6.885 Algebra and Computation	November 7, 2005
Lecture 16	
Lecturer: Madhu Sudan	*Scribe: Sergey Yekhanin*

1 Today

- Decoding of Reed-Solomon (RS) codes.
- Decoding of Chinese Remainder (CR) codes.

2 Error-correcting codes

The theory of error-correcting codes studies the ways one should add redundancy to data, in order to allow reliable transmission over noisy channels. The theory was founded by Claude Shannon in the 1940-s. Shannon proposed the following architecture:

$$m \in \Sigma^k \to \text{ENCODER} \to E(m) \in \Sigma^n \to \text{NOISY CHANNEL} \to y \approx E(m) \to \text{DECODER} \to m' = E(m).$$

Here

- m is the message one wants to transmit.
- $E(m)$ is the encoding of m. I.e. m plus some extra redundant bits.
- y is the corrupted version of $E(m)$. I.e. y agrees with $E(m)$ in most of the locations except those that got flipped in the channel during the transmission.
- m' is the corrected version of of $E(m)$. For a good and appropriately used error-correcting code m' should (most of the time) be equal to $E(m)$.

3 Reed-Solomon code

In this section we consider a classical error-correcting code known as a Reed-Solomon code.

Assume we have a bijection between our message alphabet Σ and some finite field F_q. Fix some subset $M \subseteq F_q$. Let $M = \{\alpha_1, \ldots, \alpha_n\}$.

We represent messages $m = (m_0, \ldots, m_{k-1}) \in \Sigma^k$ by univariate polynomials

$$m(x) = \sum_{i=0}^{k-1} m_i x^i \in F_q[x].$$

We define the encoding of m to be the evaluation of the corresponding polynomial at every point of the set M. I.e.

$$Enc(m) = \{m(\alpha_1), \ldots, m(\alpha_n)\}.$$

After some bits of $\{m(\alpha_1), \ldots, m(\alpha_n)\}$ are flipped in the channel decoder gets the sequence $\{y_1, \ldots, y_n\}$ as an input. Assuming that the number of errors in the channel is upper bounded by $(n - t)$, the goal of the decoder is to find a polynomial (or better - all polynomials) $m \in F_q[x]$ such that $m(\alpha_i) = y_i$ for al least t values of $i \in [1, n]$.

It is convenient to think of pairs (α_i, y_i) as points in the plane F_q^2. Our goal is to find all curves of the form $y - m(x) = 0$, (where $m(x)$ is of degree $\leq k - 1$) that pass through at least t points of the set $S = \{(\alpha_i, y_i)\}_{i \in [1,n]}$.

Instead of trying to find the curves of the form $y - f(x)$ directly, we will first fit all the points of the set S to some low-degree curve (with no other restriction on the form of the equation defining the curve). It is easy to see that there exists a bivariate polynomial $Q(x, y)$ where $\deg_x Q \leq \sqrt{n}$ and $\deg_x Q \leq \sqrt{n}$ such that $Q(\alpha_i, y_i) = 0$ for all $(\alpha_i, y_i) \in S$. Moreover one can compute the polynomial $Q(x, y)$ efficiently in time $O(n^3)$ by solving a system of n homogeneous linear equations in the coefficients of Q.

Given the polynomial $Q(x, y)$ we want to claim that for every polynomial $m(x)$ such that $y - m(x)$ contains sufficiently many points from S, $y - m(x)|Q(x, y)$. Formally,

Claim 1: Assume the following hold:

- $\deg_x Q \leq D$, $\deg_y Q \leq D$,
- $\deg m(x) \leq k - 1$,
- $Q(\alpha_i, y_i) = y_i - m(\alpha_i) = 0$ for at least t values of $i \in [1, n]$,
- $t > 2(D + 1)k$;

then $y - m(x)|Q(x, y)$.

Proof: It is clear that the polynomial $y - m(x)$ is irreducible. Assume $y - m(x) \not| Q(x, y)$; then there exist polynomials $A(x, y), B(x, y) \in F_q[x, y]$ such that

$$R(x) = A(x, y)Q(x, y) + B(x, y)(y - m(x)).$$

is non-zero. Namely, $R(x)$ is a resultant of Q and $y - m(x)$ computed with respect to y. From the degree bound for the resultant we conclude that $\deg R(x) \leq 2(D + 1)k$. However $R(\alpha_i) = 0$ for all α_i such that $Q(\alpha_i, y_i) = y_i - m(\alpha_i) = 0$. Therefore $R(x)$ has at least $2(D + 1)k + 1$ roots. Thus we arrive at a contradiction. Proof complete.

Given the claim above we are ready to formulate the (list) decoding algorithm for Reed-Solomon codes:

Algorithm 1:

1. Input: $k, n, \{(\alpha_i, y_i)\}_{i \in [1,n]}$.
2. Find $Q(x, y)$ such that $\deg_x Q \leq \sqrt{n}$, $\deg_y Q \leq \sqrt{n}$, $Q(\alpha_i, y_i) = 0$, and $Q(x, y) \neq 0$.
3. Find all factors of $Q(x, y)$ of the form $y - m(x)$.
4. Output: A list of polynomials $m(x)$ such that $y - m(x)|Q(x, y)$ and $y - m(x)$ passes through at least t points (α_i, y_i).

Claim 1 implies that our algorithm successfully decodes RS code from up to $2\sqrt{n}k$ agreement.

Exercise 1: Improve the decoding algorithm to decode successfully from $t > \min\{(n - k)/2, 2\sqrt{kn}\}$ agreement.

We conclude our discussion of decoding algorithm for RS codes with a historical overview:

- The first decoding algorithm for RS codes was developed by Peterson in the sixties. The algorithm runs in time $O(n^3)$. Petersen claimed his algorithm to be *efficient* as it avoided the brute-force search. Note that this work precedes the work of Edmonds!
- Later the running time was brought down to $O(n^2)$ by Berlekamp. Berlekamp's algorithm relies on efficient randomized factorization of univariate polynomials. (Which is also due to Berlekamp.)
- The algorithm that we have just seen was developed by Sudan in 1996. It allows to correct a larger number of errors than previously known algorithms. The algorithm relies on efficient factorization of multivariate polynomials.

4 Chinese Remainder code

We use p_i to denote the i-th prime. Let $K = \prod_{i=1}^{k} p_i$. Assume our message m is a sequence of $\log K$ bits. We can think of it as an integer $0 \le m \le K-1$. Let n be an integer such that $k \le n$. the Chinese Remainder encoding of m is:

$$Enc(m) = \{(m \bmod p_1), \ldots, (m \bmod p_n)\}.$$

Clearly, any k coordinates of the $Enc(m)$ suffice to reconstruct the message m. This follows from the standard CRT "interpolation".

The decoding problem for the CR code is the following. Given integers $\{r_1, \ldots, r_n\}$ find some integer $m \in [0, K-1]$ (or better - all such integers), such that $m \bmod p_i = r_i$ for at least t values of i. In what follows we will sketch the solution of this problem assuming $t \ge 2k\sqrt{n} \log p_n / \log p_1$.

Our approach is to extend the technique we have for decoding of RS codes. We start by building some informal dictionary between Z and $F_q[x]$.

$$\begin{array}{ccc} Z & & F_q[x] \\ \text{small integer} & \leftrightarrow & \text{low degree polynomial} \\ Z[y] & \leftrightarrow & F_q[x,y] \\ Q(r_i) = 0 \bmod p_i & \leftrightarrow & Q(\alpha_i, y_i) = 0. \end{array}$$

Using the dictionary above we "translate" the algorithm for decoding of RS codes into an algorithm for decoding of CR codes.

Algorithm 2:

1. Input: $k, n, \{(p_i, r_i)\}_{i \in [1,n]}$.
2. Find $Q(y) \in Z[y]$ such that $\deg Q \le \sqrt{n}$, $Q(r_i) = 0 \bmod p_i$, $Q(y) \ne 0$, and all coefficients of $Q(y)$ are small. (The particular meaning of small that we need here is $\le p_n^{\sqrt{n}}/2$ in the absolute value.)
3. Find all linear factors of $Q(y)$. They are of the form $y - m$.
4. Output: A list of integers m such that $y - m | Q(y)$ and $m = r_i \bmod p_i$ for at least t values of i.

There is a simple counting argument that allows one to conclude that a polynomial $Q(y)$ that we want to find in step 1 really exists. One needs to look at the number of polynomails of degree $\le \sqrt{n}$ with coefficients in the range $[-p_n^{\sqrt{n}}/2, p_n^{\sqrt{n}}/2]$, and compare this number to $\prod_{i=1}^{n} p_i$. However, finding such a $Q(y)$ is no longer a linear algebra problem. Luckily it can be reduced to finding short vector in the lattice and thus be (approximately) resolved using the LLL algorithm.

Exercise 2: Prove the correctness of step 2 of the decoding algorithm for CR codes.