

**Algorithm 1** Construction of a set  $\mathcal{A}$  with  $H(\mathbf{c}_{\mathcal{A}}) < ML$  for a shift-XOR LRC

---

```

1: Set  $\mathcal{A}_0 = \emptyset$  and  $i = 1$ 
2: while  $H(\mathbf{c}_{\mathcal{A}_{i-1}}) < ML$  do
3:   Pick  $j_i \notin \mathcal{A}_{i-1}$  s.t.  $|\Gamma(j_i) \setminus \mathcal{A}_{i-1}| \geq \delta - 1$ 
4:   if  $H(\mathbf{c}_{\mathcal{A}_{i-1}}, \mathbf{c}_{\Gamma(j_i)}) < ML$  then
5:     set  $\mathcal{A}_i = \mathcal{A}_{i-1} \cup \Gamma(j_i)$ 
6:   else if  $H(\mathbf{c}_{\mathcal{A}_{i-1}}, \mathbf{c}_{\Gamma(j_i)}) \geq ML$  then
7:      $\mathcal{T} = \arg \max_{\mathcal{T}' \subset \Gamma(j_i); H(\mathbf{c}_{\mathcal{A}_{i-1}}, \mathbf{c}_{\mathcal{T}'}) < ML} |\mathcal{T}'|$ 
8:     if  $\mathcal{T} = \emptyset$  then
9:       break
10:    else
11:      set  $\mathcal{A}_i = \mathcal{A}_{i-1} \cup \mathcal{T}$ 
12:    break
13:  end if
14: end if
15:   $i = i + 1$ 
16: end while
17: Output:  $\mathcal{A} = \mathcal{A}_{i-1}$ 

```

---

*Proof of Theorem 1.* Let file  $\mathbf{X}$  consist of  $M$  sequences, where each sequence contains  $L$  bits that are independent and identically distributed (i.i.d.) uniform random variables over the binary field. The (binary) entropy of this file is

$$H(\mathbf{X}) = ML. \quad (3)$$

For a general shift-XOR code that encodes the file  $\mathbf{X}$  into  $n$  code blocks  $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n]$ , where by the independence bound, each coded block has entropy

$$H(\mathbf{c}_i) \leq \alpha L + S. \quad (4)$$

By the definition of  $(r, \delta)$  locality, for any indices  $j \in [n]$ , there exists a set of indices  $\Gamma(j) \subseteq [n]$ , where  $j \in \Gamma(j)$ ,  $|\Gamma(j)| \leq r + \delta - 1$  and any code block indexed by this group can be reconstructed from any other  $r$  blocks in the same group. Then, we can deduce that  $H(\mathbf{c}_{\Gamma(j)}) \leq r(\alpha L + S)$ , where  $\mathbf{c}_{\Gamma(j)} \triangleq \{\mathbf{c}_i : i \in \Gamma(j)\}$ .

As the original file can be reconstructed from any  $k$  nodes, for any subset  $\mathcal{A} \subseteq [n]$  with  $|\mathcal{A}| \geq k$ , we have  $H(\mathbf{c}_{\mathcal{A}}) = ML$ . Then we have

$$k \geq 1 + \max_{H(\mathbf{c}_{\mathcal{A}}) < ML} |\mathcal{A}|. \quad (5)$$

To establish a lower bound on  $k$ , we follow the approach in [1], [9], [21] to construct the largest possible subset  $\mathcal{A} \subseteq [n]$  of indices that the corresponding blocks cannot reconstruct the original file. It is achieved by adapting an algorithm from [9], [21], as given in Algorithm 1. The algorithm starts with  $\mathcal{A}_0 = \emptyset$ , and iteratively expands  $\mathcal{A}_{i-1}$  to  $\mathcal{A}_i$  by adding at most  $r + \delta - 1$  elements. For each iteration  $i$ , we define:

$$a_i = |\mathcal{A}_i| - |\mathcal{A}_{i-1}| \quad (6)$$

$$h_i = H(\mathbf{c}_{\mathcal{A}_i}) - H(\mathbf{c}_{\mathcal{A}_{i-1}}) \quad (7)$$

Let  $\ell$  denote the number of iterations before termination, i.e.,  $\mathcal{A} = \mathcal{A}_{\ell}$ . Then the final set size and entropy can be expressed as the sum of their increments:

$$|\mathcal{A}| = |\mathcal{A}_{\ell}| = \sum_{i=1}^{\ell} a_i \quad (8)$$

$$H(\mathbf{c}_{\mathcal{A}}) = H(\mathbf{c}_{\mathcal{A}_{\ell}}) = \sum_{i=1}^{\ell} h_i \quad (9)$$

In step 3, such a  $j_i$  always exists, since otherwise, by the  $(r, \delta)$  locality,  $\mathbf{c}_{\mathcal{A}_{i-1}}$  can recover all other coded blocks not indexed in  $\mathcal{A}_{i-1}$  and hence  $H(\mathbf{c}_{\mathcal{A}_{i-1}}) = ML$ , which is a contradiction. Therefore, the algorithm can terminate at either step 9 or step 12, which are analyzed separately:

*Case 1:* Suppose that the algorithm terminates at step 9, i.e., after adding  $\Gamma(j_{\ell})$  to  $\mathcal{A}_{\ell-1}$  in the  $\ell$ -th iteration. For each iteration  $i \leq \ell$ , we analyze the entropy increment  $h_i$  as follows:

$$\begin{aligned}
h_i &= H(\mathbf{c}_{\mathcal{A}_i}) - H(\mathbf{c}_{\mathcal{A}_{i-1}}) \\
&= H(\mathbf{c}_{\mathcal{A}_{i-1} \cup (\mathcal{A}_i \setminus \mathcal{A}_{i-1})}) - H(\mathbf{c}_{\mathcal{A}_{i-1}}) \\
&= H(\mathbf{c}_{\mathcal{A}_{i-1}}) + H(\mathbf{c}_{\mathcal{A}_i \setminus \mathcal{A}_{i-1}} | \mathbf{c}_{\mathcal{A}_{i-1}}) - H(\mathbf{c}_{\mathcal{A}_{i-1}}) \\
&= H(\mathbf{c}_{\mathcal{A}_i \setminus \mathcal{A}_{i-1}} | \mathbf{c}_{\mathcal{A}_{i-1}}) \\
&\leq (a_i - \delta + 1)(\alpha L + S).
\end{aligned} \quad (10)$$

The last inequality holds as follow: in iteration  $i$ ,  $a_i = |\Gamma(j_i) \setminus \mathcal{A}_{i-1}| \geq \delta - 1$  new blocks are added to  $\mathcal{A}_{i-1}$ . Among the  $a_i$  newly added blocks in iteration  $i$ , at least  $\delta - 1$  blocks are deterministic functions of the other blocks indexed in  $\Gamma(j_i)$ . Therefore, these  $\delta - 1$  blocks do not contribute to the entropy increment, leaving at most  $(a_i - \delta + 1)$  blocks that can contribute at most  $(\alpha L + S)$  each to the entropy.

From (10), we can derive a lower bound on the size increment:

$$a_i \geq \frac{h_i}{\alpha L + S} + \delta - 1. \quad (11)$$

Substituting this into (8), we obtain:

$$\begin{aligned}
|\mathcal{A}| &= |\mathcal{A}_{\ell}| = \sum_{i=1}^{\ell} a_i \\
&\geq \sum_{i=1}^{\ell} \left( \frac{h_i}{\alpha L + S} + \delta - 1 \right) \\
&= \frac{1}{\alpha L + S} \sum_{i=1}^{\ell} h_i + (\delta - 1)\ell
\end{aligned} \quad (12)$$

$$= \frac{1}{\alpha L + S} H(\mathbf{c}_{\mathcal{A}}) + (\delta - 1)\ell \quad (13)$$

We now give a lower bound on  $H(\mathbf{c}_{\mathcal{A}})$  and  $\ell$ . Since the algorithm is exiting, we know that no additional index  $j \notin \mathcal{A}$  can be added to our current index set  $\mathcal{A}$  without violating  $H(\mathbf{c}_{\mathcal{A}}) < ML$ . This means that  $H(\mathbf{c}_{\mathcal{A}})$  must be sufficiently close to  $ML$ . Specifically, we argue that

$$H(\mathbf{c}_{\mathcal{A}}) \geq ML - (\alpha L + S). \quad (14)$$

This holds because if we assume otherwise, i.e.,  $H(\mathbf{c}_A) \leq ML - (\alpha L + S) - \epsilon$  for any  $\epsilon > 0$ , then we could always find a new index  $j \notin A$  such that adding the corresponding coded block  $\mathbf{c}_j$  would increase the total entropy by at most  $(\alpha L + S)$ , keeping the total entropy below  $ML$ . This contradicts the exit condition of the algorithm.

For the lower bound on  $\ell$ , we analyze the entropy increment in each iteration. From the locality constraint, we know  $a_i - \delta + 1 \leq r$ , which together with (10) implies that  $h_i \leq r(\alpha L + S)$ . This means each iteration can contribute at most  $r(\alpha L + S)$  to the total entropy increment. Furthermore, from (14) we know that the total entropy increment must reach at least  $ML - (\alpha L + S)$ . Therefore:

$$\ell \geq \left\lceil \frac{H(\mathbf{c}_A)}{r(\alpha L + S)} \right\rceil \geq \left\lceil \frac{M}{r(\alpha + \frac{S}{L})} \right\rceil - 1 \quad (15)$$

Combining (13), (14), and (15) yields:

$$|\mathcal{A}_\ell| \geq \left\lceil \frac{M}{\alpha + \frac{S}{L}} \right\rceil - 1 + \left( \left\lceil \frac{M}{r(\alpha + \frac{S}{L})} \right\rceil - 1 \right) (\delta - 1). \quad (16)$$

*Case 2:* Here, the algorithm terminates at step 12 in the  $\ell$ -th iteration because:

$$H(\mathbf{c}_{\mathcal{A}_{\ell-1} \cup \Gamma(j_\ell)}) \geq ML. \quad (17)$$

Since each iteration increases the entropy by at most  $r(\alpha L + S)$ , we have:

$$\ell \geq \left\lceil \frac{ML}{r(\alpha L + S)} \right\rceil. \quad (18)$$

The size increments are bounded as in *Case 1*:

$$a_i \geq \frac{h_i}{\alpha L + S} + \delta - 1, \quad \text{for } i \leq \ell - 1. \quad (19)$$

Note that in the last iteration, we only add a subset  $\mathcal{T} \subset \Gamma(j_\ell)$  rather than the set, thus we cannot guarantee  $|\mathcal{T}| \geq \delta - 1$  and the existence of  $\delta - 1$  deterministic blocks. For the last iteration,

$$a_\ell \geq \frac{h_\ell}{\alpha L + S}. \quad (20)$$

Combining (8), (14), (18), (19), and (20), we obtain:

$$\begin{aligned} |\mathcal{A}_\ell| &= \sum_{i=1}^{\ell} a_i \\ &\geq \sum_{i=1}^{\ell-1} \left( \frac{h_i}{\alpha L + S} + \delta - 1 \right) + \frac{h_\ell}{\alpha L + S} \\ &= \frac{1}{\alpha L + S} H(\mathbf{c}_A) + (\ell - 1)(\delta - 1) \\ &\geq \left\lceil \frac{M}{\alpha + \frac{S}{L}} \right\rceil - 1 + \left( \left\lceil \frac{M}{r(\alpha + \frac{S}{L})} \right\rceil - 1 \right) (\delta - 1). \end{aligned} \quad (21)$$

Finally, combining (5), (16), and (21), we establish the lower bound:

$$k \geq \left\lceil \frac{M}{\alpha + \frac{S}{L}} \right\rceil + \left( \left\lceil \frac{M}{r(\alpha + \frac{S}{L})} \right\rceil - 1 \right) (\delta - 1). \quad \square$$

*Proof of Theorem 2.* Each sequence with subscript  $j$  in the fail node of repair group  $i, i \in [N]$  is contained in a  $(\delta - 1, r)$  shift-XOR system

$$[p_j^1, p_j^2, \dots, p_j^{\delta-1}]^T = \mathbf{\Omega}[y_j^1, \dots, y_j^r]^T,$$

where  $j \in [(i - 1)(r + \delta - 1), i(r + \delta - 1)], i \in [N]$ . We have  $\sum_{i(r+\delta-1)}^{(i-1)(r+\delta-1)} \sigma_j = r(\delta - 1)$ , which is independent of helper node selection. The recovery of sequences in  $\mathbf{Y}$  requires  $(r - 1)r(\delta - 1)L$  XOR operations. Generating local parity sequences requires an additional  $(r - 1)(\delta - 1)(L + O(r\delta))$  XOR operations, totaling  $(r^2 - 1)(\delta - 1)(L + O(r\delta))$  XOR operations.  $\square$

*Proof of Theorem 3.* First, we analyze the difference between  $B$  and  $k$ . From (2), we have  $B \triangleq \left\lceil \frac{M}{\alpha} \right\rceil + \left( \left\lceil \frac{M}{r\alpha} \right\rceil - 1 \right) (\delta - 1)$ . Substituting  $k = r_\delta \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) + t$  and simplifying:

$$\begin{aligned} B - k &= \left\lceil \frac{r(r_\delta \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) + t)}{r_\delta} \right\rceil + \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) (\delta - 1) \\ &\quad - [r_\delta \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) + t] \\ &= r \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) + \left\lceil \frac{rt}{r_\delta} \right\rceil + \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) (\delta - 1) \\ &\quad - r_\delta \left( \left\lceil \frac{k}{r_\delta} \right\rceil - 1 \right) - t \end{aligned}$$

Therefore,

$$B - k = \left\lceil \frac{rt}{r_\delta} \right\rceil - t \quad (22)$$

Now we prove the bounds:

1) Lower bound: Since  $\frac{rt}{r_\delta} = \frac{rt}{r+\delta-1} < t$ , from (22) we have  $B - k \leq 0$ .

2) Upper bound: Using the properties of the ceiling function and the fact that  $1 \leq t < r_\delta$ :

$$\begin{aligned} B - k &= \left\lceil \frac{rt}{r_\delta} \right\rceil - t \geq \frac{rt}{r_\delta} - t \\ &= -t \left( \frac{\delta - 1}{r_\delta} \right) \geq -(r_\delta - 1) \left( \frac{\delta - 1}{r_\delta} \right) > -(\delta - 1) \end{aligned}$$

Combining these bounds and noting that both  $B$  and  $k$  are integers:

$$0 \leq k - B \leq \delta - 2$$

For the equality condition  $B = k$ , from (22), we need:

$$\left\lceil \frac{rt}{r_\delta} \right\rceil = t \Leftrightarrow \frac{rt}{r_\delta} > t - 1 \Leftrightarrow r > (\delta - 1)(t - 1)$$

When  $\delta = 2$ , since  $r_\delta = r + 1$ , we have  $\frac{rt}{r+1} > t - 1$  always holds for  $t \geq 1$ , implying  $k = B$ .  $\square$

*Proof of Theorem 4.* When  $b = 0$ , for each codeword, accessing any  $r$  nodes from each of the  $q$  parity groups allows recovery of all  $rg^F$  global parity sequences and computation of new global parity sequences. This requires accessing  $\lambda q r$

nodes in the initial codewords and  $g^F$  nodes in the final codeword for data writing.

When  $b \neq 0$  and  $g^F \leq b$ , all required global parity sequences are in the mixed group. By accessing any  $r$  nodes in the mixed group, we need  $\lambda r$  node accesses in the initial codewords. Additionally, regrouping the systematic sequences in the mixed group of each initial codeword requires accessing  $\lambda(r_\delta - d)$  nodes in the final codeword.

When  $b \neq 0$  and  $g^F > b$ : If  $h = 0$ , the required global parity sequences span over  $(q+1)$  repair groups, requiring  $\lambda(q+1)r$  node accesses in the initial codewords, plus  $\lambda(r_\delta - b)$  nodes in the final codeword for regrouping.

If  $h > 0$ , the required global parity sequences span over  $(q+2)$  repair groups, necessitating  $\lambda(q+2)r$  node accesses in the initial codewords, plus  $\lambda(r_\delta - b)$  nodes in the final codeword for regrouping. Due to the cyclic shift pattern, collecting global parity sequences with the same subscript requires accessing  $r$  nodes within a repair group, making this strategy consistently more efficient than direct access.

In all cases, we need additional  $g^F$  nodes access for writing the new global parity sequences.  $\square$

*Proof of Theorem 5.* From [17], for an  $(n, k)$  Two-Tone shift-XOR system, the storage overhead is at least

$$\begin{cases} \frac{(n^2-1)(k-1)}{4}, & \text{if } n \text{ is odd} \\ \frac{n^2(k-1)}{4}, & \text{if } n \text{ is even} \end{cases}$$

with equality if and only if using a reflected Vandermonde matrix.

For the global parity generation, matrix  $\Phi$  can be viewed as  $r$  separate  $(n-k, k)$  shift-XOR systems. Each system has a storage overhead of at least:

$$S(\Phi) = (k-1)\eta_1$$

where  $\eta_1 = \frac{(n-k)^2 - \alpha_1}{4}$  and  $\alpha_1 = 1$  if  $n-k$  is odd (0 if even).

Similarly, the local parity generation through matrix  $\Omega$  can be viewed as  $n$  separate  $(\delta-1, r)$  shift-XOR systems, where

$$S(\Omega) = (r-1)\eta_2$$

where  $\eta_2 = \frac{(\delta-1)^2 - \alpha_2}{4}$  and  $\alpha_2 = 1$  if  $\delta-1$  is odd (0 if even).

Since the local parity generation operates on shifted sequences from  $\mathbf{Y}$ , it introduces an extra storage overhead of  $(\delta-1)S(\Phi)$ .

Therefore, the total storage overhead is  $S' = (r + \delta - 1)S(\Phi) + nS(\Omega)$ , which equals  $\hat{S}$  if and only if both  $\Phi$  and  $\Omega$  are Reflected Vandermonde matrices.  $\square$