CS 229r Essential Coding Theory March 4, 2020

Lecture 11

Instructor: Madhu Sudan *Scribe: Andrew Lu*

1 Today

- Decoding Concatenated Codes
- List-Decoding to Capacity

2 Review of Code Concatenation

Suppose we are given an $[N, K, D]_Q$ code C_{out} and an $[n, k, d]_q$ code C_{in} with $Q = q^k$. Recall from Lecture 7 that we can construct the *concatenated code* $C_{\text{out}} \circ C_{\text{in}}$, which constitutes an $[Nn, Kk, Dd]_q$ code. In this lecture, we will focus on the decoding process for concatenated codes.

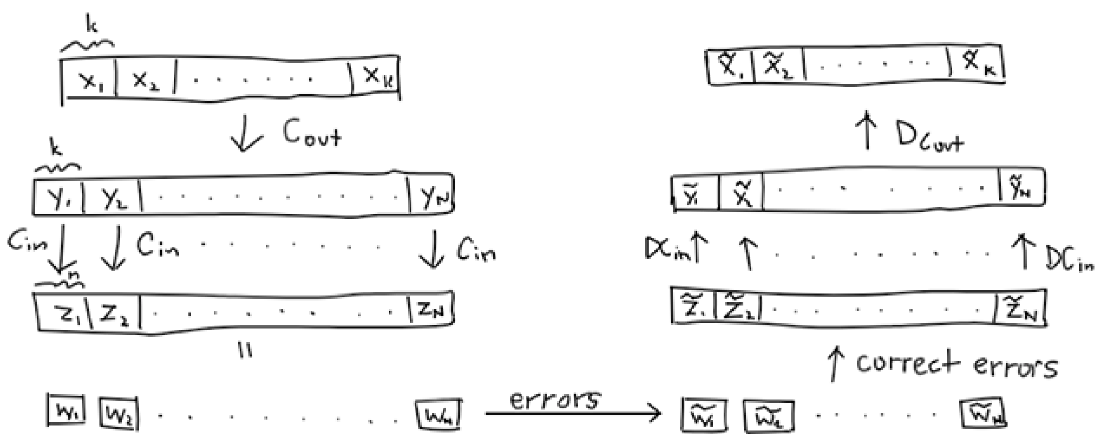

Figure 1: Encoding (left) and decoding (right) for concatenated codes.

Note that for concatenated codes, the natural idea for a decoding algorithm is to use the individual decoders $D_{C_{\text{in}}}$ and $D_{C_{\text{out}}}$. If the $D_{C_{\text{out}}}$ runs in time T_{out} and $D_{C_{\text{in}}}$ runs in time T_{in}, then such an algorithm would run in $O(T_{\text{out}} + N \cdot T_{\text{in}})$ time, where T_{in} is polynomially bounded in terms of Q.

Question: How many errors does this decoding algorithm correct?

Note that the decoding algorithm would fail to retrieve the original message only when at least $D/2$ errors are present among the $\tilde{y}_i$. For this to have happened, for each erroneous $\tilde{y}_i$ there must have been at least $d/2$ errors present in the $\tilde{w}_i$ (causing $\tilde{w}_i$ to become mistakenly decoded to $\tilde{z}_i$ and then $\tilde{y}_i$), for a total of at least $(D/2)(d/2) = \frac{Dd}{4}$ errors. Therefore, the decoding algorithm can correct $< \frac{Dd}{4}$ errors.

Using the naive decoder, we are a factor of 2 away from being able to correct up to the theoretically optimal bound of $\frac{Dd}{2}$ errors.

3 Generalized Minimum Distance Decoding

Recall from Problem Set 1 that a decoder for a distance-d code C can decode a codeword with s erasures and e errors if $s + 2e < d$.

Remark Reed-Solomon codes satisfy this property due to the fact that an $[n, k, d]$ Reed-Solomon code with s erasures can be viewed as an $[n-s, k, d-s]$ Reed-Solomon code by the puncturing principle, in which we have dropped s points to evaluate. Since $\frac{d-s}{2} > e$, we can still correct up to e errors.

By incorporating erasures, we can get rid of parts of the received word that deviate in the Hamming distance from the decoded word by a large margin. Note that this information was not used by the naive decoder.

We aim to artificially construct a codeword $\hat{y}_1\hat{y}_2\cdots\hat{y}_N$ to replace the original codeword $\tilde{y}_1\tilde{y}_2\cdots\tilde{y}_N$ obtained from the inner decoder, with the new codeword containing erasures in locations where a large Hamming distance is present (and therefore a large likelihood of error).

GMD Algorithm: Define $\tilde{e}_i = \Delta(\tilde{z}_i, \tilde{w}_i)$ for each $1 \le i \le N$. If $\tilde{e}_i > \frac{d}{2}$, set $\hat{y}_i = ?$. Otherwise, for $0 \le \tilde{e}_i \le \frac{d}{2}$, set $\hat{y}_i = ?$ with probability $\frac{\tilde{e}_i}{d/2}$ and $\hat{y}_i = \tilde{y}_i$ with probability $1 - \frac{\tilde{e}_i}{d/2}$. Return $D_{C_{\text{out}}}(\hat{y}_1\hat{y}_2\cdots\hat{y}_N)$.

We make a couple of sanity checks in the algorithm above. If $\tilde{e}_i = 0$, either 0 or more than d errors occurred (i.e. we corrected the intended message to a different message of distance at least d away). The second case isn't likely, so we should be reasonably sure no errors occurred. If $\tilde{e}_i > \frac{d}{2}$, then more than $\frac{d}{2}$ errors occurred, so we should erase the symbol. For the in-between cases, we should erase the symbol on a sliding scale of certainty.

Claim 1. *Suppose $\Delta(w_1w_2\cdots w_n, \tilde{w}_1\tilde{w}_2\cdots\tilde{w}_N) < \frac{Dd}{2}$. From the above algorithm, suppose $\hat{y}_1\hat{y}_2\cdots\hat{y}_N$ contains s erasures and e errors compared with the correct encoding $y_1y_2\cdots y_n$. Then $\mathbb{E}[2e+s] < D$. In other words, we will obtain the correct original message in expectation.*

Proof. Fix a position i, and let $e_i = \Delta(w_i, \tilde{w}_i)$. We consider two cases:

Case 1: $e_i < d/2$. Then $\tilde{w}_i$ must have been corrected to the right $\tilde{z}_i$, so $\tilde{z}_i = w_i$ and $\tilde{e}_i = e_i$. An error could not have occurred, but the symbol could have been erased with probability $\frac{e_i}{d/2}$.

Case 2: $e_i \ge d/2$. If $\tilde{z}_i = w_i$, then $e_i = \tilde{e}_i$ and $\tilde{e}_i + e_i \ge d$. Otherwise, by Triangle Inequality,

$$\tilde{e}_i + e_i = \Delta(\tilde{z}_i, \tilde{w}_i) + \Delta(w_i, \tilde{w}_i) \ge \Delta(\tilde{z}_i, w_i) \ge d,$$

so $\tilde{e}_i + e_i \ge d$ holds regardless. Then the probability of erasure is $\frac{\tilde{e}_i}{d/2}$, and the probability of error is $1 - \frac{\tilde{e}_i}{d/2}$, so the amount contributed to the expectation is

$$\frac{\tilde{e}_i}{d/2} + 2\left(1 - \frac{\tilde{e}_i}{d/2}\right) = 2 - \frac{\tilde{e}_i}{d/2} \le \frac{e_i}{d/2}.$$

By linearity of expectation, we have that

$$\mathbb{E}[2e+s] \le \sum_{i=1}^{N} \frac{e_i}{d/2} < \frac{Dd/2}{d/2} = D,$$

as desired. □

Remark An alternative but essentially similar algorithm is based on choosing a single threshold $\tau \in [0, d/2]$ at random to compare every $\tilde{e}_i$ against to determine whether $\tilde{y}_i$ should be erased, i.e. we set $\hat{y}_i = ?$ whenever $\tilde{e}_i > \tau$, and $\hat{y}_i = \tilde{y}_i$ otherwise. Both algorithms run in time $O(dT_{\text{out}} + NT_{\text{in}})$.

Exercise 2. *Show that with this modified version of the GMD algorithm, it is still the case that $\mathbb{E}[2e+s] < D$.*

4 List Decoding to Capacity

Definition 3. *For $0 \leq \rho \leq 1$ and $L \geq 1$, a code $C \subset \Sigma^n$ is (rho, L)-list decodable if for all $w \in \Sigma^n$,*

$$|Ball(w, pn) \cap C| \leq L.$$

Exercise 4. *Show that $\rho \leq 1 - R$. Moreover, a random code of rate R satisfies $\rho \approx 1 - R$.*

Exercise 5. *(Hard) Show that $\rho \geq 1 - \sqrt{1-\delta}$. Moreover, this bound is tight.*

Question: Given a code C, what is the relationship between its rate R and ρ if C is $(\rho, \text{poly}(n))$-list decodable?

To answer the above question, we now introduce *folded Reed-Solomon codes.* Recall that Reed-Solomon codes are obtained by mapping polynomials encoded by elements of $\mathbb{F}_q^k$ to evaluation points encoded by elements of $\mathbb{F}_q^n$. For $s \geq 1$, we simply group the n evaluation points into n/s clusters of size s. This changes the definition of what constitutes an error.

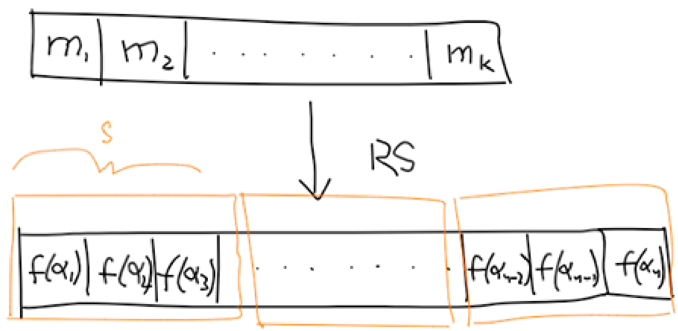

Figure 2: Folded Reed-Solomon code, with $f = \sum_{i=1}^{k} m_i X^{i-1}$.