Codes with Locality

Sushanta Paudyal

December 15, 2014

We assume that the reader is familiar with the basics of Coding Theory.

1 Introduction

Consider a linear $[n, k, d]_q$ code over $\mathbb{F}_q$; call this code $\mathscr{C}$. We say that the ith symbol of a codeword in $\mathscr{C}$ has locality r if this symbol can be reconstructed from r other symbols of the codeword. In this report, we derive a lower bound on the length of $\mathscr{C}$ with the added constraint of locality.

Next, we add another constraint named availability and derive a lower bound on the length of $\mathscr{C}$ with this constraint. We say that the jth symbol of a codeword in $\mathscr{C}$ has availability t if this symbol can be reconstructed from t disjoint groups of other symbols; the size of each of the t groups is at most r. So, a symbol with availability equal to 1 has locality r.

Recall that the length of a linear code is bounded as $n \geq k + d - 1$; this is referred to as the Singleton bound. We will see that with additional constraints like locality and availability, the bounds become larger, as we would expect.

2 Preliminaries

Let $\mathscr{C}$ be an $[n, k, d]_q$ linear code over $\mathbb{F}_q$. We describe $\mathscr{C}$ (as in [1]) through a set of vectors: $C = \{\vec{c_1}, \vec{c_2}, \cdots, \vec{c_n}\}$, where each vector ($\vec{c_i} \in \mathbb{F}_q^k$) is a column of the generator matrix of $\mathscr{C}$.

The following fact provides a bound for the minimum distance of $\mathscr{C}$.

Fact 1. The code $\mathscr{C}$ has a distance d if and only if for every $S \subseteq C$ such that $\mathrm{Rank}(S) \leq k - 1$,

$$|S| \leq n - d.$$

2.1 Locality

We now formally define locality, abbreviated as Loc.

Definition 1. For $\vec{c_i} \in C$, we define $\text{Loc}(\vec{c_i})$ to be the smallest integer r for which there exists a set $R \subseteq [n] \setminus \{i\}$ of cardinality r such that

$$\vec{c_i} = \sum_{j \in R} \lambda_j \vec{c_j}. \tag{1}$$

We further define $\text{Loc}(C) = \max_{i \in [n]} \text{Loc}(c_i)$, where $[n]$ is a shorthand for the set: $\{1, 2, \cdots, n\}$.

We elaborate further on the preceding definition. Recall that $\vec{c_i}$ is a column of the generator matrix of $\mathscr{C}$. Let $\vec{x}$ be an information vector that needs to be encoded. Then, the scalar product $\vec{x} \cdot \vec{c_i}$ gives us a corresponding encoded symbol (y_i) from the encoded vector. Hence, (1) can be rewritten as

$$\begin{aligned} \vec{x} \cdot \vec{c_i} &= \sum_{j \in R} \lambda_j \vec{x} \cdot \vec{c_j}, \\ y_i &= \sum_{j \in R} \lambda_j y_j. \end{aligned}$$

Put in words, an encoded symbol is a linear combination of r other symbols, which is what locality intuitively means.

The authors of [1] have liberally extended the notion of a *rank* of a matrix in the definition that follows.

Definition 2. We say that a code $\mathscr{C}$ has information locality r if there exists a set $I (\subseteq C)$ of full rank such that $\text{Loc}(\vec{c}) \leq r$ for all $\vec{c} \in I$.

Next, we introduce a hypergraph construction that captures linear dependencies of elements in the set C.

Consider a hypergraph $H(V, E)$. Let the set of vertices be $V = \{1, 2, \cdots, n\}$, such that a vertex in V corresponds to an element in C. We have a hyperedge for each set of linearly dependent elements of size not exceeding $r + 1$. Symbolically,

$$E = \left\{ S : S \subseteq V, \qquad |S| \leq r + 1, \qquad \exists \lambda_i \neq 0 \text{ s.t.} \sum_{i \in S} \lambda_i \vec{c_i} = 0 \right\}.$$

The remainder of this section will give a background for codes with an added constraint called availability. The reader may choose to skip to Theorem 1 (next section) at this point.

2.2 Availability

We define something called an (n, k, r, t)-LRC (Locally Repairable Code), as in [3], below.

Definition 3. An (n, k, r, t)-LRC satisfies the following three properties:

1. For each encoded information (systematic) symbol y_i, $i \in [k]$, there exist t sets $\Gamma_1(y_i), \Gamma_2(y_i), \cdots, \Gamma_t(y_i) \subset [n] \setminus \{i\}$, such that y_i is a function of the encoded symbols indexed by $\Gamma_j(y_i)$; $j \in [t]$.

2. $|\Gamma_j(y_i)| \leq r$, for all $i \in [k]$, $j \in [t]$.

3. $\Gamma_j(y_i) \cap \Gamma_l(y_i) = \varnothing$ for all $i \in [k]$ and $j \neq l \in [t]$.

Put in words, we are now considering codes where each encoded information symbol can be reconstructed from t disjoint sets, with each set containing r symbols.

By t, we denote the availability of the code: the number of 'available' sets from which the encoded symbol under question can be recovered. We ask the reader to think about what happens when $t = 1$.

Next, we introduce the concept of repair groups, which we will use in Theorem 2 (next section). By a repair group, we denote the following set: $\Gamma_j(y_i) \cup \{i\}$; $\Gamma_j(y_i)$ was introduced earlier. We denote by m the total number of distinct repair groups.

Example 1. Consider a $(7, 3, 2, 2)$-LRC with the following encoding of $\vec{x} = (m_1, m_2, m_3)$:

$$\vec{y} = (m_1,\ m_2,\ m_3,\ m_1,\ m_1 + m_2,\ m_2 + m_3,\ m_1 + m_3).$$

This code satisfies Definition 3 with

$$\begin{aligned} \Gamma_1(y_1) &= \{4\}, \quad \Gamma_2(y_1) = \{2, 5\}, \\ \Gamma_1(y_2) &= \{1, 5\}, \quad \Gamma_2(y_2) = \{3, 6\}, \\ \Gamma_1(y_3) &= \{2, 6\}, \quad \Gamma_3(y_3) = \{1, 7\}. \end{aligned}$$

In this example, symbols y_1, y_2, and y_3 have availability 2, as shown by the available sets Γ_1 and Γ_2 for each y_i. Also notice that $|\Gamma_j(y_i)| \leq 2$, implying that the locality for each symbol is 2 (see point 2 in Definition 3). We will now construct repair groups and introduce a matrix representing those repair groups.

For this example, we have the following distinct repair groups:

- repair group 1: $\{1, 4\}$,

- repair group 2: $\{1,2,5\}$,
- repair group 3: $\{2,3,6\}$,
- repair group 4: $\{1,3,7\}$.

We have 4 distinct repair groups; we will represent these by a $k \times m$ matrix $\mathbf{R}$ such that each row represents one of the k encoded information symbols and each column represents a repair group.

The matrix $\mathbf{R}$ for our example is the following:

$$\mathbf{R} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

The matrix may be read as: repair group 1 (i.e. column 1 of $\mathbf{R}$) has information symbol y_1 associated with it, repair group 2 (column 2) has information symbols y_1 and y_2 associated with it, and so on. ■

3 Lower bound on length of codes with added requirements

The first requirement we add is locality. We compute a lower bound on the length of a linear code with a given information locality in the following theorem.

Theorem 1. For any $[n,k,d]_q$ linear code with information locality r,

$$n \geq k + d + \left\lceil \frac{k}{r} \right\rceil - 2.$$

Proof. We see from Fact 1 (section 2) that

$$n - d \geq |S|,$$

where $S \subseteq C$ such that $Rank(S) \leq k-1$. We first construct a set S that satisfies said properties; Algorithm 1 constructs such a set.

Put in words, the algorithm picks a vector $\vec{c}$ in C that is not yet contained in S and puts a hyperedge (section 2.1) containing $\vec{c}$ into S, letting S grow, until the rank of S exceeds $k-2$.

We define the following for our analysis:

- s_i measures the increase in the size of S_i in Algorithm 1,

Algorithm 1

1: Let $i = 1$, $S_0 = \{\}$.
2: **while** $Rank(S_{i-1}) \leq k - 2$ **do**
3: Pick $\vec{c_i} \in C \setminus S_{i-1}$ s.t. there is a hyperedge T_i in H containing $\vec{c_i}$.
4: **if** $Rank(S_{i-1} \cup T_i) < k$ **then** set $S_i = S_{i-1} \cup T_i$.
5: **else** pick $T' \subset T_i$ s.t. $Rank(S_{i-1} \cup T') = k-1$ and set $S_i = S_{i-1} \cup T'$.
6: Increment i.

- t_i measures the increase in the rank of S_i in Algorithm 1.

Observe the following:

$$s_i = |S_i| - |S_{i-1}|.$$

$$|S_l| = \sum_{i=1}^{l} s_i, \text{ where } l \text{ denotes the number of times } S_i \text{ has grown.}$$

$$t_i = Rank(S_i) - Rank(S_{i-1}).$$

$$Rank(S_l) = \sum_{i=1}^{l} t_i = k - 1. \tag{2}$$

Statement (2) follows from observing that the while loop in Algorithm 1 breaks when $Rank > k - 2$.

We now compute $|S|$ and use Fact 1 to subsequently construct a bound. For this, we count the number of loop iterations in Algorithm 1; we analyze two cases depending on whether line 5 (of the algorithm) executes.

Case 1: only line 4 executes until the loop breaks.

In each iteration, we pick a vector $\vec{c_i}$ such that $\vec{c_i}$ is contained in a hyperedge T_i. Since the size of T_i is at most $r + 1$, we add at most $r + 1$ vectors to S_i, which means $s_i \leq r + 1$.
Notice that $t_i \leq s_i - 1 \leq r$. This is because, first, the increase in rank (i.e., t_i) cannot exceed the increase in the number of vectors, s_i. Furthermore, since the vectors are linearly dependent, the rank of the collection can be at most $s_i - 1$. Thus, $t_i \leq s_i - 1$.

Since $t_i \leq r$,

$$\begin{aligned}
\sum_{i=1}^{l} t_i &\leq \sum_{i=1}^{l} r = lr, \\
k-1 &\leq lr \qquad \text{(from (2))}, \\
l &\geq \left\lceil \frac{k-1}{r} \right\rceil, \\
k-1+l &\geq k-1+\left\lceil \frac{k-1}{r} \right\rceil. \qquad (3)
\end{aligned}$$

And since $s_i - 1 \geq t_i$, we have

$$\begin{aligned}
|S| = \sum_{i=1}^{l} s_i &\geq \sum_{i=1}^{l} (t_i + 1). \\
\Rightarrow |S| &\geq \sum_{i=1}^{l} t_i + \sum_{i=1}^{l} 1. \\
\Rightarrow |S| &\geq k-1+l \qquad \text{(from (2))}. \qquad (4)
\end{aligned}$$

Notice that

$$k-1+\left\lceil \frac{k-1}{r} \right\rceil \geq k + \left\lceil \frac{k}{r} \right\rceil - 2. \qquad (5)$$

Hence, from (3), (4) and (5), $|S| \geq k-1+l \geq k-1+\left\lceil \frac{k-1}{r} \right\rceil \geq k+\left\lceil \frac{k}{r} \right\rceil - 2$, giving us:

$$|S| \geq k + \left\lceil \frac{k}{r} \right\rceil - 2.$$

Case 2: line 5 executes in the last (l-th) iteration of the loop in Algorithm 1.

Recall the following from Case 1:

$$|S| = \sum_{i=1}^{l} s_i \geq \sum_{i=1}^{l} (t_i + 1).$$

For this case,

$$\begin{aligned}
|S| &= \sum_{i=1}^{l-1} s_i + s_l. \\
&\geq \sum_{i=1}^{l-1} (t_i + 1) + s_l. \qquad (6)
\end{aligned}$$

We need to compute s_l. Recall from Case 1 that $|T_i| \leq r+1$. However, in the l-th step (line 5, i.e.), we pick $T' \subset T_l$ and put T' into S. Thus, we put at most r vectors into S in this step; this means that $s_l \leq r$. Also, $t_l \leq s_l$ because the increase in rank should not exceed the increase in the size of S.

The inequality (6) then becomes

$$\begin{aligned} |S| &\geq \sum_{i=1}^{l-1} t_i + \sum_{i=1}^{l-1} 1 + t_l. \\ &\geq \sum_{i=1}^{l} t_i + \sum_{i=1}^{l-1} 1. \\ &= k - 1 + l - 1. \end{aligned} \tag{7}$$

Notice the expression $k-1$ in (7) appears because the final rank is $k-1$ after line 5 executes.

Finally, we compute the total number of while loop iterations (l) in Algorithm 1. We know that the rank can increase by at most r in each step except the last, there are l steps in total and we hit a rank of k in the the l-th iteration (at line 4). So, $k \leq rl$. This means $l \geq \left\lceil \frac{k}{r} \right\rceil$.

Hence, from (7), we have

$$|S| \geq k + \left\lceil \frac{k}{r} \right\rceil - 2.$$

In both Cases 1 and 2 of the analysis, we obtain the same bound for $|S|$. We combine this bound with Fact 1, giving us

$$n \geq k + d + \left\lceil \frac{k}{r} \right\rceil - 2.$$

□

Example 2. We will construct a code that achieves the bound established in Theorem 1 with equality. In other words, we show that a linear $[n,k,d]_q$ code with information locality r has length $n = k + d + \left\lceil \frac{k}{r} \right\rceil - 2$.

We construct pyramid codes (as in [2]) from systematic (n,k,d) Maximum Distance Separable (MDS) codes of distance d in the following way: consider the first parity symbol of the MDS code; call this symbol y_p. Replace this symbol with $\left\lceil \frac{k}{r} \right\rceil$ symbols (y_{p_i}), such that

$$y_p = \sum_{i=1}^{\left\lceil \frac{k}{r} \right\rceil} y_{p_i}.$$

Each y_{p_i} is a parity symbol for r information symbols and no two y_{p_i} have common information symbols.

Then, the pyramid code would contain k information symbols; of the $d-1$ parity symbols of the MDS code, 1 symbol was replaced with $\lceil \frac{k}{r} \rceil$ symbols during the construction above. Thus the length of the constructed code would be

$$\begin{aligned} n &= k + (d-1) - 1 + \left\lceil \frac{k}{r} \right\rceil, \\ n &= k + d + \left\lceil \frac{k}{r} \right\rceil - 2, \end{aligned}$$

as required.

This code has information locality r since every information symbol can be recovered in the following way:

- pick the parity symbol (y_{p_i}) corresponding to the lost information symbol,
- pick the other $r-1$ information symbols that are *contained* in y_{p_i}.

Notice that, the minimum distance of the newly constructed code is at least d (i.e., unchanged). We leave it to the reader to think about this further. ■

Recall that **R** is a matrix whose columns represent repair groups of a code, as defined in section 2.2. We state, without proof, the following lemma.

Lemma 1. The number of columns in **R** satisfies the inequality

$$m \geq \left\lceil \frac{kt}{r} \right\rceil.$$

We end this section by computing a lower bound on the length of codes with another added requirement: availability. The following theorem derives a lower bound on the length of a linear LRC (Locally Repairable Code).

Theorem 2. Let $\mathscr{C}$ be a linear (n,k,r,t)-LRC such that any repair group defined by **R** contains only 1 parity symbol. Then, the length of the code is

$$n \geq k + d - 1 + \left\lceil \frac{kt}{r} \right\rceil - t.$$

Proof. Recall Fact 1:

$$n - d \geq |S|. \tag{8}$$

We will construct a set S that satisfies properties stated in Fact 1; we consider two cases.

Case 1: There is an information symbol which has *exactly* t disjoint repair groups associated with it.

Call this information symbol i. This then means that row i in matrix $\mathbf{R}$ has exactly t ones.
We define the following set: $S = ([k] \setminus \{i\}) \cup P_{R_i}$, where P_{R_i} denotes the set of parity symbols of columns of $\mathbf{R}$ (repair groups) that have zero as their i-th entry.

We now compute $|S|$. Notice that $|[k] \setminus \{i\}| = k - 1$; this should be obvious. Further, notice $|P_{R_i}| = m - t$. This is because there are t ones in row i (seen earlier) and $m - t$ zeroes in the same row. Each one of those $m - t$ columns (or repair groups) have 1 parity symbol, as mandated by the theorem.

Observe that $([k] \setminus \{i\})$ and P_{R_i} are disjoint because the first set refers to information symbols and the second set refers to parity symbols. Thus, $|S| = |[k] \setminus \{i\}| + |P_{R_i}| = k - 1 + m - t$. From the discussion so far, it can also be seen that we cannot recover the i-th information symbol from the symbols indexed by S. Thus, from (8), we have the following:

$$\begin{aligned} n - d &\geq |S|, \\ n - d &\geq k - 1 + m - t, \\ n &\geq k + d - 1 + m - t. \end{aligned}$$

Since $m \geq \lceil \frac{kt}{r} \rceil$ from Lemma 1,

$$n \geq k + d - 1 + \left\lceil \frac{kt}{r} \right\rceil - t. \tag{9}$$

Case 2: There is an information symbol which has more than t disjoint repair groups associated with it.

We pick an information symbol with the smallest number of repair groups larger than t. Call this information symbol y_j; assume that the jth row in $\mathbf{R}$ (corresponding to the information symbol) has Hamming weight $t' > t$. Then, $kt' \leq \{\text{number of 1s in } \mathbf{R}\} \leq mr$, giving us

$$m \geq \left\lceil \frac{kt'}{r} \right\rceil. \tag{10}$$

We can now plug in our reasoning from Case 1 to construct $S = ([k] \setminus \{j\}) \cup P_{R_j}$ such that the j-th symbol cannot be recovered from the encoded symbols indexed by S.

Thus we have,

$$\begin{aligned} |S| &\leq n-d, \\ k-1+m-t' &\leq n-d, \\ n &\geq k+d-1+m-t', \\ n &\geq k+d-1+\left\lceil \frac{kt'}{r} \right\rceil - t'. \end{aligned} \tag{11}$$

We used the result from (10) to obtain (11).

Since $t' > t$, for $r \leq k$, we have $k+d-1+\left\lceil \frac{kt'}{r} \right\rceil + t' > k+d-1+\left\lceil \frac{kt}{r} \right\rceil + t$. Combining this with (9) and (11), we finally obtain

$$n \geq k+d-1+\left\lceil \frac{kt}{r} \right\rceil - t.$$

□

4 Comments

Code locality is of interest in distributed storage systems: we can think of the i-th storage server in such a system as the i-th symbol of a codeword. In an event where the i-th server fails, we can use r other servers to reconstruct the data lost from failure. Since data transmission incurs costs, it is of interest to find a lower bound on the number of servers needed to repair the information in the failed server.

References

[1] Parikshit Gopalan, Cheng Huang, Huseyin Simitci, and Sergey Yekhanin. On the locality of codeword symbols. *Information Theory, IEEE Transactions on*, 58(11):6925–6934, Nov 2012.

[2] Cheng Huang, Minghua Chen, and Jin Li. Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems. In *Network Computing and Applications, 2007. NCA 2007. Sixth IEEE International Symposium on*, pages 79–86, July 2007.

[3] Ankit Singh Rawat, Dimitris S. Papailiopoulos, Alexandros G. Dimakis, and Sriram Vishwanath. Locality and availability in distributed storage. *CoRR*, abs/1402.2011, 2014.