CS 229r Essential Coding Theory March 2, 2020

Lecture 10

Instructor: Madhu Sudan *Scribe: David Xiang*

1 Today

- Reed-Solomon Decoding + List Decoding
- Abstracting Reed-Solomon
- Decoding Concatenated Codes

2 Reed-Solomon Decoding

We recall from last lecture the Reed-Solomon Decoding problem:
Input: code parameters $n, k, \mathbb{F}_q$ and points $\{(\alpha_i, \beta_i)\}_{i=1}^n$, $\alpha_i, \beta_i \in \mathbb{F}_q$.
Output: Find a polynomial $M(x)$ of degree $< k$ such that the number of errors

$$|\{i : M(\alpha_i) \neq \beta_i\}| \leq t$$

for some parameter t. In our initial analysis we will take $t = \frac{n-k}{2}$.

We also discussed the Berlekamp-Welch algorithm for solving this problem. The algorithm proceeds in two parts:

(1) Find a pair of polynomials (E, W) with

- $E \neq 0$
- $\deg E \leq t$
- $\deg W < k + t$

such that $E(\alpha_i)\beta_i = W(\alpha_i)$ for all i.

(2) Output $\frac{W}{E}$.

We claim that if the β_i arise from an instance of Reed-Solomon encoding with fewer than t errors (i.e. there is some polynomial M with degree less than k such that $M(\alpha_i) = \beta_i$ for at least $n - t$ of the α_i) then there will exist a solution to (1). This will follow from explicit construction: defining E to be the *error locator polynomial* $E(x) = \prod_{i:M(\alpha_i \neq \beta_i)}(x - \alpha_i)$ we will have that

$$E(\alpha_i)\beta_i = E(\alpha_i)M(\alpha_i)$$

since if $M(\alpha_i) \neq \beta_i$ both sides will be zero. This implies the pair (E, ME) is a pair of polynomials satisfying the conditions in (1).

We next check that any two pairs (E, W), (F, U) satisfying the conditions of (1) give the same answer, i.e $\frac{W}{E} = \frac{U}{F}$. This follows from noting that for all i, we have

$$U(\alpha_i) = F(\alpha_i)\beta_i \Rightarrow U(\alpha_i)E(\alpha_i)\beta_i = F(\alpha_i)\beta_i W(\alpha_i)$$

Either β_i is zero, in which case the condition $U(\alpha_i) = F(\alpha_i)\beta_i$ (respectively W, E) forces $U(\alpha_i) = 0$ (resp. $W(\alpha_i)$ so that both sides are zero, or β_i is nonzero and we may simply divide both sides by β_i. Either way, we have that

$$U(\alpha_i)E(\alpha_i) = F(\alpha_i)W(\alpha_i)$$

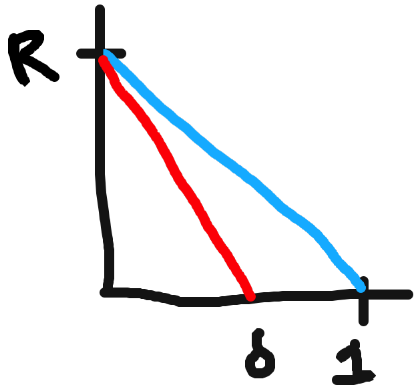
for all i. On the other hand, $\deg UE$ and $\deg FW$ are less than $k + 2t$, under the assumption that $t \leq \frac{n-k}{2}$ this degree is strictly less than n. It follows that UE and FW are the same polynomial, as desired.

Combining the two previous observations shows the correctness of this algorithm.

Finally, we note that step (1) can be carried out in polynomial time by viewing the condition $E(\alpha_i)\beta_i = W(\alpha_i)$ as a linear system with variables the coefficients of E, W. The division $\frac{W}{E}$ can also be implemented in polynomial time, so we have a genuinely polynomial time decoding algorithm for the Reed-Solomon code.

3 Reed-Solomon List Decoding

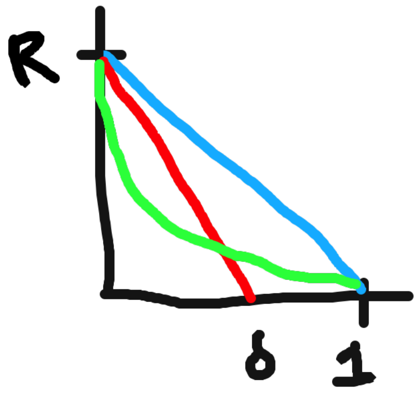
Where will we go with the above idea today? Let's take $q \to \infty$, so that by the Reed-Solomon construction we know we can achieve everything on the line $R + \delta = 1$ (which we know is optimal, pictured below in blue). We have explicit decoding algorithms when the error is half the distance (represented by the red line), in the next few lectures we will try to push the red line towards the blue line.



Of course, this problem is somewhat ill-posed, as moving anywhere past the dotted line requires algorithms capable of correcting more than 50% errors. We already know in this case there is no way our answers can be well-defined: we do not know which of two possible outputs (or possibly more than two outputs) is the correct answer. To remedy this problem, we ask that we output all possible codewords (i.e. "list-decoding") and it is under this paradigm that we will reach the exact line desired.

Definition 1. *We say a code C is (ρ, L) list-decodable if for all words w in Σ^n, $|Ball(w, \rho n) \cap C| \leq L$, where $Ball(w, \rho n)$ refers to the ball of radius ρn about w.*

Exercise 2. *(Extension of the Elias-Bassalygo bound) Show that Reed-Solomon codes are $(1 - \sqrt{R}, n)$-list decodable.*



The above RS list decoding bound corresponds to the green line (we will see how to explicitly decode near this bound by the end of the lecture).

Now, we have the corresponding algorithmic problem of Reed-Solomon List Decoding:

Input: code parameters $n, k, \mathbb{F}_q$ and points $\{(\alpha_i, \beta_i)\}_{i=1}^n$, $\alpha_i, \beta_i \in \mathbb{F}_q$.

Output: Find **all** polynomials $M(x)$ of degree $< k$ such that the number of errors

$$|\{i : M(\alpha_i) \neq \beta_i\}| \leq t$$

for some parameter t.

In regular Reed-Solomon decoding, we considered the equation $E(\alpha_i)\beta_i = W(\alpha_i)$, which corresponds to looking at zeros of the curve $E(x)y - W(x)$. The main idea to list-decode the RS code is to change this curve into something of higher degree. More explicitly, our first attempt at an algorithm will be given by

(1) Find $Q(x, y)$ with $\deg_x Q, \deg_y Q \leq \sqrt{n}$ such that Q is not identically zero and $Q(\alpha_i, \beta_i) = 0$ for all i.

(2) Factor $Q(x, y)$ and if there exists a factor of form $y - M(x)$, report M. (Implicitly, this will only every output M with degree at most $\sqrt{n}$, i.e. we are list-decoding at a rate of $\frac{1}{\sqrt{n}}$. We will fix this in the next iteration of this algorithm)

Runtime: Part (1) can again be viewed as solving a system of linear equations, with the unknowns this time being the coefficients of the $x^i y^j$ terms in Q (there are at most $O(n)$ of them) and the equations given by $Q(\alpha_i, \beta_i) = 0$, this can be solved in polynomial time.

As for part (2), we claim factorization of bivariate polynomials can be done in polynomial time. We won't prove this today (see http://people.csail.mit.edu/madhu/ST12/scribe/lect11.pdf for a reference).

Correctness: We first show that (1) there exist solutions Q, and then that (2) $y - M(x)$ divides Q if M corresponds to a valid codeword.

(1) Again, this follows from linear algebra. We have a linear system with n equations but $(\sqrt{n} + 1)^2 > n$ unknowns, dimension considerations will tell us that a solution will exist.

(2) As in regular RS decoding, we will need an additional assumption on the number of errors. Let us assume for now that the quantity $n - t$ (the number of agreements of M with the (α_i, β_i)) is larger than $2k\sqrt{n}$.

Proof. Let $S = \{(\alpha_i, \beta_i) : M(\alpha_i) = \beta_i)\}$. S is a subset of the zeros of polynomial $y - M(x) \triangleq P(x, y)$, it's also a subset of the zeros of $Q(x, y)$. By assumption $|S| \geq 2k\sqrt{n}$.

By Bezout's theorem, if P and Q do not have a common factor, then since $\deg P < k$ and $\deg Q \leq 2\sqrt{n}$ they can have no more than $2k\sqrt{n}$ common zeros, but this contradicts the assumption that we have a large number of agreements.

We conclude that P and Q must then share a common factor, but since P is irreducible we conclude $P|Q$ as desired. □

Since the y degree of Q is at most $\sqrt{n}$ we will not output more than $\sqrt{n}$ factors (in fact, by considering the x-degree we see we will not output more than $\sqrt{n}/k$ list elements). Putting this all together, we've shown that

Theorem 3. *The Reed-Solomon Code is* $(1 - 2\frac{k}{\sqrt{n}}, \sqrt{n})$ *list-decodeable.*

What have we shown? We've shown that we can list-decode from a very large fraction of errors, but in order to get this we need k to be sublinear in n (in particular, we needed to assume that the number of agreements is at least $2k\sqrt{n}$, so k must be $O(\sqrt{n})$).

The fact that there exists asymmetry in the x-degree and y-degree (as we observed earlier, by considering the x-degree we actually show that the lists are not larger than $\sqrt{n}/k$) gives us room to exploit. More concretely, we should think of $Q(x, y)$ as a polynomial of x, into which we substitute $y = M(x)$ (from this perspective, a factor $y - M(x)$ corresponds to a root of $Q(x, y)$ viewed as a polynomial over $\mathbb{F}_q[x][y]$), and so in reality the y degrees are much much impactful than the x degrees.

So, let's repeat the above analysis, except now take $\deg_x Q \leq \sqrt{kn}$, and $\deg_y Q \leq \sqrt{\frac{n}{k}}$ (ignore integrality issues). We show the same claims (1) there exist a solution Q and (2) $y - M(x)$ divides Q under assumptions on t.

(1) The analysis here works out to be basically the same (again, we just compare number of unknowns to number of equations).

(2) We take $n - t \geq 2\sqrt{kn}$ now. Again, define $S = \{(\alpha_i, \beta_i) : M(\alpha_i) = \beta_i)\}$, so that $|S| \geq 2\sqrt{kn}$. Define $g(x) = Q(x, M(x))$, we observe that $\deg g = 2\sqrt{kn}$ since $\deg_x M < k, \deg_y Q < \sqrt{n/k}$.

On the other hand, $g(\alpha_i) = 0$ if $(\alpha_i, \beta_i) \in S$, this is by definition. By assumption on the size of S then, it follows that $g(X)$ is identically zero, so $Q(x, M(x))$ is identically zero. This is just the statement that $y - M(x)$ divides $Q(x, y)$. (again, view $Q(x, y)$ as an element of $\mathbb{F}_q[x][y]$)

Interpreting this in the language of list-decoding shows that

Theorem 4. *The Reed-Solomon Code is* $(1 - 2\sqrt{R}, \sqrt{1/R})$*-list decodeable.*

More careful fine-tuning lets one move $n - t \geq \sqrt{2kn}$. One can get rid of the factor of 2 altogether by complicated algebraic geometry, (so we have explicit algorithms achieving the bound of exercise 2) we will take a different route (in a later class).

4 Abstracting Reed-Solomon

If one carefully follows the analysis for Reed-Solomon decoding, one observes that essentially the main property we used is the fact that Reed-Solomon codes have large distance. The only algebraic fact we really exploited is that the product of a degree a and degree b polynomial has degree at most $a + b$. This lets us generalize our Reed-Solomon decoding algorithm to arbitrary codes by generalizing the notion of

This gives us an abstract decoding algorithm (to half the distance) for general codes.

Let $\mathcal{E}, \mathcal{W}$ be error-correcting codes (an "error locating pair" for the code C) such that

1. $\dim \mathcal{E} \geq t + 1$

2. The distance of $\mathcal{W}$ is $\geq t$

3. $\mathcal{E} * \mathcal{C} \subseteq \mathcal{W}$

What is the $*$ operation?

Definition 5. *Given $a, b \in \mathbb{F}_q^n$, we define $(a * b) \in \mathbb{F}_q^n$ to be the vector whose jth coordinate equals $a_j b_j$ (i.e. $a * b$ is the coordinatewise product of a and b).*

The motivation here is that if a and b are the evaluations of polynomials f and g on n points in $\mathbb{F}_q$, then $a * b$ is the evaluation of fg on the same n points in $\mathbb{F}_q$.

Definition 6. *Given two codes $\mathcal{E}, \mathcal{C}$, we define $\mathcal{E} * \mathcal{C}$ to be the set $\{e * c, e \in \mathcal{E}, c \in \mathcal{C}\}$*

Note that $\mathcal{E} * \mathcal{C}$ is not generally a vector space.

This lets us define a generalization of our previous Reed-Solomon algorithm. Given r in $\mathbb{F}_q^n$, we decode it to a codeword in C by

(1) Find $a, b \in \mathcal{E}, \mathcal{W}$ with $b = a * r$, a not identically zero.

(2) Let $b'_i = \frac{b_i}{a_i}$ if $a_i \neq 0$, and $b'_i = ?$ if $a_i = 0$. Finally, erasure decode b' for C.

Exercise 7. *Fill in the details/prove correctness of the above scheme.*