PCPs and HDX - Lecture 3

November 22, 2016

In the previous lecture we saw the linearity testing theorem. Today we will prove a generalization of the linearity testing to low degree testing. We will define locally testable codes. In these terms we have seen the local testability of the Hadamard code, and will today see the local testability of the Reed-Muller code.

1 Coding theory - quick glance

Coding theory began in the 1940s with works of Shannon and Hamming as a theory studying resilient communication. Today error correcting codes are used in abundance for communication, storage, as well as in many theoretical applications in TCS. We will see some of these in this course.

The idea is to encode a message of k bits using a codeword of n bits so that it is resilient to up to t bit flips. Namely, a code is a mapping $E : \{0,1\}^k \to \{0,1\}^n$ such that if $\mathrm{dist}(w, E(m)) \le t$ then m is the unique codeword whose distance is at most t to w.

It is good to think of the codewords as points in a geometric space $\{0,1\}^n$ where the distances are Hamming distances. A code is resilient to t bit flips if every point $w \in \{0,1\}^n$ has at most one codeword in a Hamming ball of radius t. In this way of thinking we don't focus on the encoding that maps messages to codewords, but rather only about the set $C \subset \{0,1\}^n$ of codewords.

Definition 1.1 (Distance). *The distance of a code $C \subset \{0,1\}^n$ is δ if for any pair of distinct codewords $w_1 \neq w_2 \in C$, $\mathrm{dist}(w_1, w_2) = \Pr_{i\in[n]}[w_1(i) \neq w_2(i)] \ge \delta$.*

How many codewords can we pack into a code with distance δ ? Consider the greedy construction: pick a first word $w_1 \in \{0,1\}^n$ arbitrarily, and remove all of the Hamming ball around w_1 of radius up to δn. Continue by picking another point w_2 arbitrarily from the remaining space, and remove the ball around w_2. Each ball has $\sum_{i=0}^{\delta n} \binom{n}{i} \approx 2^{H(\delta)n}$ points, so we can continue for at least $2^n/2^{nH(\delta)} = 2^{n(1-H(\delta))}$.

Definition 1.2 (Rate). *The rate of a code $C \subset \{0,1\}^n$ is $\frac{\log_2 |C|}{n}$.*

The gold standard of error correcting codes is having a code that has both constant rate and constant distance.

Examples of error correcting codes.

- Greedy - as we have seen above, it has very good rate and distance, but no effective algorithm for encoding nor decoding.
- Random linear code - choosing a random linear transformation turns out to give a very good code, with good rate and distance, very simple encoding (multiply by a matrix), but there is no known way to decode and this is actually believed to be a hard problem (used for crypto in fact).

- Hadamard - this is the code that maps a message $a \in \{0,1\}^k$ to the string $s = (\sum_i a_i x_i \mod 2)_{x \in \{0,1\}^k} \in \{0,1\}^{2^k}$. The codewords are the rows of the Hadamard matrix, hence the name. This code has distance 1/2 but the rate is pretty bad: $k/2^k$.

2 Low degree polynomials over a finite field - the Reed-Solomon and the Reed-Muller Codes

Fix a finite field $\mathbb{F}$, e.g. for a prime q it is the field whose elements are $\{0, 1, \ldots, q-1\}$ and addition and multiplication are done modulo q. All we need to know about $\mathbb{F}$ is that it supports addition with an inverse and multiplication with an inverse for all elements except 0.

Reed-Solomon Code. The codewords in the Reed-Solomon code are functions $f : \mathbb{F} \to \mathbb{F}$ that can be written as polynomial functions with degree at most d, i.e.

$$f(x) = \sum_{i=0}^{d} a_i x^i,$$

for some coefficients $a_i \in \mathbb{F}$. A codeword is a string $w \in \mathbb{F}^{|F|}$ whose t-th element is $f(t)$. This is sometimes called a points-evaluation table, and from now on we do not distinguish between the function f and the codeword representing it.

What is the distance of this code?

Fact 2.1. *If $f \neq f'$ are two degree d functions then* $\text{dist}(f, f') = \Pr_{z \in \mathbb{F}}[f(z) \neq f'(z)] \geq 1 - \frac{d}{q}$.

The reason for this is that $f - f'$ is a non-zero polynomial funcion, whose degree is at most d, so it can only be zero on at most d out of the q points.

What is the rate? it is $(d+1)/q$ because every set of $d+1$ coefficients define a valid degree d function.

One very important property of low degree polynomials in one variable is that given the value of f on $d+1$ points, it determines the value of f on all other points. Moreover, the value can be obtained from the $d+1$ values via a linear combination,

Fact 2.2 (Lagrange linear interpolation). *Let $x_1, \ldots, x_{d+1} \in \mathbb{F}$ be distinct and let $b_1, \ldots, b_{d+1} \in \mathbb{F}$ (not necessarily distinct). There is a unique polynomial $p(x)$ of degree at most d for which $p(x_i) = b_i$ for all $i = 1, \ldots, d+1$.*

Proof. The function $m_j(t) = \prod_{i \neq j} \frac{t - x_i}{x_j - x_i}$ has degree d in t (we think of $x_1, \ldots, x_{d+1}$ as coefficients, and division by $x_j - x_i$ should be interpreted as multiplication by $(x_j - x_i)^{-1} \in \mathbb{F}$) and also $m_j(x_i)$ equals 0 if $i \neq j$ and it equals 1 if $i = j$. Now take $p(t) = \sum_{j=1}^{d+1} b_j \cdot m_j(t)$. Uniqueness follows because distinct degree d polynomials cannot agree on $d+1$ points. $\square$

An important feature of the above is that the interpolation is *linear* in the values $b_1, \ldots, b_{d+1}$. The Reed Solomon code has very good (optimal, in fact) rate and distance. It also has very efficient encoding and decoding algorithms, and is used in practice. However, it is limited in that the length of a codeword is at most $|\mathbb{F}| = q$. If we want longer messages, we can move to multivariate polynomials, which is exactly the Reed-Muller code.

Reed-Muller Code. The codewords of this code are (points-evaluation-tables of) functions $f : \mathbb{F}^m \to \mathbb{F}$ that can be written as multivariate polynomial functions of degree at most d,

$$f(x_1, \dots, x_m) = \sum_S a_S x^S$$

where $a_S \in \mathbb{F}$ are coefficients, $S = (d_1, \dots, d_m)$ and x^S denotes $x_1^{d_1} x_2^{d_2} \cdots x_m^{d_m}$, and $a_S = 0$ whenever $\sum_i d_i > d$.

One can prove (this is by easy induction on m) that the distance here too is $1 - \frac{d}{q}$. What is the rate? The number of distinct polynomials can be counted by looking at how many coefficients it takes to define a polynomial whose degree is at most d. There is a monomial $x_1^{d_1} x_2^{d_2} \cdots x_m^{d_m}$ as long as $\sum_i d_i \leq d$. The number of such tuples $(d_1, \dots, d_m)$ is $\binom{m+d}{d}$. For smallish m this is on the order of d^m (think of $m = 2, 3$ for example). The length of the codewords is q^m, so the relative rate is roughly $(d/q)^m$.

One very important feature of low degree multi-variate polynomials is that their restriction to any affine line is a univariate function *of degree d.*

Affine lines. A line in $\mathbb{F}^m$ is defined by two distinct points $x \neq y \in \mathbb{F}^m$,

$$\ell_{x,y} = \{x + t(y - x) \mid t \in \mathbb{F}\} \subset \mathbb{F}^m$$

Every line has exactly $|\mathbb{F}|$ points, and there is a unique line passing between every pair of distinct points. We can also think of the line s a function parameterized by t, $\ell_{x,y}(t) = x + t(y - x)$.

Fact 2.3. *Let $f : \mathbb{F}^m \to \mathbb{F}$ be a degree d function, and let $\ell_{x,y}$ be an arbitrary line. Then the restriction of f to the line is a univariate function $f \circ \ell_{x,y} : \mathbb{F} \to \mathbb{F}$ whose degree is at most d.*

This simple fact is very important : it means that even though f is defined on a huge space of $\mathbb{F}^m$, there are many collections of $|\mathbb{F}|$ points (all lines), where there are non-trivial relations between the values of *every* codeword of the RM code. Namely, every low degree function must look low degree even on tiny windows that are the affine lines!

3 Locally Testable Codes

A Locally Testable Code (LTC) is an error correcting code that is also locally testable. Let us define this in terms of constraint systems.

Definition 3.1 (LTC). *A constraint system $(V, \mathbf{C}, \Sigma)$ defines a code whose codewords are*

$$C := \mathrm{SAT}(\mathbf{C}) = \{\mathrm{f} : \mathrm{V} \to \Sigma \mid \mathrm{f} \textit{ satisfies every constraint in } \mathbf{C}\}$$

We say that this code is an LTC with ℓ queries and strong soundness $\gamma > 0$ if the constraints in $\mathbf{C}$ are at most ℓ-local, and if the constraint system has expansion $\gamma(\mathbf{C}) \geq \gamma$. Explicitly,

- *(Completeness:) If $f : V \to \Sigma$ satisfies all of the constraints in $\mathbf{C}$ then it is in the code.*
- *(Strong soundness:) If $f : V \to \Sigma$ has distance at least δ from $\mathrm{SAT}(\mathbf{C})$, then at least $\gamma \cdot \delta$ fraction of the constraints in $\mathbf{C}$ reject.*

The second item is called in the literature strong soundness, or strong robust soundness. There is a weaker notion of soundness that we will not discuss here.

We have seen in the previous lecture that the BLR constraint system $(\{0,1\}^n, \mathbf{BLR}_n, \{0,1\})$ is an expanding constraint system, and that $\mathrm{SAT}(\mathbf{BLR}_\mathrm{n})$ is exactly the set of functions $f : \{0,1\}^n \to \{0,1\}$ that are linear, i.e. the codewords of the Hadamard code. This means that

Theorem 3.2 ([BLR90]). *The Hadamard code is locally testable with* 3 *queries and strong soundness* $\geq 2/9$.

We will begin to prove today that

Theorem 3.3 ([RS92]). *The RM code is locally testable with* $d+2$ *queries and strong soundness* $1/poly(d)$.

There are a number of testing results that are all called "low degree tests". We will prove one of them, due to Rubinfeld and Sudan. We will later prove stronger versions, in particular implying robust soundness which is a constant independent of d. The key is to use Fact 2.3 in order to define a system of constraints.

Let us define the test used by Rubinfeld and Sudan [RS92]. We use the language of constraint systems which is slightly more cumbersome, and we will soon move to talking about randomized tests. So let $(\mathbb{F}^m, \mathbf{LDT}_{d,m}, \mathbb{F})$ be a system of constraints with a constraint for every choice of $d+2$ points that reside on a single line. The constraint reads $f(z_0), \ldots, f(z_{d+1})$ and is satisfied iff there exists a degree d univariate polynomial $p : \mathbb{F} \to \mathbb{F}$ for which $p(z_i) = f(z_i)$ for every $i = 0, 1, \ldots, d+1$.

It is nice to think of this constraint system pictorially and geometrically, where we place vertices for each of the points in $\mathbb{F}^m$, and then for every $d+2$ distinct points on a line, we place a hyperedge together with a predicate (this is the constraint). One sees that on each line there are all possible $\binom{q}{d+2}$ hyperedges.

The following claim is essentially true because of interpolation,

Claim 3.4. *The code* $\mathrm{SAT}(\mathbf{LDT}_{\mathrm{d,m}})$ *is exactly the Reed-Muller code with dimension* m *and degree* d*, as long as* $d \leq q/2$ *(if* $\mathbb{F}$ *is a prime field then* $d+2 < |\mathbb{F}|$ *is sufficient, see [FS13]).*

It is interesting that when $\mathbb{F}$ is a non-prime field then the above claim fails when the degree is larger than $q/2$. An example is provided in a 1995 paper of Friedl and Sudan [FS13]. Later on a new family of codes called lifted codes were introduced in [GKS13], and shown to be a superset of the Reed-Muller code for certain degree parameters and certain fields.

References

[BLR90] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proc. 22nd ACM Symp. on Theory of Computing*, pages 73–83, 1990.

[FS13] Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. *CoRR*, abs/1307.3975, 2013.

[GKS13] Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 529–540, 2013.

[RS92] Ronitt Rubinfeld and Madhu Sudan. Self-testing polynomial functions efficiently and over rational domains. In *Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 27-29 January 1992, Orlando, Florida.*, pages 23–32, 1992.