Information and Coding Theory **Autumn 2014**

Lecture 13: November 11, 2014

Lecturer: Madhur Tulsiani & Shi Li Scribe: Hing Yin Tsang

Let $q \geq n$, the Reed-Solomon code over $\mathbb{F}_q$ is a linear code $C : \mathbb{F}_q^k \to \mathbb{F}_q^n$ defined as follows: fix $\{a_1, \ldots, a_n\} \subseteq \mathbb{F}_q$, to encode a message $(m_0, \ldots, m_{k-1})$, identify it with $P(x) = m_0 + m_1 x + \cdots + m_{k-1} x^{k-1}$ and C maps it to the codeword $(P(a_1), P(a_2), \ldots, P(a_n))$. In the last lecture we showed that C has distance $d = n - k + 1$, which is optimal under Singleton bound. We will see the unique decoding and list-decoding algorithms for Reed-Solomon codes up to $\lfloor \frac{n-k}{2} \rfloor$ (which is $\lfloor \frac{d-1}{2} \rfloor$) and $n - 2\sqrt{kn}$ errors respectively.

1 Unique decoding Reed-Solomon

We show that the following algorithm by Berlekemp and Welch [2] decodes Reed-Solomon codes up to t errors when $t \leq \lfloor \frac{n-k}{2} \rfloor$. Note that the message $(m_0, \ldots, m_{k-1})$ is identified with the polynomial $P(x) = \sum_{i=0}^{k-1} m_i x^i$. Let $(P(a_1), \ldots, P(a_n))$ be the original codeword and let $y = (y_1, \ldots, y_n)$ be it's corrupted version (we only know y). The distance between them is at most t.

Unique decoding for Reed-Solomon codes
Input: $\{(a_i, y_i)\}_{i=1,\ldots,n}$

1. Find $E, Q \in \mathbb{F}_q[x]$ such that $E \not\equiv 0$, $\deg(E) \leq t$, $\deg(Q) \leq k - 1 + t$

$$\forall i \in [n] \quad Q(a_i) = y_i \cdot E(a_i)\,.$$

2. Output $\frac{Q}{E}$.

We first observe that Step 1 in the algorithm can be done by solving linear system. Let $E(x) = e_0 + e_1 x + \cdots + e_t x^t$ and $Q(x) = q_0 + q_1 x + \cdots + q_{k-1+t} x^{k-1+t}$. Then for each given (a_i, y_i), the equation $Q(a_i) = y_i E(a_i)$ is linear in variables $e_0, \ldots, e_t$ and $q_0, \ldots, q_{k-1+t}$. Note that such system is homogeneous and hence it always has a trivial solution. We need to show that there is a solution with nonzero E.

Lemma 1.1 *There exists (E, Q) that satisfies the conditions in Step 1 of the algorithm.*

Proof: Let $I = \{i \in [n] \mid P(a_i) \neq y_i\}$ and $E^* = \prod_{i \in I}(x - a_i)$ (in particular, $E^* \equiv 1$ if I is empty). Let $Q^* = P \cdot E^*$. Then for all $i \in [n]$, we have $y_i E^*(a_i) = P(a_i) \cdot E^*(a_i) = Q^*(a_i)$. ■

If there is a unique nonzero solution to the linear system in Step 1, then Step 2 outputs the correct polynomial. But in general there can be more than one such solution. The following lemma guarantees the correctness of Step 2.

Lemma 1.2 *For any two solutions (Q_1, E_1) and (Q_2, E_2) that satisfy the conditions in Step 1,*

$$\frac{Q_1}{E_1} = \frac{Q_2}{E_2}.$$

Proof: It suffices to show $Q_1E_2 = Q_2E_1$. Indeed, since they satisfy the equation $Q(a_i) = y_iE(a_i)$ for each $i \in [n]$, we have

$$(Q_1E_2)(a_i) = y_iE_1(a_i)E_2(a_i) = (E_1Q_2)(a_i)$$

for each $i \in [n]$. It implies that they must be the same as there are n zeros of $Q_1E_2 - Q_2E_1$ but its degree is at most $k + 2t - 1 \leq k + 2\lfloor \frac{n-k}{2} \rfloor - 1 \leq n - 1$. ∎

2 List-decoding

The decoding algorithm in the previous section requires the number of errors to be at most $\lfloor \frac{n-k}{2} \rfloor$, i.e. it requires error rate to be less than rougly $\frac{1}{2}(1 - \frac{k}{n}) \approx \frac{1}{2}$. But for some applications, we want to decode the corrupted codeword even when the error rate is close to 1. In this case, there can be multiple possible codewords correspond to the corrupted codeword. It leads to the notion of list-decodable code.

Definition 2.1 (list-decodable) *For $e \in (0, 1)$. A code $\mathcal{C} \subseteq \Sigma^n$ is an (e, ℓ)-*list-decodable code *if*

$$\left| \left\{ c \in \mathcal{C} \mid \frac{\Delta(x, c)}{n} \leq e \right\} \right| \leq \ell$$

for all $x \in \Sigma^n$

Note that when $\ell = 1$, list-decodable reduces to unique decodable. For most application, we would like to have a code with small (polynomial in n) list size ℓ under large error rate e. We state without proof the following result that relates the parameters of the code, error rate and the list size.

Theorem 2.2 (Johnson bound) *Let $\mathcal{C}$ be a $[n, k, d]_q$ code. If*

$$e \leq \left(1 - \frac{1}{q}\right)\left(1 - \sqrt{1 - \frac{q}{q-1} \cdot \frac{d}{n}}\right),$$

then $\mathcal{C}$ is (e, dqn)-decodable.

Recall that Reed-Solomon code is a $[n, k, n - k + 1]_q$-code where $q \geq n$, which gives $e \approx 1 - \sqrt{k/n}$ and thus Johnson bound says that it is list-decodable (with list size polynomial in n and q) up to error rate $1 - \sqrt{k/n}$, which is much larger than the error rate $\frac{1}{2}(1 - \frac{k}{n})$ in the unique decoding case.

3 List-decoding Reed-Solomon Codes

We show that the following algorithm by Sudan [1] list-decodes Reed-Solomon codes up to error rate $1 - 2\sqrt{k/n}$.

List-decoding for Reed-Solomon codes
Input: $\{(a_i, y_i)\}_{i=1,\dots,n}$

1. Find nonzero $Q \in \mathbb{F}_q[x, y]$ such that $\deg_x(Q) \le \sqrt{kn}$, $\deg_y(Q) \le \sqrt{\frac{n}{k}}$ and $Q(a_i, y_i) = 0$ for each $i \in [n]$.
2. Compute all factors of Q that are of the form $y - f(x)$.
3. Output all f from Step 2 such that $\{i \in [n] \mid f(a_i) = y_i\}| \ge 2\sqrt{kn}$.

Lemma 3.1 *There exists $Q(x, y)$ that satisfies the conditions in Step 1 of the algorithm.*

Proof: We observe that finding Q is again equivalent to solving linear system. By writing $Q(x, y) = \sum_{0 \le i \le \sqrt{kn}} \sum_{0 \le j \le \sqrt{n/k}} c_{i,j} x^i y^j$, the equation $Q(a_i, y_i) = 0$ for $i \in [n]$ gives n linear equations in the coefficients $c_{i,j}$'s. Note that there are $(\sqrt{kn} + 1)(\sqrt{n/k} + 1) > n$ unknowns and n equations. Since $c_{i,j} = 0$ for all i, j is a solution, i.e. there exists at least one solution, it follows that there exist many solutions and one of them must be nonzero. ■

Lemma 3.2 *Let $Q \in \mathbb{F}_q[x, y]$ that satisfies the conditions in Step 1 of the algorithm. If $\deg(f) < k$ and $|\{i \in [n] \mid f(a_i) = b_i\}| \ge 2\sqrt{kn}$, then $y - f(x) | Q(x, y)$.*

Proof: Let $I = \{i \in [n] \mid f(a_i) = y_i\}$. Then $Q(a_i, f(a_i)) = 0$ for all $i \in I$. It follows that the univariate polynomial $Q(x, f(x))$ has at least $|I| \ge 2\sqrt{kn}$ roots. But $Q(x, f(x))$ has degree less than $\deg_x(Q) + \deg(f)\deg_y(Q) < \sqrt{kn} + k\sqrt{n/k} = 2\sqrt{kn}$. Thus $Q(x, f(x)) \equiv 0$. It follows that $y - f(x)|Q(x, y)$. Indeed, we can write $Q(x, y) = (y - f(x))A(x, y) + R(x, y)$ where $\deg_y(R) < \deg_y(y - f(x)) = 1$. So $R(x, y)$ does not depend on y. Now $Q(x, f(x)) \equiv 0$ implies $R(x, y) = R(x) \equiv 0$. ■

Remark 3.3 *The algorithm can be implemented in polynomial time. Step 1 is just solving linear system. For Step 2, there exists efficient algorithm for finding factors of such form. And the number of factors are bounded by $\deg_y(Q)$, which is polynomial in n, so we can afford to check all the factors in Step 3 to see whether they are closed to the corrupted codeword.*

4 Reed-Muller codes

One limitation of Reed-Solomon code is that it requires large field, in particular, $q \ge n$. Reed-Muller codes are generalization of Reed-Solomon codes that can be defined on any field size, even over $\mathbb{F}_2$.

Specifically, a Reed-Muller code $\mathrm{RM}_q(r,m) : \mathbb{F}_q^{\binom{m+r}{r}} \to \mathbb{F}_q^{q^m}$ is a linear code over $\mathbb{F}_q$. The message $(c_{i_1,\ldots,i_n})_{0 \le i_1+\cdots+i_n \le r}$ is identified with the polynomial

$$Q(\mathbf{x}) = Q(x_1,\ldots,x_m) = \sum_{0 \le i_1+\cdots+i_m \le r} c_{i_1,\ldots,i_m} x_1^{i_1} \cdots x_m^{i_m},$$

which is a multivariate polynomial of total degree at most r in m variables. $\mathrm{RM}_q(r,m)$ maps Q to $(Q(\mathbf{x}))_{\mathbf{x} \in \mathbb{F}_q^m}$, i.e. the codeword is the evaluation of Q over all points in $\mathbb{F}_q^m$. We will see the parameters and decoding algorithm for it in the next lecture.

References

[1] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997.

[2] L.R. Welch and E.R. Berlekamp. Error correction for algebraic block codes, December 30 1986. US Patent 4,633,470.