Lecture 8: Local Codes

Sivakanth Gopi

October 21, 2019

In this part of the course, we will talk about the notion of "locality". We can construct codes with good rate and distance which are encodable and decodable in near-linear time (e.g., Expander codes). And over large alphabets, we can achieve optimal rate-distance tradeoff with near-linear time encoding and decoding (e.g., Reed-Solomon codes). We will also see later in the course that we can now construct optimal binary codes for many random noise channels with near-linear time encoding and decoding (e.g., Polar codes). This is amazing progress. Can we do even better? In many scenarios, both theoretical and practical, we will see that we only need to decode a small part of the message or correct a small part of the corrupted encoding. For example in distributed storage, to recover a single crashed server, we shouldn't be reading all the rest of the servers. In such cases, linear time is not good enough, we need to do things in sublinear time or perhaps even in constant time! Codes which support sublinear algorithms for tasks like decoding, correction or testing are called "local codes".

Codes with locality

Let $C : \Sigma^k \to \Sigma^n$ be some code. Given some string z which is close to some codeword $C(x)$ in Hamming distance, there are three natural algorithmic tasks we would like to do:

- Correction: Correct the errors in z to get $C(x)$
- Decoding: Decode z to get x
- Testing: Test whether z is actually close to some codeword.

Local codes allow us to do these tasks in sublinear time. We will define locally decodable codes formally and give informal definitions for locally correctable codes and locally testable codes. A q-query locally decodable code (LDC) allows us to decode any message bit x_i with high probability by reading at most q locations of z. We will define LDCs formally now.

Definition 1 (Locally decodable code (LDC) [KT00]). *A code $C : \Sigma^k \to \Sigma^n$ is a (q, δ, η)-LDC if, for every $i \in [k]$, there exists a randomized decoder (a probabilistic algorithm) $\mathcal{A}_i$ such that:*

- *For every message $x \in \Sigma^k$ and $z \in \Sigma^n$ such that $\Delta(C(x), z) \leq \delta n$,*

$$\Pr[\mathcal{A}_i(z) = x_i] \geq \frac{1}{2} + \eta. \qquad (1)$$

- *The decoder $\mathcal{A}_i$ queries non-adaptively at most q coordinates of z.*

Locally Correctable Code: C is called a (q, δ, η)-LCC if there is a randomized local correction algorithm which given some $i \in [n]$, reads at most q locations of z, which is δ-close to $C(x)$, and outputs $C(x)_i$ with good probability at least $1/2 + \eta$.

Any LCC can be converted into an LDC while preserving relevant parameters. In fact any linear code can be made systematic i.e. the message is part of the encoding. Therefore any linear LCC is also an LDC trivially.

Locally Testable Code: C is called a q-query LTC if there is a randomized local testing algorithm which reads at most q locations of z and accepts if there are no corruptions, and rejects with good probability if there are too many corruptions.

Hadamard Code

To get familiar with the definitions, let us look at the example of Hadamard code which is simultaneously a 2-query LDC, 2-query LCC and 3-query LTC!

The local correctability and testability of Hadamard code plays an important role in the proof of the PCP theorem.

The Hadamard code is a exponential length linear code, $H : \mathbb{F}_2^k \to \mathbb{F}_2^n$ where $n = 2^k$. The codeword coordinates are indexed by $y \in \mathbb{F}_2^k$ and for a message $x \in \mathbb{F}_2^k$, the encoding is given by $H(x)_y = \langle x, y \rangle$ i.e. the codewords are just *evaluations of linear functions* on $\mathbb{F}_2^k$. It is not hard to see that it is a linear code and the minimum distance of the code is $n/2$, in fact every non-zero codeword has weight exactly $n/2$. We will now prove that the Hadamard is a 2-query LCC. We can achieve local correction way up to half-minimum-distance.

Lemma 2. *For every $\delta \in (0, 1/4)$, Hadamard code is a $(2, \delta, 1/2 - 2\delta)$-LCC.*

Proof. Suppose we are given a corrupted version of a codeword $H(x)$, say $\widetilde{H(x)}$ s.t. $\Delta(H(x), \widetilde{H(x)}) \leq \delta n$. To correct the symbol at $y \in \mathbb{F}_2^k$, the local corrector queries $\widetilde{H(x)}$ at $z, z + y$ for a uniformly random $z \in \mathbb{F}_2^k$ and computes the parity of the two bits. With probability at least $1 - 2\delta$, both the queries land in the uncorrupted part of $\widetilde{H(x)}$, and if that's the case,

$$\widetilde{H(x)}_z + \widetilde{H(x)}_{z+y} = \langle x, z \rangle + \langle x, z + y \rangle = \langle x, y \rangle$$

which is the correct symbol. □

Since the message symbols are part of the codeword ($H(x)_{e_i} = \langle x, e_i \rangle = x_i$), it is also a 2-query LDC.

To test if some given word is close to some codeword is equivalent to testing if a function $f : \mathbb{F}_2^k \to \mathbb{F}_2$ is close to being linear. To test this, a local tester can sample $z, y \in \mathbb{F}_2^k$ and query f at $z, y, z+y$ and accept if

$$f(z+y) = f(z) + f(y).$$

This is the famous linearity test of Blum, Luby and Rubinfeld. It clearly accepts a linear function and it will reject a function which is far from linear with good probability (which requires a proof, but we omit it here). Thus H is a 3-query LTC.

Reed-Muller Codes

We have seen that the Hadamard code, which is obtained by evaluating linear functions at all points of $\mathbb{F}_q^k$ has good local correction, decoding and testing properties. But it is exponentially long. It is natural to try evaluating all low-degree polynomials over $\mathbb{F}_q^k$. Since there are more low-degree polynomials, the rate of our codes will improve. These are precisely the *Reed-Muller Codes*.

Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ be an m-tuple of non-negative integers and let $|\alpha| = \sum_{i=1}^m \alpha_i$ denote the sum of its entries. Let $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ be a tuple of variables. We will denote the monomial $x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_m^{\alpha_m}$ by $\mathbf{x}^\alpha$. Note that the degree of $\mathbf{x}^\alpha$ is $|\alpha|$. An m-variate polynomial over $\mathbb{F}_q$ is given by

$$f(x_1, x_2, \ldots, x_m) = \sum_\alpha c_\alpha \mathbf{x}^\alpha$$

where $c_\alpha \in \mathbb{F}_q$ are some coefficients. The set of all such polynomials is denoted by $\mathbb{F}_q[x_1, \ldots, x_m]$. The (total) degree of the polynomial f is defined as $\deg(f) = \max\{|\alpha| : c_\alpha \neq 0\}$. The set of all degree such polynomials of degree at most d is denoted by $\mathbb{F}_q^{\leq d}[x_1, \ldots, x_m]$. We will also use $\deg_{x_i}(f)$ to denote the maximum degree of x_i variable in f, i.e., $\deg_{x_i}(f) = \max\{\alpha_i : c_\alpha \neq 0\}$. Let $\Pi_q(m, d)$ denote the set of all polynomials in $\mathbb{F}_q^{\leq d}[x_1, \ldots, x_m]$ with individual degrees of each variable at most $q-1$, i.e.,

$$\Pi_q(m, d) = \{f \in \mathbb{F}_q^{\leq d}[x_1, \ldots, x_m] : \forall i \ \deg_{x_i}(f) \leq q-1\}.$$

When $q > d$, $\Pi_q(m, d) = \mathbb{F}_q^{\leq d}[x_1, \ldots, x_m]$. For example, $\Pi_2(m, d)$ is the set of all m-variate "multilinear" polynomials of degree at most d.

The restriction that $\deg_{x_i}(f) \leq q-1$ is because of the fact that $x_i^q = x_i$ for all $x_i \in \mathbb{F}_q$. Therefore, if we only care about evaluations over $\mathbb{F}_q$, we can reduce the degree of each variable to at most $q-1$.

Fact 3. *There is a bijection between $\Pi_q(m, d)$ and evaluations of polynomials in $\mathbb{F}_q^{\leq d}[x_1, \ldots, x_m]$ over all points of $\mathbb{F}_q^m$. The bijection is given by $f \in \Pi_q(m, d) \to \langle f(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q^m}$. In particular, two distinct polynomials in $\Pi_q(m, d)$ cannot have the same evaluation over all points of $\mathbb{F}_q^m$.*

Definition 4. *A degree-d Reed-Muller code, denoted by $\mathrm{RM}_q(m, d)$, is the set of evaluations of polynomials $\Pi_q(m, d)$ over all points in $\mathbb{F}_q^m$. Formally,*

$$\mathrm{RM}_q(m, d) = \left\{ \langle f(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q^m} : f \in \Pi_q(m, d) \right\}.$$

Note that Reed-Muller codes are linear codes, because an $\mathbb{F}_q$-linear combination of polynomials in $\Pi_q(m,d)$ is also in $\Pi_q(m,d)$. We will now try to understand the rate and distance of Reed-Muller codes. $\mathrm{RM}_q(m,d)$ is a subspace of $\mathbb{F}_q^n$ for $n = q^m$. The dimension of $\mathrm{RM}_q(m,d)$ is:

Is $\mathrm{RM}_2(m,1)$ the same the Hadamard code? Almost, think about it!

$$\dim(\mathrm{RM}_q(m,d)) = \dim(\Pi_q(m,d)) = \left|\{\alpha : |\alpha| \le d,\ 0 \le \alpha_i \le q-1\}\right|.$$

There is no simple closed form expression for $\dim(\mathrm{RM}_q(m,d))$. For $q = 2$, $\dim(\mathrm{RM}_2(m,d)) = \sum_{r=0}^{d}\binom{m}{r} = \binom{m}{\le d}$.

To simplify presentation, from now onwards, we will assume that $q > d$ **unless otherwise specified.** In this case, $\dim(\mathrm{RM}_q(m,d)) = \binom{m+d}{d} \gtrsim_d m^d$. We will now show that $\mathrm{RM}_q(m,d)$ also have good distance.

Lemma 5 (Schwartz-Zippel)**.** *Suppose $q > d$ and let $f \in \mathbb{F}_q^{\le d}[x_1,\ldots,x_m]$ be a non-zero polynomial. Then:*

$$\Pr_{\mathbf{a}\in\mathbb{F}_q^m}[f(\mathbf{a}) = 0] \le \frac{d}{q}.$$

By Lemma 5, any non-zero codeword of $\mathrm{RM}_q(m,d)$ has weight at least $(1-(d/q))n$. Therefore the relative distance of $\mathrm{RM}_q(m,d)$ is at least $1-(d/q)$.

Local correctability of Reed-Muller codes

We will now show that $\mathrm{RM}_q(m,d)$ is a q-query LCC.

Lemma 6. *Suppose $d < q-1$ and $(d+1)\delta < 1/2$. Then $\mathrm{RM}_q(m,d)$ is a $\left(d+1, \delta, \frac{1}{2} - (d+1)\delta\right)$-LCC.*

Proof. Suppose we are given some codeword $\tilde{f}$ which is δ-close to some codeword obtained by the evaluation of a polynomial $f \in \mathbb{F}_q[x_1,\ldots,x_m]$ at all points of $\mathbb{F}_q^m$. There are two main ideas.

- The restriction of f onto a line in $\mathbb{F}_q^m$ is a univariate polynomial of degree at most d. Let $\ell(\lambda) = \mathbf{a} + \lambda\mathbf{b}$ ($\lambda \in \mathbb{F}_q$) be a line passing through $\mathbf{a} \in \mathbb{F}_q^m$ in direction $\mathbf{b} \in \mathbb{F}_q^m$. Then the restriction of f to ℓ is given by $f|_\ell(\lambda) = f(\mathbf{a} + \lambda\mathbf{b})$ which is a univariate polynomial of degree at most d in λ.

- Given the values of a univariate polynomial $p(\lambda) \in \mathbb{F}_q[\lambda]$ of degree at most d at $d+1$ points, we can find p (which will be unique) using interpolation.

Now say we want to correct the corrupted codeword $\tilde{f}$ at a point $\mathbf{a} \in \mathbb{F}_q^m$, i.e., we want to find $f(\mathbf{a})$. Then we can pass a line through

$\mathbf{a}$ in a random direction $\mathbf{b} \in \mathbb{F}_q^m \setminus \{0\}$ and query $\tilde{f}$ at $d+1$ points (other than $\mathbf{a}$) on the line $\ell(\lambda) = \mathbf{a} + \lambda\mathbf{b}$. Since each point on the line other than $\mathbf{a}$ is uniformly distributed, with probability at least $1-(d+1)\delta$, $\tilde{f}$ is uncorrupted at the these points, i.e., it agrees with f. Therefore we will know the values of $p(\lambda) = f|_\ell(\lambda)$ at $d+1$ points with that probability. Now we can use interpolation to find p and thus $p(0) = f|_\ell(0) = f(\mathbf{a})$. □

Note that the using Lemma 6, we can only tolerate $\delta < \frac{1}{2(d+1)}$ corruptions. Ideally, we want to be able to do local correction up to half the minimum distance i.e. $\frac{1}{2} \cdot (1-d/q)$. One can improve the above decoding by querying it on q points as follows.

Lemma 7. *Suppose $d < q$ and $\delta < \frac{1}{4}(1-d/q)$. Then $\mathrm{RM}_q(m,d)$ is a $\left(q, \delta, \frac{1}{2} - \frac{2\delta}{1-d/q}\right)$-LCC.*

Proof. Suppose we are given some codeword $\tilde{f}$ which is δ-close to some codeword obtained by the evaluation of a polynomial $f \in \mathbb{F}_q[x_1, \ldots, x_m]$ at all points of $\mathbb{F}_q^m$. Say we want to correct the corrupted codeword $\tilde{f}$ at a point $\mathbf{a} \in \mathbb{F}_q^m$, i.e., we want to find $f(\mathbf{a})$. For a random direction $\mathbf{b} \in \mathbb{F}_q^m \setminus \{0\}$, query $\tilde{f}$ at all points on the line $\ell(\lambda) = \mathbf{a} + \lambda\mathbf{b}$. Let $p(\lambda) = f(\mathbf{a} + \lambda\mathbf{b})$ which is a univariate polynomial of degree at most d. Now we are given $\tilde{p}(\lambda) = \tilde{f}(\mathbf{a} + \lambda\mathbf{b})$ which is a noisy version of p. How do we find p from $\tilde{p}$? This is precisely Reed-Solomon decoding! So we can use the Berlekamp-Welch algorithm we discussed in class previously to decode from half the minimum distance of degree d Reed-Solomon codes. So, as long as, the number of errors in $\tilde{p}$ is less than $\frac{1}{2}(1-d/q)$ we can find p. In expectation, the number of errors in $\tilde{p}$ is at most δ. By Markov inequality, the probability the number of errors in $\tilde{p}$ will be more than $\frac{1}{2}(1-d/q)$ is at most $2\delta/(1-d/q)$. Once we find p, we can find $f(\mathbf{a})$ as $p(0) = f(\mathbf{a})$. □

Lemma 7 allows local correction as long as δ is at most $(1/4)^{th}$ of the minimum distance. This is already a big improvement over Lemma 6. To get local correction closer to half-minimum-distance, we just need one more idea. We need to query $\tilde{f}$ on a random degree two curve passing through $\mathbf{a}$ instead of a random line! Because points on a random two curve through $\mathbf{a}$ are pairwise independent, we can use Chebychev inequality instead of Markov. This allows us to do local correction up to $\frac{1}{2}(1-2d/q)$ distance which is closer to half-minimum-distance.

Can you think of a way to locally correct Reed-Muller codes all the way up to half-minimum-distance?

Table 1 shows the parameters of q-query LCCs $C : \Sigma^k \to \Sigma^n$ that can be achieved using Reed-Muller codes.

In particular, Reed-Muller can achieve n^ε-query complexity with constant δ and rate $\varepsilon^{1/\varepsilon}$. In fact, Reed-Muller codes with locality of

q	n
$q = O(1)$	$\exp\left(O_q(k^{1/(q-1)})\right)$
$\log n$	$k^{O(\log\log k)}$
$(\log n)^t, t > 1$	$k^{1+1/(t-1)+o(1)}$
$n^{1/t}, t \geq 1$	$t^{t+o(t)} \cdot k$

Table 1: Local correctability of Reed-Muller codes. The fraction of errors which can be tolerated by the local correction algorithm, δ is some fixed constant.

$n^{1/2}$ cannot have rate more than $1/2$. Can we construct n^{ε}-query LCCs with rate approaching 1 for every $\varepsilon > 0$? Yes, multiplicity codes from [KSY14] achieve this. The key idea is polynomial interpolation using derivatives. Multiplicity codes are evaluations of low-degree polynomials along with all partial derivatives up to a certain order.

References

[KSY14] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Journal of the ACM (JACM)*, 61(5):28, 2014.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd annual ACM symposium on Theory of computing (STOC 2000)*, pages 80–86. ACM Press, 2000.