1)}_{\pi^{(i-1)}}(l)$ and $\pi^{(i-1)}(l') \in D^{(i-1)}$, i.e., $l'$ is the largest integer such that $\pi^{(i-1)}(l')$ is a helper of $D^{(i-1)}$ in the node class $A_g^{(i-1)}$, where $g = h^{(i-1)}_{\pi^{(i-1)}}(l)$. We have the following claim based on the definition of $l'$.

Claim 2: $h^{(i)}_{\pi^{(i)}(1)} + \cdots + h^{(i)}_{\pi^{(i)}(l)} > c_1(h^{(i)}) + \cdots + c_{l'}(h^{(i)})$. We note that $l' \leq k$.

Proof: By the definition of $l$, we have that

\[ h^{(i)}_{\pi^{(i)}(1)} + \cdots + h^{(i)}_{\pi^{(i)}(l-1)} \leq c_1(h^{(i)}) + \cdots + c_{l-1}(h^{(i)}). \]

We then get by (8) that $h^{(i)}_{\pi^{(i)}(l)} > c_l(h^{(i)})$. Now, we have

\[ h^{(i)}_{\pi^{(i)}(1)} + \cdots + h^{(i)}_{\pi^{(i)}(l-1)} + h^{(i)}_{\pi^{(i)}(l)} > c_1(h^{(i)}) + \cdots + c_{l-1}(h^{(i)}) + (l' - l)h^{(i)}_{\pi^{(i)}(l')} > c_1(h^{(i)}) + \cdots + c_l(h^{(i)}) + (l' - l)c_l(h^{(i)}) \geq c_1(h^{(i)}) + \cdots + c_{l'}(h^{(i)}) + c_{l+1}(h^{(i)}) + \cdots + c_{l'}(h^{(i)}), \]

where (10) is by the fact that $h^{(i)}_{\pi^{(i)}(l)} = h^{(i)}_{\pi^{(i)}(l'+1)} = \cdots = h^{(i)}_{\pi^{(i)}(l)}$, (11) by the fact that $h^{(i)}_{\pi^{(i)}(l)} > c_l(h^{(i)})$, and (12) is by the fact that $c_l(h^{(i)}) \geq c_{l+1}(h^{(i)})$. By (12), the proof of this claim is complete.

Recall that we are proving that $h^{(i)} \in \mathcal{H}$ by contradiction. We have two cases:
Case 1: if $1 \in \{\pi^{(t)}(1), \ldots, \pi^{(t)}(l')\}$. We have that
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} \\
\geq h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} \\
\geq h^{(t)}_{\pi^{(t)}(1)} + \cdots + h^{(t)}_{\pi^{(t)}(l')} \tag{13} \\
> c_1(\pi^{(t)}) + \cdots + c_{l'}(\pi^{(t)}) \tag{14} \\
= \min\{b_1(\pi^{(t)}) + \cdots + b_1(\pi^{(t)}, M)\} \tag{15} \\
= \min\{b_1(\pi^{(0)}) + \cdots + b_{l'}(\pi^{(0)}, M)\} \tag{16} \\
= c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) \tag{17},
\]
by the fact that only node 1 moves in permutation $\pi^{(0)}$ as discussed before. Thus, by (20), we get
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} + h_{1}^{(0)} \\
> c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) + (d - v). \tag{23}
\]
If $h^{(0)}_{\pi^{(0)}(l'+1)} = h_{1}^{(0)}$, swap nodes 1 and $\pi^{(0)}(l + 1)$ in $\pi^{(0)}$ to get a new permutation $\pi'$, and then using (23) we get
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} + h_{1}^{(0)} \\
> c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) + (d - v) \tag{24} \\
= c_1(\pi') + \cdots + c_{l'}(\pi'),
\]
where we get (24) by the fact that $\{\pi^{(0)}(1), \ldots, \pi^{(0)}(l')\} = \{\pi'(1), \ldots, \pi'(l')\}$ and the other fact that $c_{l'+1}(\pi') = (d - v)$ since $\{\pi^{(0)}(1), \ldots, \pi^{(0)}(l')\} = \{\pi'(1), \ldots, \pi'(l')\}$. By Claim 1 equation (24) yields a contradiction. Now, $h_{\pi^{(0)}(l'+1)} < h_{1}^{(0)}$ is not possible since $1 \notin \{\pi^{(0)}(1), \ldots, \pi^{(0)}(l'+1)\}$. Therefore we are left with the case that $h^{(0)}_{\pi^{(0)}(l'+1)} > h_{1}^{(0)}$. In this case, we can write
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l'+1)} \\
\geq h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} + h_{1}^{(0)} + 1 \tag{18} \\
\geq h_{\pi^{(0)}(1)} + \cdots + h_{\pi^{(0)}(l')} + (d - v) + 1 \tag{19} \\
> c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) + (d - v) \tag{20} \tag{21} \\
\]
where (18) is by the fact that $h_{\pi^{(0)}(i)} \geq h_{\pi^{(0)}(i+1)}$, (18) follows by the fact that all the helper nodes in $\{\pi^{(0)}(l') + 1, \ldots, \pi^{(0)}(n)\} \cap D^{(0)}$ have been incremented by 1 so far or otherwise helper node $l$ would not have been picked by CONNECT to be incremented. This means that at least $(d - v)$ nodes not in $\{\pi^{(0)}(1), \ldots, \pi^{(0)}(l')\}$ have been incremented by 1. Now, we have two sub-cases.

Sub-case 2.1: if $1 \in \{\pi^{(0)}(1), \ldots, \pi^{(0)}(l' + 1)\}$. Then, we have that
\[
c_1(\pi^{(t)}) + \cdots + c_{l'}(\pi^{(t)}) + (d - v) = \\
c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}), \tag{22}
\]
The reason is the following. By the construction of $\pi^{(0)}$ (made possible by Claim 1 in Step 1 of CONNECT, we can see that the order of the nodes in the node permutation excluding node 1 does not change. Only node 1 may move to a later coordinate. This gives us that $\{\pi^{(0)}(1), \ldots, \pi^{(0)}(l' + 1)\} = \{\pi^{(t)}(1), \ldots, \pi^{(t)}(l' + 1)\}$. Using the fact that the order of the nodes in the permutation does not change the MBR point, in a similar fashion to Case 1, we got (21). Therefore, we get by (22) that $c(\pi^{(0)}) \neq h^{(0)}$, i.e., a contradiction with the fact that $h \in \mathcal{H}$.

Sub-case 2.2: if $1 \notin \{\pi^{(0)}(1), \ldots, \pi^{(0)}(l' + 1)\}$. We have that
\[
c_1(\pi^{(t)}) + \cdots + c_{l'}(\pi^{(t)}) = c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) \tag{23},
\]
by the fact that only node 1 moves in permutation $\pi^{(0)}$ as discussed before. Thus, by (20), we get
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} + h_{1}^{(0)} \\
> c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) + (d - v). \tag{24}
\]
If $h^{(0)}_{\pi^{(0)}(l'+1)} = h_{1}^{(0)}$, swap nodes 1 and $\pi^{(0)}(l + 1)$ in $\pi^{(0)}$ to get a new permutation $\pi'$, and then using (23) we get
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} + h_{1}^{(0)} \\
> c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) + (d - v) \tag{25} \\
= c_1(\pi') + \cdots + c_{l'}(\pi'),
\]
where we get (25) by the fact that $\{\pi^{(0)}(1), \ldots, \pi^{(0)}(l')\} = \{\pi'(1), \ldots, \pi'(l')\}$ and the other fact that $c_{l'+1}(\pi') = (d - v)$ since $\{\pi^{(0)}(1), \ldots, \pi^{(0)}(l')\} = \{\pi'(1), \ldots, \pi'(l')\}$. By Claim 1 equation (25) yields a contradiction. Now, $h_{\pi^{(0)}(l'+1)} < h_{1}^{(0)}$ is not possible since $1 \notin \{\pi^{(0)}(1), \ldots, \pi^{(0)}(l' + 1)\}$. Therefore we are left with the case that $h^{(0)}_{\pi^{(0)}(l'+1)} > h_{1}^{(0)}$. In this case, we can write
\[
h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l'+1)} \\
\geq h^{(0)}_{\pi^{(0)}(1)} + \cdots + h^{(0)}_{\pi^{(0)}(l')} + h_{1}^{(0)} + 1 \tag{18} \tag{19} \tag{20} \tag{21} \\
\geq h_{\pi^{(0)}(1)} + \cdots + h_{\pi^{(0)}(l')} + (d - v) + 1 \tag{19} \\
\geq c_1(\pi^{(0)}) + \cdots + c_{l'}(\pi^{(0)}) + (d - v) \tag{20} \tag{22} \\
\]
III. Exact Repair Linear Code Construction for $(n, k, d, r) = (6, 3, 2, 1)$

In the following, we present a linear exact repair LRRC construction that can achieve the MBR point of the MFHS scheme for $(n, k, d, r) = (6, 3, 2, 1)$. By Proposition 12, we have that $M = 4$ packets and $a = 2$ packets for $\beta = 1$ packet. Recall that according to the MFHS scheme, we can divide the storage nodes into two complete families, family 1 consisting of $\{1, 2, 3\}$ and family 2 consisting of $\{4, 5, 6\}$.

Step 1: Generate a $(6, 4)$ systematic MDS code over a finite field $\mathbb{F}$. Denote the generating matrix of this MDS code by
\[
G = \begin{pmatrix}
1 & 0 & 0 & 0 & a_1 & a_2 \\
0 & 1 & 0 & 0 & a_2 & a_3 \\
0 & 0 & 1 & 0 & b_1 & b_2 \\
0 & 0 & 0 & 1 & b_1 & b_2
\end{pmatrix} \tag{28}
\]
packets are linearly independent, we can reconstruct the two packets of node 1. In a similar fashion we repair node 1 for any node 5 is unavailable. We access node 4 and get the packet of nodes 5 and 6 given any node unavailability is also similar.

We divide the proof into two cases:

Case 1: $h_{\pi(1)} + \cdots + h_{\pi(i)} = c_1(\pi) + \cdots + c_i(\pi)$. Since $c(\pi) \geq h$, we have that $c(\pi) \geq h$. Recall that $c(\pi)$ is the truncated version of vector $b(\pi)$. If $F I(\pi(i)) = F I(\pi(i + 1))$, we have that $b_1(\pi) = b_i(\pi) = b_i(\pi') = b_{i+1}(\pi')$. Thus, this case is trivial.

If $F I(\pi(i)) \neq F I(\pi(i + 1))$, we have that $b_1(\pi') \geq b_{i+1}(\pi)$ (by the definition of function $z_i$). Now, if $b_1(\pi') = c_i(\pi')$, then we have that $b_1(\pi') = c_i(\pi') \geq c_{i+1}(\pi)$. If on the other hand, $b_1(\pi') > c_i(\pi')$, since $\sum_{m=1}^{i-1} b_m(\pi) = \sum_{m=1}^{i-1} b_m(\pi')$, then $c_i(\pi') = c_i(\pi') \geq c_{i+1}(\pi)$ where the first inequality follows by the fact that $c_i(\pi') > c_i(\pi')$ is not possible since truncated vector $c(\pi)$. A contradiction, and the second inequality follows by Lemma 2. By the above arguments, we get that $c_i(\pi') \geq c_{i+1}(\pi)$. Thus, we get that $h_{\pi(i+1)} \leq c_{i+1}(\pi) \leq c_i(\pi')$. Therefore,

$$h_{\pi(1)} + \cdots + h_{\pi(i)} = h_{\pi(1')} + \cdots + h_{\pi(i')} \leq c_1(\pi') + \cdots + c_i(\pi').$$

Now, we have that $\pi'(i + 1) = \pi(i)$. Originally, we had

$$h_{\pi(i)} + \cdots + h_{\pi(i+1)} \leq c_1(\pi) + \cdots + c_{i+1}(\pi).$$

Since

$$b_1(\pi) + \cdots + b_{i+1}(\pi) = b_1(\pi') + \cdots + b_{i+1}(\pi'),$$

as we saw in the proof of the MBR point in Proposition 7 when $d + 1 \geq k$, we get

$$\min\{b_1(\pi) + \cdots + b_{i+1}(\pi), \mathcal{M}\} = \min\{b_1(\pi') + \cdots + b_{i+1}(\pi'), \mathcal{M}\}.$$
Hence, \( c' \geq h \) for Case 1.

Case 2: \( h_{\pi(1)} + \cdots + h_{\pi(i)} < c_1(\pi) + \cdots + c_i(\pi) \). We have \( b_1(\pi') \geq b_{i+1}(\pi') \geq b_i(\pi) - 1 \), where the first inequality follows by Lemma \ref{lemma:1} and the second inequality by the fact that \( F1(\pi(i)) \neq F1(\pi(i+1)) \). We now want to show that this implies that \( c_i(\pi') \geq c_i(\pi) - 1 \).

- If \( b_{i-1}(\pi) > c_{i-1}(\pi) \), then \( c_i(\pi) = 0 \) and we thus trivially have \( c_i(\pi') \geq c_i(\pi) - 1 \).
- If \( b_i(\pi) > c_i(\pi) \), then \( b_i(\pi') \geq b_i(\pi) - 1 \geq c_i(\pi) \).
  
  Since \( \sum_{m=1}^{i-1} b_m(\pi) = \sum_{m=1}^{i-1} b_m(\pi') \), \( c_i(\pi') = c_i(\pi) \geq c_i(\pi) - 1 \).

Therefore, we indeed have that \( c_i(\pi') \geq c_i(\pi) - 1 \).

By this fact, we have that

\[
\begin{align*}
b_{\pi(1)}(\pi') + \cdots + b_{\pi(i)}(\pi') & = b_{\pi'(1)}(\pi') + \cdots + b_{\pi'(i)}(\pi') \\
& \leq c_1(\pi') + \cdots + c_i(\pi'),
\end{align*}
\]

Now, by the same argument as in Case 1, we get that

\[
b_{\pi(1)}(\pi) + \cdots + b_{\pi(i+1)}(\pi) \leq c_1(\pi) + \cdots + c_{i+1}(\pi).
\]

Therefore, we get that \( c(\pi') \geq h \) for Case 2 too. Hence, the proof of this claim is complete.

**References**

[1] I. Ahmad and C.-C. Wang, “Locally repairable regenerating codes: Node unavailability and the insufficiency of stationary local repair,” [Online]. Available: arXiv:1505.05124 [cs.IT].

[2] ———, “When can helper node selection improve regenerating codes? Part I: Graph-based analysis,” [Online]. Available: arXiv:1604.08231 [cs.IT].

[3] ———, “When can helper node selection improve regenerating codes? Part II: An explicit exact-repair code construction,” [Online]. Available: arXiv:1604.08230 [cs.IT].

[4] ———, “When and by how much can helper node selection improve regenerating codes?” in Proc. 52nd Ann. Allerton Conf. Communication, Control, and Computing, Monticello, IL, Oct. 2014, pp. 459–466.

[5] ———, “When locally repairable codes meet regenerating codes — what if some helpers are unavailable,” in Proc. IEEE Int. Symp. Information Theory (ISIT), Hong Kong, Jun. 2015, pp. 849–853.

[6] H. D. L. Hollmann, “On the minimum storage overhead of distributed storage codes with a given repair locality,” in Proc. IEEE Int. Symp. Information Theory (ISIT), Honolulu, HI, Jun. 2014, pp. 1041–1045.

[7] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar, “Codes with local regeneration and erasure correction,” IEEE Trans. Inf. Theory, vol. 60, no. 8, pp. 4637–4660, 2014.

[8] R. Koetter and M. Médard, “An algebraic approach to network coding,” IEEE/ACM Transactions on Networking, vol. 11, no. 5, pp. 782–795, 2003.

[9] R. Motwani and P. Raghavan, Randomized algorithms. Chapman & Hall/CRC, 2010.

[10] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, “Optimal locally repairable and secure codes for distributed storage systems,” IEEE Trans. Inf. Theory, vol. 60, no. 1, pp. 212–236, 2014.

[11] Y. Wu, “Existence and construction of capacity-achieving network codes for distributed storage,” IEEE J. Select. Areas Commun., vol. 28, no. 2, pp. 277–288, 2010.