CS 229r Essential Coding Theory Apr 13, 2020

Lecture 20

Instructor: Madhu Sudan *Scribe: Noah Singer*

1 Introduction

Last time, we defined the notion of an *ℓ-locally-recoverable code*, in which any character of a codeword is completely determined by a set of ℓ other characters; we proved a combinatorial upper bound on the rate of such codes and constructed an algebraic code matching this bound. Today, we will discuss two more notions of locality, those of *local correctability* and *local testability*. Local correctability is roughly a probabilistic analogue of local recoverability that allows errors instead of erasures in the codeword, while the goal of local testability is more broad; it seeks to efficiently determine whether a particular string is a codeword or not. In both cases, there is a parameter ℓ bounding the number of queries (i.e., enforcing locality), as well another parameter that governs the amount of allowed error. In each case, we will define the notion and discuss its application to two well-known families of codes, Hadamard codes and Reed-Muller codes. Finally, we will briefly outline the definition of *probabilistically checkable proofs (PCPs)* and how the above notions of local codes can help achieve it.

2 Local correctability

Note that a codeword c of a code $\mathcal{C} \subset \Sigma^N$ can be equivalently be thought of as a map $f : [N] \to \Sigma : i \mapsto c_i$. This is very fruitful for designing and analyzing algebraic codes; for instance, in the Reed-Solomon code, we identify $[N]$ with $\mathbb{F}_q^n$, and the code consists of low-degree polynomials. Today, we will use this view of codewords. For two functions $f, g : [N] \to \Sigma$, we can naturally define a measure of distance $\delta(f, g) := \Pr_{x \in [N]}[f(x) \neq g(x)]$.

Definition 1. *A code $\mathcal{C} \subset \{f : [N] \to \Sigma\}$ is (ℓ, ε)-locally-correctable if there is some (randomized) decoder D such that for all codeword functions $f \in \mathcal{C}$ and corrupted functions $g : [N] \to \Sigma$ such that $\delta(f, g) < \varepsilon$, for all $x \in [N]$, $D^g(x)$ makes ℓ queries to locations in g and outputs $f(x)$ with probability at least $\frac{1}{2}$.*

Here, the notation $D^g(x)$ means that D is given *oracle access* to g, i.e., it can query g's evaluations at arbitrary points in its domain. We will show that Hadamard and Reed-Muller codes are locally-correctable.

2.1 Local correctability of Hadamard codes

Recall that the *Hadamard code* $\mathcal{H}(n)$ is the set of all *linear* maps $f : \mathbb{F}_2^n \to \mathbb{F}_2$, i.e.,

$$f \in \mathcal{H}(n) \iff \forall x, y \in \mathbb{F}_2^n, \quad f(x) + f(y) = f(x + y).$$

Let's restate the decoding problem: Given oracle access to $g : \mathbb{F}_2^n \to \mathbb{F}_2$ such that there is some codeword $f \in \mathcal{H}(n)$ with $\delta(f, g) < \varepsilon$, and a location $x \in \mathbb{F}_2^n$, we want to compute $f(x)$. In other words, we have global but corrupted access to g, we want local but uncorrupted access to f, and we have a guarantee that f and g are ε-close. We will construct and analyze a simple local decoder.

Algorithm 1 Local decoder for Hadamard codes

Require: Oracle access to $g : \mathbb{F}_2^n \to \mathbb{F}_2$ and a desired location $x \in \mathbb{F}_2^n$

1: Randomly pick $y \sim \mathbb{F}_2^n$
2: **return** $g(x + y) - g(y)$

Analyzing this algorithm will yield the following theorem:

Theorem 2. *The Hadamard code $\mathcal{H}(n)$ is $(2, \frac{1}{4})$-locally-correctable.*

Proof. Let ε be arbitrary, and let D be the decoder in Algorithm 1. We claim that

$$\Pr_y[D^g(x) \neq f(x)] < 2\varepsilon,$$

yielding the desired result for $\varepsilon = \frac{1}{4}$.

Firstly, since $\delta(f, g) < \varepsilon$,

$$\Pr_y[g(y) \neq f(y)] < \varepsilon.$$

Furthermore, for *any* fixed x, if y is uniformly random, then $x + y$ is also uniformly random, so

$$\Pr_y[g(x + y) \neq f(x + y)] < \varepsilon.$$

So by the union bound,

$$\Pr_y[g(x + y) \neq f(x + y) \text{ or } g(y) \neq f(y)] < 2\varepsilon,$$

and so

$$\Pr_y[g(x + y) - g(y) = f(x + y) - f(y)] > 1 - 2\varepsilon.$$

But since $f \in \mathcal{H}(n)$, $f(x + y) - f(y) = f(x)$, yielding the desired result. □

A famous paper with numerous applications in not just coding theory but also complexity theory and cryptography is the paper of Goldreich and Levin [GL89] on local list-decoding of Hadamard codes using the notion of *hard-core predicates*.

2.2 Local correctability of Reed-Muller codes

The *Reed-Muller code* $\mathcal{RM}(q, r, m)$ is given by m-variate polynomials of degree at most r over $\mathbb{F}_q$, i.e.,

$$\mathcal{RM}(q, r, m) := \{f : \mathbb{F}_q^m \to \mathbb{F}_q \text{ s.t. } \deg(f) \leq r\}.$$

We will assume that $r < q - 1$, and $m < q$.

Given any $a, b \in \mathbb{F}_q^m$, we define the line $\ell_{a,b} := \{ta + b : t \in \mathbb{F}_q\}$; furthermore, for any m-variate polynomial f over $\mathbb{F}_q$, we can define $f|_{\ell_{a,b}}(t) = f(ta + b)$. Note also that if $f \in \mathcal{RM}(q, r, m)$, then $f|_{\ell_{a,b}} \in \mathcal{RM}(q, r, 1)$.

Exercise 3. *Show a partial converse to the above: If $r < q/2$, then $f \in \mathcal{RM}(q, r, m)$ if for all $a, b \in \mathbb{F}_q^m$, $f|_{\ell_{a,b}} \in \mathcal{RM}(q, r, 1)$.* Hint: *Use induction on m.*

We define an algorithm for locally decoding Reed-Muller codes:

Algorithm 2 Local decoder for Reed-Muller codes

Require: Oracle access to $g : \mathbb{F}_q^n \to \mathbb{F}$ and a desired location $x \in \mathbb{F}_q^n$. Distinct nonzero field elements $\alpha_1, \ldots, \alpha_{r+1}$.

1: Randomly pick $y \sim \mathbb{F}_q^n$
2: **for** i = 1 **to** r+1 **do**
3: $\quad \beta_i \leftarrow g(x + \alpha_i y)$
4: **end for**
5: Interpolate $h \in \mathcal{RM}(q, r, 1)$ such that $h(\alpha_i) = \beta_i$ for each i
6: **return** $h(0)$.

Theorem 4 ([BF90])**.** *For all $m < q$, $r < q - 1$, $\mathcal{RM}(q, r, m)$ is $(r + 1, O(\frac{1}{r+1}))$-locally-correctable.*

Proof. Suppose that there is some $f \in \mathcal{RM}(q, r, m)$ such that $\delta(f, g) < \varepsilon$, and let D be the decoder of Algorithm 2. Fix $x \in \mathbb{F}_q^m$ and ε.

Firstly, note that for uniformly random $y \in \mathbb{F}_q^n$ and fixed $i \in [r+1]$, since $\alpha_i \neq 0$, $x + \alpha_i y$ is marginally uniformly random, so

$$\Pr_y[f(x + \alpha_i y) \neq g(x + \alpha_i y)] < \varepsilon$$

since $\delta(f, g) < \varepsilon$. So union-bounding over i,

$$\Pr_y[\exists i \in [r+1] \text{ s.t. } f(x + \alpha_i y) \neq g(x + \alpha_i y)] < (r+1)\varepsilon.$$

Thus, with probability $1 - (r+1)\varepsilon$, $f|_{\ell_{x,y}}$ and h agree on $\alpha_1, \ldots, \alpha_{r+1}$. But they are both univariate polynomials of degree at most r agreeing on $r+1$ distinct points; hence, they are the same polynomial, and so $h(0) = f|_{\ell_{x,y}}(0) = f(x)$. So picking $\varepsilon < \frac{1}{2(r+1)}$ yields the desired result. □

Note that we required that $r < q - 1$ in order for there to be $r + 2$ distinct elements in the field (i.e., 0, as well as nonzero, distinct elements $\alpha_1, \ldots, \alpha_{r+1}$).

Exercise 5. *Show that if $r = o(q)$, $\mathcal{RM}(q, r, m)$ is $(O(r), \frac{1}{4} - o(1))$-locally-correctable.* Hint: *Instead of applying the union bound, apply a concentration inequality.*

We conclude by noting that in the proofs of both Theorems 2 and 4, we relied on the ability to restrict codeword functions to low-dimensional subspaces while preserving their degrees; in particular, in Theorem 2, we restricted to a two-dimensional linear subspace, whereas in Theorem 4, we restricted to a one-dimensional affine subspace.

Theorem 4 implies a theorem of Lipton [Lip91] called the *worst-case-to-average-case reduction for matrix permanents*; it gives a way to convert algorithms that can compute a random matrix's permanent with high probability into algorithms that can compute the permanent of any matrix. The idea is just to note that matrix permanents are multivariate polynomials!

3 Local testability

We give another definition of a local property we might want a code to have. In this case, the goal is not to actually locally decode specific bits, but to locally tell whether a function is a codeword at all. This is perhaps less practical but of great theoretical interest, as we will see. In what follows, we measure how close a function $g : [N] \to \Sigma$ is to being a codeword by $\delta(g, \mathcal{C}) := \min_{f \in \mathcal{C}} \delta(f, g)$.

Definition 6. *A code $\mathcal{C} \subset \{f : [N] \to \Sigma\}$ is (ℓ, α)-locally-testable if there is a tester T such that:*

1. *If $g \in \mathcal{C}$, $\Pr[T^g = 1] = 1$.*
2. *If $g \notin \mathcal{C}$, $\Pr[T^g = 0] \geq \alpha \cdot \delta(g, \mathcal{C})$.*
3. *T^g always makes at most ℓ queries to g.*

In other words, the farther g is from being a codeword, the more likely the tester T should be to reject it. We have the following theorems about local testability:

Theorem 7 ([BLR93, Theorem 1])**.** *The Hadamard code $\mathcal{H}(n)$ is $(3, \Omega(1))$-locally-testable.*

Theorem 8 ([RS96, Theorem 5])**.** *The Reed-Muller code $\mathcal{RM}(q, r, m)$ is $(r + 2, \frac{1}{r^2})$-locally-testable.*

Interestingly, Theorems 7 and 8 both require one more query than their counterparts we proved above for local correctability. Indeed, both testers have the following outline: To test a function g, pick a location x at random, and accept if $g(x) = D^g(x)$, where D is the appropriate local decoder. So if the decoder requires ℓ queries, the tester requires $\ell + 1$ queries. But unfortunately, this isn't known to work in general, and each

Algorithm 3 Local tester for Hadamard codes

Require: Oracle access to $g : \mathbb{F}_2^n \to \mathbb{F}$ and a desired location $x \in \mathbb{F}_2^n$.

1: Randomly pick $x, y \sim \mathbb{F}_2^n$.
2: **if** $g(x) + g(y) = g(x + y)$ **then**
3: **return** 1
4: **else**
5: **return** 0
6: **end if**

specific theorem is rather nontrivial to prove! We will only prove Theorem 7 below; the proof of Theorem 8 is similar but involves a messier analysis.

Intuitively, this theorem is affirmatively answering the question "Can we tell whether a function is linear via local tests?" The tester, Algorithm 3, is remarkably simple, and its analysis is as follows:

Proof of Theorem 7. We will show that the tester given in Algorithm 3 is a good local tester. Fix a function $g : \mathbb{F}_2^n \to \mathbb{F}_2$. Let $\varepsilon(g) := \Pr_{x,y}[g(x) + g(y) \neq g(x + y)]$ be T's probability of rejecting g. We wish to show that if g has "high" distance to any linear function, then $\varepsilon(g)$ is "high". We will construct a function f and show two statements: f and g are $O(\varepsilon(g))$-close, and if $\varepsilon(g) < \Omega(1)$, f is linear; with appropriate constants, these two statements will give us the bound that we want.

Specifically, let $D^g(x; y) = g(x + y) - g(y)$, and let

$$f(x) := \operatorname*{argmax}_r \left\{ \Pr_y[D_g(x; y) = r] \right\}.$$

For fixed $x \in \mathbb{F}_2^n$, we can think of each value $y \in \mathbb{F}_2^n$ as having one "vote" (i.e., $D^g(x; y)$) as to the true value of $g(x)$, and $f(x)$ reports a plurality opinion. Note that a priori, it is not even obvious that f might be linear. We will state some lemmas and prove them subsequently; in particular, Lemma 9 will give us the $O(\varepsilon(g))$-closeness of f and g, and Lemma 12 will give us the linearity of f for small $\varepsilon(g)$.

Lemma 9. $\delta(f, g) \leq 2\varepsilon(g)$.

Lemma 10. *For all x,*

$$\Pr_{y,z}[D^g(x; y) \neq D^g(x; z)] \leq 2\varepsilon(g).$$

Corollary 11. *For all x,*

$$\Pr_y[f(x) \neq D^g(x; y)] \leq 2\varepsilon(g).$$

This follows via a straightforward probability argument from Lemma 10. Intuitively, Lemma 10 says that two random opinions agree with high probability; this corollary says that a random opinion agrees with the most popular opinion with similarly high probability.

Lemma 12. *If $\varepsilon(g) < \frac{1}{8}$, then for all $x, y \in \mathbb{F}_2^n$, $f(x) + f(y) = f(x + y)$, i.e., $f \in \mathcal{H}(n)$.*

Together, the lemmas let us establish the theorem as follows. We will pick $\alpha = \frac{1}{8}$. Note that our test requires exactly three queries, and outputs 1 with probability 1 if $g \in \mathcal{H}(n)$. On the other hand, if $g \notin \mathcal{H}(n)$, we wish to show that $\varepsilon(g) \geq \frac{1}{8}\delta(f^*, g)$, where f^* minimizes $\delta(f^*, g)$. But note that if $\varepsilon(g) \geq \frac{1}{8}$, since $\delta(f^*, g) \leq 1$, the result follows immediately. Otherwise, we know that $f \in \mathcal{H}(n)$ by Lemma 12, so $\delta(f, g) \geq \delta(f^*, g)$; thus, since $\varepsilon(g) \geq 2\delta(f, g)$ by Lemma 9, $\varepsilon(g) \geq 2\delta(f^*, g) \geq \frac{1}{8}\delta(f^*, g)$, as desired.

Finally, the proofs of the lemmas (which were not given class, but were discussed in office hours):

Proof of Lemma 9. Firstly, fix $x \in \mathbb{F}_2^n$. Note that if $\Pr_y[g(x) + g(y) = g(x + y)] > \frac{1}{2}$ then $f(x) = g(x)$ (i.e., the plurality opinion is the correct one). Thus, if $f(x) \neq g(x)$, $\Pr_y[g(x) + g(y) \neq g(x + y)] > \frac{1}{2}$. So

$$\frac{1}{2}\delta(f, g) = \frac{1}{2}\Pr_x[f(x) \neq g(x)] < \Pr_{x,y}[g(x) + g(y) \neq g(x + y)] = \varepsilon(g),$$

as desired. □

Proof of Lemma 10. We will do a "proof by magical matrix". Let $x \in \mathbb{F}_2^n$ be fixed and $y, z \in \mathbb{F}_2^n$ be random, and consider the arrangement of numbers

$$\begin{array}{ccc} 0 & -g(x+y) & g(y) \\ -g(x+z) & g(x+y+z) & -g(y). \\ g(z) & -g(z) & 0 \end{array}$$

Let R_i denote the sum of the i-th row and C_j the sum of the j-th column. Observe that $R_1 = -D^g(x; y)$ while $C_1 = -D^g(x; z)$, so we want to show that $R_1 = C_1$. But we know that $R_1 + R_2 + R_3 = C_1 + C_2 + C_3$; thus, if $R_2 = R_3 = C_2 = C_3 = 0$, we can conclude that $R_1 = C_1$. Trivially, $R_3 = C_3 = 0$ for all choices of x, y, z. Thus, to prove the lemma suffices to show that with probability at least $1 - 2\varepsilon(g)$ over y, z, $R_2 = C_2 = 0$.

Note that $C_2 = D^g(x + y; z) - g(x + y)$. But x is fixed and y is uniform, so $x + y$ is marginally uniform; z is also uniform and independent of $x + y$, so by definition, $\varepsilon(g)$ upper-bounds the probability that $C_2 \neq 0$. It also upper-bounds the probability that $R_2 \neq 0$ by a symmetric argument. So by the union bound, the probability that either $R_2 \neq 0$ or $C_2 \neq 0$ is at most $2\varepsilon(g)$. □

Proof of Lemma 12. This proof will use yet another "magical matrix" as well as the probabilistic method. Let $x, y \in \mathbb{F}_2^n$ be fixed and $u, v \in \mathbb{F}_2^n$ be random, and consider the arrangement of numbers

$$\begin{array}{ccc} f(x) & -f(x+y) & f(y) \\ -g(x+u) & g(x+y+u+v) & -g(y+v). \\ g(u) & -g(u+v) & g(v) \end{array}$$

Again, let R_i denote the sum of the i-th row and C_j the sum of the j-th column, and we know that $R_1 + R_2 + R_3 = C_1 + C_2 + C_3$; this time, we will use the implication that if $R_2 = R_3 = C_1 = C_2 = C_3 = 0$, then $R_1 = 0$. To prove the lemma, we will show that, as long as $\varepsilon(g) < \frac{1}{8}$, R_2, R_3, C_1, C_2, C_3 all vanish with nonzero probability; this implies that $R_1 = 0$ with nonzero probability, but since R_1 is constant (i.e., not varying with u, v), we will be able to conclude unconditionally that $R_1 = 0$.

By Corollary 11, since x is fixed and u is uniform, $C_1 = 0$ with probability $1 - 2\varepsilon(g)$; this also holds for C_2 and C_3 (in the case of C_2, we observe that $x + y$ is fixed and $u + v$ is marginally uniform). Furthermore, $R_3 = 0$ with probability $1 - \varepsilon(g)$ by the definition of $\varepsilon(g)$ (since u and v are uniform and independent), and $R_2 = 0$ with probability $1 - \varepsilon(g)$ (since $x + u$ and $y + v$ are marginally uniform and independent). So by the union bound, with probability $1 - 8\varepsilon(g)$, R_2, R_3, C_1, C_2, C_3 all vanish, and $1 - 8\varepsilon(g)$ is nonzero as long as $\varepsilon(g) < \frac{1}{8}$, as desired. □

□

4 Local coding and probabilistically checkable proofs

Finally, we will briefly outline one of the first applications of all the work we've been doing to complexity theory: The local correctability and testability of the Reed-Muller codes gives a good example of a *probabilistically checkable proof (PCP)* for an NP-complete problem, which is roughly a proof that is locally checkable with high probability. More formally, fixing the problem as Graph 3-Coloring, we can define PCPs as follows:

Definition 13. *A ℓ-PCP system for* Graph 3-Coloring *is a polynomial-time verifier V accessing proofs $\pi \in \{0, 1\}^*$:*

1. *If a graph G is three-colorable, there exists some π such that $V^\pi(G) = 1$ with probability 1, with $|\pi| = poly(|G|)$.*

2. *If a graph G is not three-colorable, for all π, $\Pr[V^\pi(G) = 1] \leq \frac{1}{2}$.*

3. *$V^\pi(G)$ makes at most ℓ queries to π.*

Note how similar our definition is to Definition 6 for locally-testable codes! Ideally, we want ℓ to be constant (and the existence of such PCPs is the full *PCP theorem* of [Aro+98] — on which Madhu was a co-author!); but for our purposes, we're happy if ℓ is negligible in $|\pi|$ or negligible in G — even getting ℓ to be sublinear in the number of vertices is nontrivial.

We claim that we can can get a PCP requiring queries polylogarithmic in the graph and proof size. If we had more time, we could do this in two steps:

1. First, we want a polylogarithmic PCP for Reed-Solomon Proximity. Intuitively, we want to be able to efficiently and locally prove that a given function is close to a low-degree polynomial. That is, we want a PCP such that, for a function $g : \mathbb{F}_q \to \mathbb{F}_q$, if $\deg(g) \leq k$, there is a proof π such that

$$\Pr[V^{g,\pi}(k) = 1] = 1,$$

   whereas if $\delta(f, g) \geq 0.1$ for all f such that $\deg(f) \leq 2k$, then for all π,

$$\Pr[V^{g,\pi}(k) = 1] \leq \frac{1}{2}.$$

   And we want the number of required queries to g and π to be $\ell = \text{polylog}(q)$. This is an exercise on Problem Set 6; the idea is to embed high-degree univariate polynomials as low-degree multivariate polynomials (i.e., the embedded degree should be logarithmic in the original degree), and then apply Theorem 8.

2. Next, we could give a standard NP-reduction from Graph 3-Coloring to Reed-Solomon proximity.

Next time, we will see *efficient* optimal local decoding.

5 Solution sketches to exercises

Exercise 5: Replicate the analysis of Theorem 4, but evaluate on roughly $2r$ points instead, and consider roughly $1/4$ fraction of errors. (We only need roughly $r < q/2$.) We know that

$$\mathop{\mathbb{E}}_{y}[\#\{i : f(x + \alpha_i y) \neq g(x + \alpha_i y)\}] \lessapprox r/2.$$

Markov's inequality gives us that

$$\Pr_{y}[\#\{i : f(x + \alpha_i y) \neq g(x + \alpha_i y)\} > r] \lessapprox \frac{1}{2}.$$

Thus, we have $2r$ evaluations of an r-degree polynomial with roughly r errors, so we can use Reed-Solomon decoding to recover the polynomial.

References

[Aro+98] Sanjeev Arora et al. "Proof verification and the hardness of approximation problems". In: *Journal of the ACM* 45.3 (1998), pp. 501–555.

[BF90] Donald Beaver and Joan Feigenbaum. "Hiding instances in multioracle queries". In: *Proceedings of the 7th Annual Symposium on Theoretical Aspects of Computer Science*. Feb. 1990, pp. 37–48.

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. "Self-testing/correcting with applications to numerical problems". In: *Journal of Computer and System Sciences* 47.3 (Dec. 1993), pp. 549–595.

[GL89] Oded Goldreich and Leonid Levin. "A hard-core predicate for all one-way functions". In: *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*. Feb. 1989, pp. 25–32.

[Lip91] Richard Lipton. "New directions in testing". In: *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 2 (1991), pp. 191–202.

[RS96] Ronitt Rubinfeld and Madhu Sudan. "Robust characterizations of polynomials with application to program testing". In: *SIAM Journal of Computing* 25.2 (1996), pp. 252–271.