Lecture 11: Locally Decodable Codes

Error-Correcting Codes (Spring 2016)
Rutgers University
Swastik Kopparty
Scribe: Alex Conway

1 Definition of a Locally Decodable Code

Let $C \subseteq \Sigma^n$ be a code with encoding map $E : \Sigma^k \to C$. Then C is called a q, ϵ-*locally decodable code* if there exists an algorithm A such that for all messages $x \in \Sigma^k$ and all $r \in \Sigma^n$ such that $\Delta(r, E(x)) < \epsilon$, we have

$$\Pr(A(i, r) \neq x_i) < 0.01,$$

for all $i \in [k]$, and moreover, A accesses only q coordinates of r.

It is important here that the probability is only over the internal randomness of the algorithm A. There is no randomness over $i \in [k]$: the algorithm needs to work for all i.

In general, we will be interested in achieving local decodability with query complexity $q = o(k)$. In this lecture we will see subexponential

2 Recap: Reed-Muller Locally Decodable Codes

Reed-Muller codes turn out to be locally decodable with interesting parameters. This is the first time we see Reed-Muller codes achieve something that we could not achieve with simply Reed-Solomon codes.

Consider the Reed-Muller code of degree-d m-variate polynomials over F_q. The choice of encoding map turns out to be important here. Choose a set $S \subseteq \mathbb{F}_q^m$ which is an *interpolating set* for degree-d m-variate polynomials over $\mathbb{F}_q$. This means that for any assignment $u : S \to \mathbb{F}_q$, there is a unique degree-d polynomial $P(X_1, \ldots, X_m)$ such that $P|_S = u$ (in particular we have that $|S| = \binom{d+m}{m}$). We will use S to specify the encoding map: given a message $u \in \mathbb{F}_q^{\binom{d+m}{m}}$, we view it as a function $u : S \to \mathbb{F}_q$; The encoding of u is then the unique polynomial extension P.

Suppose we are given $r : \mathbb{F}_q^n \to \mathbb{F}_q$, with the promise that r is close to some polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$ of degree less than or equal to d. The problem of local decoding is now the problem of recovering $P(x)$ for a given $x \in S$ (we will even be able to recover $P(x)$ for arbitrary $x \in \mathbb{F}_q^m$).

The main observation is that restricting a low-degree multivariate polynomial to a line gives a low-degree univariate polynomial. This motivates the following algorithm.

- Pick a random line ℓ through x.

- Query r on $\ell \setminus \{x\}$.
- We should see a univariate polynomial of degree less than or equal to d.
- Use this to deduce $p(x)$.

2.1 Parameters: Constant Query

With t queries, Reed-Muller codes have length:

$$k \to n^{O\left(k^{\frac{1}{t-\Sigma}}\right)}.$$

There are known lower bounds:

- For $t = 2$, we must have $n = 2^{\Omega(k)}$.
- For $t > 2$, we must have $n \geq k^{1+O(1/t)}$.

It is an open question whether it is possible to have $t = O(\log k)$ and $n = O(k)$.

2.2 Parameters: Constant rate

For m-variable Reed-Muller codes, we have:

$$k \to m! \cdot 2^{O(m)} \cdot k,$$

with rate $R = \frac{1}{m!} \cdot \frac{1}{2^{O(m)}}$, and $t = k^{1/m}$ queries. (We'll see another code that gives a better constant rate later...)

2.3 State of the art

The best known constructions give:

- For fixed t, we can have $n = 2^{2^{\tilde{O}\left((\log k)^{1/\log t}\right)}}$.
- For fixed rate R, we can have $t = 2^{\sqrt{\log k}}$.

In this lecture, we will see the first of these constructions. To motivate it, we will begin with a new construction of a 2-query locally-decodable code.

3 Another Hadamard-like Locally Decodable Code

We will write $\mathbb{F}_3 = \{0, 1, -1\}$. Given a message $c \in \mathbb{F}_3^k$, consider

$$\begin{aligned} f &: \mathbb{Z}_2^k \to \mathbb{F}_3 \\ f &: x \mapsto \sum_{i=1}^{k} c_i \cdot (-1)^{\langle x, e_i \rangle}, \end{aligned}$$

where e_i is the i^{th} standard basis vector.

We use the following method to locally decode c_j. Given $r : \mathbb{Z}_2^k \to \mathbb{F}_3$, pick x uniformly in $\mathbb{Z}_2^k$. Then query $r(x) \approx f(x)$ and $r(x + e_j) \approx f(x + e_j)$, where by $a \approx b$, we mean a and b are "supposed to be equal." Consider

$$\begin{aligned} r(x + e_j) - r(x) &\approx f(x + e_j) - f(x) \\ &= \sum_{i=1}^{k} c_i \left[(-1)^{\langle x + e_j, e_i \rangle} - (-1)^{\langle x, e_i \rangle} \right] \\ &= c_j (-1)^{\langle x, e_j \rangle} \left((-1)^{\langle e_j, e_j \rangle} - 1 \right) \\ &= c_j (-1)^{x_j}. \end{aligned}$$

Therefore, we output $(-1)^{x_j} \left(r(x + e_j) - r(x) \right)$.

4 Matching Vector Codes

These amazing codes were constructed by Efremenko, based on an important breakthrough of Yekhanin (see also the paper of Raghavendra, and the follow-up papers of Dvir-Gopalan-Yekhanin and BenAroya-Efremenko-TaShma).

Let $S \subseteq \mathbb{Z}_m$. Then we say a collection of vectors $u_1, v_1, u_2, v_2, \ldots, u_k, v_k$ is S-matching if

$$\begin{aligned} \langle u_i, v_j \rangle &= 0 \quad \text{for } i = j \\ \langle u_i, v_j \rangle &\in S \quad \text{for } i \neq j. \end{aligned}$$

As an aside, if $S = \mathbb{Z}_m \setminus \{0\}$, m=2, how large can an S-matching collection be? One example is with $u_i = (1, e_i) = v_i$, where we have $k = h - 1$. Note that if we do a similar adjoining to a S-matching collection, we get a set of orthogonal vectors, so this limits $k \leq h$ when $m = 2$.

Now, let $u_1, v_1, u_2, v_2, \ldots, u_k, v_k$ be a matching vector family in $\mathbb{Z}_m^k$. Let $\mathbb{F}$ be some finite field with $m \big| (|\mathbb{F}| - 1)$, and let $\omega \in \mathbb{F}^*$ be a primitive m^{th} root of unity in $\mathbb{F}$. Define

$$\begin{aligned} \chi_i &: \mathbb{Z}_m^n \to \mathbb{F} \\ \chi_i &: x \mapsto \omega^{\langle x, u_i \rangle}. \end{aligned}$$

For a message $c \in \mathbb{F}^k$, define the codeword $f : \mathbb{Z}_m^k \to \mathbb{F}$ by

$$f = \sum_{i=1}^{k} c_i \chi_i.$$

To locally decode (and recover c_j), we first pick $x \in \mathbb{Z}_m^n$ uniformly at random. Then we query $r(x), r(x+v_j), r(x+2v_j), \ldots, r(x+(m-1)v_j)$, which "should be equal to" $f(x), f(x+v_j), \ldots, f(x+(m-1)v_j)$. (Note that if m is a constant, then this is a constant query decoder.) Now,

$$\begin{aligned} f(x + \lambda v_j) &= \sum_{i=1}^{h} c_i \chi_i(x + \lambda v_j) \\ &= c_j \chi_j(x + \lambda v_j) + \sum_{i \neq j} c_i \chi_i(x + \lambda v_j) \end{aligned}$$

The first term is equal to $c_j \omega^{\langle x, u_j \rangle}$, by the definition of χ_j and because $\langle u_j, v_j \rangle = 0$. Then the second term

$$\begin{aligned} \sum_{i \neq j} c_i \chi_i(x + \lambda v_j) &= \sum_{i \neq j} c_i \omega^{\langle x + \lambda v_j, u_i \rangle} \\ &= \sum_{i \neq j} c_i \omega^{\langle x, u_i \rangle} \omega^{\lambda \langle v_j, u_i \rangle} \\ &= \sum_{\alpha \in S} \left(\sum_{\langle v_j, u_i \rangle = \alpha} c_i \omega^{\langle x, u_i \rangle} \right) \omega^{\lambda \alpha} \\ &= \sum_{\alpha \in S} B_\alpha \omega^{\lambda \alpha}, \end{aligned}$$

where B_α is defined to be the inner sum for each α.

Thus, as a function of λ,

$$f(x + \lambda v_j) = c_j \omega^{\langle x, u_j \rangle} + \sum_{\alpha \in S} B_\alpha \omega^{\lambda \alpha}.$$

In other words, to decode we just need to find the constant term. The procedure is then to recover the coefficients B_0, $\{B_\alpha\}_{\alpha \in S}$ such that

$$f(x + \lambda y) = B_0 + \sum_{\alpha \in S} B_\alpha \omega^{\alpha \lambda},$$

which is solving an interpolation problem, where we need only query $|S| + 1$ values of λ. Then output $B_0 \cdot \omega^{-\langle x, u_j \rangle}$.

4.1 Parameters of the Locally Decodable Code in terms of Parameters of the Matching Vector Family

We have $n = m^h$, $k = k$, and the number of queries is $|S| + 1 \leq m$

We are happy if S (or m) is constant, with k as large as possible; then we can handle $\frac{1}{100|S|}$ errors. This begs the question: how large a matching vector family can you construct with these parameters?

4.2 Constructing a Matching Vector Family

We will first consider the case where m is prime, then later we will take m composite. For now, consider the case where m is prime, $S = \{1\}$, and our vectors are in $\mathbb{Z}_m^\ell$. Take each $\widetilde{u}_i$ to be a vector with 1's in exactly $m-1$ places, and 0's elsewhere, and $\widetilde{v}_i$ to be the vector (with $\ell - (m-1)$ 1's) such that $\widetilde{u}_i + \widetilde{v}_i$ is the vector with 1's in every place. These vectors have the property that:

$$\begin{aligned}&\langle \widetilde{u}_i, \widetilde{v}_i \rangle = 0\\&\langle \widetilde{u}_i, \widetilde{v}_j \rangle \in \{1, \ldots, m-1\} \quad \text{for } i \neq j\end{aligned}$$

Before continuing, let's review the definition of the tensor product. Given $a \in \mathbb{F}^k, b \in \mathbb{F}^\ell$, $a \otimes b \in \mathbb{F}^{k\ell}$ is the vector $(\ldots, a_i b_j, \ldots)$, which is to say the vector with $a_i b_j$ in the $(i,j)^{\text{th}}$ place. It follows from the definition that $\langle a \otimes b, c \otimes d \rangle = \langle a, c \rangle \langle b, d \rangle$.

We now define $u_i = (\widetilde{u}_i)^{\otimes(m-1)}$ and $v_i = (\widetilde{v}_i)^{\otimes(m-1)}$. The above property implies that

$$\begin{aligned}&\langle u_i, v_i \rangle = 0\\&\langle u_i, v_j \rangle = 1 \quad \text{for } i \neq j.\end{aligned}$$

This construction gives us $k = \binom{\ell}{m-1}$ and $h = \ell^{m-1}$.

Now take m to be the product of two distinct primes: $m = pq$. Take $\widetilde{u}_i$ and $\widetilde{v}_i$ as above, and let

$$\begin{aligned}u_i &= \left(A(\widetilde{u}_i)^{\otimes(p-1)}, B(\widetilde{u}_i)^{\otimes(q-1)}\right)\\u_i &= \left((\widetilde{v}_i)^{\otimes(p-1)}, (\widetilde{v}_i)^{\otimes(q-1)}\right).\end{aligned}$$

Then, $\langle u_i, v_j \rangle = A\langle u_i, v_j \rangle^{p-1} + B\langle u_i, v_j \rangle^{q-1}$. Now choose A and B such that

$$\begin{matrix}A \equiv 0 \pmod{q}, & A \equiv 1 \pmod{p}\\B \equiv 1 \pmod{q}, & B \equiv 0 \pmod{p}\end{matrix},$$

which is possible by the Chinese Remainder Theorem.

Thus,

$$\langle u_i, v_j \rangle \pmod{p} = \begin{cases}0 & \langle \widetilde{u}_i, \widetilde{v}_j \rangle \equiv 0 \pmod{p}\\1 & \text{otherwise}\end{cases}$$
$$\langle u_i, v_j \rangle \pmod{q} = \begin{cases}0 & \langle \widetilde{u}_i, \widetilde{v}_j \rangle \equiv 0 \pmod{q}\\1 & \text{otherwise}\end{cases}$$

Therefore $\langle u_i, v_j \rangle$ takes one of four values mod n. However, $\langle u_i, v_j \rangle = 0 \pmod{m}$ is equivalent to $\langle \widetilde{u}_i, \widetilde{v}_j \rangle = 0 \pmod{p}$ and $\pmod{q}$, which is turn is equivalent to $\langle \widetilde{u}_i, \widetilde{v}_j \rangle = 0 \pmod{m}$, which means $i = j$.

Finally, take $p, q \approx \sqrt{m}$. Then, we get parameters $|S| = 3$, $h = \ell^{\sqrt{m}}$, and $k = \binom{\ell}{m}$. We may also write $h = 2^{\sqrt{\log k \log \ell}}$.

Generalizing to m being a product of more primes, we get locally decodable codes with larger constant query complexity and reduced codeword length (as a function of the message length).