CSE522/Math581EB Coding Theory **Fall 2001**

Lecture 8: November 1

Lecturer: Chris Umans *Scribe: Kaustub D.*

8.1 List Decoding

In a previous lecture, we saw that unique decoding of Reed-Solomon codes was possible when we receive slightly more than 50% of the positions correctly (for an appropriate choice of n and k). What if we receive only 1% correctly? We want to return a short list of polynomials with one of them guaranteed to be the correct polynomial.

Recall from Lecture 7:

Definition 8.1 *A (t, l) error-correcting code, is a code which has at most l words in a ball of radius t.*

Johnson Bound: A $[n, k, \delta n]_2$ code is $(\tau n, n)$ error correcting, where $\tau \leq \frac{1}{2}(1 - \sqrt{1 - 2\delta})$

(t, l) **List Decoding Problem:**
Let C be a (t, l) error-correcting code. Given $r \in \mathbf{F}_q^n$, output all the codewords $\{c \mid c \in C \text{ and } \Delta(r, c) \leq t\}$. The size of this set is $\leq l$.

8.2 List Decoding for Reed-Solomon Codes

Let $RS[n, k, n - k + 1]_n$ be a Reed-Solomon code. What type of list-decoding can we expect? Our Johnson bound was earlier proved for $q = 2$; the generalized Johnson bound is given as follows:

Theorem 8.2 ***q*-ary Johnson Bound** *A $[n, k, \delta n]_q$ code is $(\tau n, n)$ error correcting, provided $\tau \leq \frac{q-1}{q}\left(1 - \sqrt{1 - \frac{q}{q-1}\delta}\right)$*

In the case of RS codes, let $\delta n = n - k + 1$. Hence from the Johnson bound RS is a $(\tau n, n)$ error-correcting code where,

$$\tau \leq \frac{n-1}{n}\left(1 - \sqrt{1 - \frac{n}{n-1}\delta}\right)$$

Let n large, we have:

$$\begin{aligned} \tau &\leq 1 - \sqrt{1 - \delta} \\ \delta &= 1 - \frac{k}{n} \end{aligned}$$

Hence,

$$\tau \leq 1 - \sqrt{\frac{k}{n}}$$

Multiplying by n we get,

$$\tau n \leq n - \sqrt{kn}$$

Recall that for unique decoding we needed,

$$\begin{aligned} e &\leq \frac{n-k}{2} \\ \tau &\leq \frac{1}{2} - \frac{R}{2}. \end{aligned}$$

For list decoding we need,

$$\tau \leq 1 - \sqrt{R}.$$

As $R \to 0$, for unique decoding $\tau \to \frac{1}{2}$, but for list decoding $\tau \to 1$. Hence for list decoding we can have almost 100% error and still recover back (a short list containing) the original polynomial.

Reed Solomon List Decoding Algorithm

Given: n pairs $(x_i, y_i), k, e$
Output: A polynomial $p(x)$ such that $\deg(p) \leq k-1$ and $p(x_i) \neq y_i$ for at most e values of i.

Here (x_i) are the n elements in the field on which the polynomial is evaluated, (y_i) is the codeword which is received. e is the maximum number of errors. The polynomial $p(x)$ matches the encoded polynomial in all except e places.

If we summarize our algorithm for unique decoding,

- $E(x)$ is an error locator polynomial, where $\deg(E(x)) \leq e$, (think $E(x_i) = 0$ if $p(x_i) \neq y_i$).
- $N(x)$ is a polynomial such that $N(x_i) = y_i E(x_i) \forall i.$ and degree of $N(x) \leq e + k - 1$. (we can think that, $N(x) = p(x)E(x)$)
- we output $\frac{N(x)}{E(x)}$ (which we showed was equal to $p(x)$ provided $n \geq 2e + k$)

We generalize this algorithm as follows [GS99]:

- Let $Q(x, y)$ be a polynomial with degree $\leq D$, such that $Q(x_i, y_i) = 0$. (the intuition being that, $Q(x, y) = E(x)y - N(x)$, which is the same as saying $Q(x, y) = E(x)y - p(x)E(x)$. $D = e + k - 1$)
- If $p(x_i) = y_i$, then $Q(x_i, p(x_i)) = 0$.
- If $p(x_i) = y_i$, for more than $n - e$ values of i, then $Q(x, p(x))$ has more than $n - e$ zeros. If $n - e > D$, then it means $Q(x, p(x)) \equiv 0$. i.e. $y - p(x)$ divides $Q(x, y)$. (Continuing the intuition above, we would need $n - e > e + k - 1$; i.e. $n > 2e + k - 1$, which is exactly what we had for unique decoding.)
- We relax the fact that the degree of $y \leq 1$.
- We "weight" the zeros of Q by a factor r so that we can obtain a contradiction whenever $(n-e)r > D$ (in contrast to $n - e > D$).

Theorem 8.3 *Given n pairs (x_i, y_i), we can find all $p(x)$ with degree $\leq k-1$, such that $p(x_i) \neq y_i$ for at most e values of i, where provided $e < n - \sqrt{kn}$.*

Step 1

Find Q with certain properties:

- One way to say that $Q(x_i, y_i) = 0$ is that $Q(x - x_i, y - y_i)$ has zero coefficient on the constant term. The r-weighted polynomial $Q(x, y)$ requires that $Q^{(i)} = Q(x + x_i, y + y_i)$ has zero coefficients on all terms of degree less than r, for all i.

  If $q^{(i)}_{j_1 j_2}$ is the coefficient of $x^{j_1} y^{j_2}$ in $Q^{(i)}$, then

  $$q^{(i)}_{j_1 j_2} = \sum_{j_1' \geq j_1} \sum_{j_2' \geq j_2} \binom{j_1'}{j_1} \binom{j_2'}{j_2} x_i^{j_1' - j_1} y_i^{j_2' - j_2} q_{j_1' j_2'}$$

  and we require that $q^{(i)}_{j_1 j_2} = 0$ for all $j_1 + j_2 < r$, and all i.

- $\deg_x + k\deg_y \leq l$
- $Q \not\equiv 0$.

Lemma 8.4 *If*

$$n\binom{r+1}{2} < \frac{l(l+2)}{2k} \tag{8.1}$$

then Q exists and can be found by solving a homogeneous linear system.

Proof: Condition (8.1) merely states that the number of linear constraints of the form, $q^{(i)}_{j_1 j_2} = 0$ should be less than the number of variables.

$$\begin{aligned} \#\text{constraints} &= n\binom{r+1}{2} \\ \#\text{variables} &= \sum_{j_2=0}^{\lfloor \frac{l}{k} \rfloor} (l + 1 - kj_2) \geq \frac{l}{k}\frac{l+2}{2} \end{aligned}$$

Hence we can solve the linear system for the coefficients of Q. ■

Step 2

Find all $p(x)$ with degree less than k, such that $y - p(x)$ divides $Q(x)$. (we assume that we know how to factorize a polynomial).

Lemma 8.5 *If $p(x)$ has degree less than k and $p(x_i) = y_i$ for more than $n - e$ values of i and $r(n - e) > l$ then $(y - p(x)) \mid Q(x, y)$.*

Lemma 8.6 *If $p(x)$ is a polynomial such that $p(x_i) = y_i$, then $(x - x_i)^r$ divides $Q(x, p(x))$.*

Proof: (Lemma 8.6)
$Q^{(i)}(x, y) = Q(x + x_i, y + y_i)$. z^r divides $Q^{(i)}(z, z)$ (because all coefficients with degree less than r are 0). Similarly z^r divides $Q(z, zp'(z))$ for any polynomial p'. Now define $p''(z) = p(z + x_i) - y_i$. $p''(0) = 0$, hence

$p''(z)$ can be written as $zp'(z)$. Hence z^r divides $Q^{(i)}(z, p(z+x_i)-y_i) = Q(z+x_i, p(z+x_i))$. Let $z = x - x_i$. Hence we get $(x-x_i)^r$ divides $Q(x, p(x))$. ■

Proof: (Lemma 8.5)
$Q(x, p(x))$ has degree $\leq l$. $p(x_i) = y_i$ for at least $n-e$ values of i. Let these values be the set $S \subseteq [n]$. Hence from Lemma 8.6 we get

$$\prod_{i \in S} (x - x_i)^r \mid Q(x, p(x))$$

where the x_i are all distinct members of the field. So we have a polynomial of degree $r(n-e)$ which divides a polynomial of degree l. But $r(n-e) > l$. Hence $Q(x, p(x)) \equiv 0$. And thus $y - p(x)$ divides $Q(x, y)$. ■

Now we need to define the parameters we have been using so far, that is r and l. Given n, k and e errors, we need to pick r, l such that:

- $n\binom{r+1}{2} < \frac{l(l+2)}{2k}$
- $l < r(n-e)$

Let $t = n - e$. Set $l = rt - 1$. Hence we have

$$\begin{aligned} n\binom{r+1}{2} &< \frac{(rt-1)(rt+1)}{2k} \\ n(r+1)r &< \frac{(rt)^2 - 1}{k} \\ knr^2 + knr &< r^2t^2 - 1 \\ r^2(t^2 - kn) - knr - 1 &> 0 \end{aligned}$$

For this equation to have a solution $t^2 > kn$. i.e. $n - e = t > \sqrt{kn}$. Setting r to be one more than the solution of the quadratic suffices; i.e.,

$$r = \left\lfloor \frac{kn + \sqrt{(kn)^2 + 4(t^2 - kn)}}{2(t^2 - kn)} \right\rfloor + 1$$

List decoding concatenated codes

1. Brute force list decode all the inner code words.
2. Use Reed-Solomon list decoding for the outer codes.

Better decoding is possible by determining weights in Step 1, and using these in a weighted Reed-Solomon list-decoding algorithm (an easy generalization of the algorithm we just saw) in Step 2. Details are in [GS00].

References

[1] V. Guruswami and M. Sudan (1999) Improved Decoding of Reed-Solomon and Algebraic-Geometric Codes. *IEEE Trans. Inform. Theory* **45**, 1757–1767.

[2] V. Guruswami and M. Sudan (2000) List-Decoding Algorithms for Certain Concatenated Codes. *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC).*