Today

- Local decoding of Reed-Muller codes.
- Local list-decodability.

Recall Problem

- Given:
  - Oracle access to $r : \mathbb{F}^m \to \mathbb{F}$.
  - Point of interest: $x \in \mathbb{F}^m$.
  - Promise: $\exists p : \mathbb{F}^m \to \mathbb{F}$ of degree D s.t. $\Delta(r, p) = \Pr_{y \in \mathbb{F}^m}[r(y) \neq p(y)] \leq \delta$.
- Goal: Compute $p(x)$ with probability $> \frac{1}{2}$.
- Desired runtime: $\mathrm{poly}(m, D, \log q)$. Can even tolerate $\mathrm{poly}(q)$,where $q = |\mathbb{F}|$.

Basic idea

- Restrict r/p to some line L.
  - For $a, b \in \mathbb{F}^m, t \in \mathbb{F}$, let $L_{a,b}(t) = a + t \cdot b$.
  - $L_{a,b} = \{L_{a,b}(t) | t \in \mathbb{F}\}$.
- Line is a function $L_{a,b} : \mathbb{F} \to \mathbb{F}^m$.
- $f : \mathbb{F}^m \to \mathbb{F}$ restricted to line L is just the composed function $f|_L : \mathbb{F} \to \mathbb{F}$, with $f|_L(t) = f(L(t))$.

Lines in $\mathbb{F}^m$

- Algebraic Property: Low-degree poly restricted to subspace is a low-degree polynomial.

  $$\deg(f) \leq D \Rightarrow \deg(f|_L) \leq D.$$

- Randomness Property: Random line is a collection of pairwise independent points.

  $$\forall t \neq s, \Pr_{a,b}[L_{a,b}(t) = c \text{ and } L_{a,b}(s) = d] = 1/q^{2m}.$$

  Random line through a is $L_{a,b}$ with b being random. Random line through a is 1-wise random, except at $t = 0$.

  $$\forall t \neq 0, \Pr_b[L_{a,b}(t) = c] = 1/q^m.$$

Decoding Algorithm

- Fix $\alpha_1, \ldots, \alpha_{D+1} \in \mathbb{F}$ non-zero and distinct.
- Pick $y \in \mathbb{F}^m$ at random.
- Let $\beta_i = r(x + \alpha_i y)$.
- Compute univ. degree D poly $h(t)$ s.t. $h(\alpha_i) = \beta_i$.
- Output $h(0)$.

Analysis

- Hope for every query Q that $r(Q) = p(Q)$.
- Bad event $E_i : p(L_{x,y}(\alpha_i)) \neq r(L_{x,y}(\alpha_i))$.
- Claim 1: $\Pr_y[\exists i \text{ s.t. } E_i] \leq (D+1)\delta$.

  $\Pr_y[E_i] = \Delta(r, p) \leq \delta +$ Union bound.
- Claim 2: $\forall i \overline{E}_i \Rightarrow$ Algorithm correct.
  - For all $i \in [D+1]$, $p|_L(\alpha_i) = h(\alpha_i)$.
  - But $p|_L, h$ of degree D.
  - So $p|_L = h$ and $h(0) = p|_L(0) = p(x + 0y) = p(x)$.

Conclude: RM code with parameters $m, D, \mathbb{F}$ is $D+1$-locally decodable for $\delta < 1/(2(D+1))$ with $\mathrm{poly}(m, D)$ field operations.

Some range of parameters

- If $D = \log^c k$ and $m = \Omega(\log k/((c-1)\log\log k))$, then # coefficients $= k$.
- Pick field size $= 2D$ to get encoding size $n = (2D)^m = k^{c/(c-1)}$ (= poly rate).
- Get D-local decodability $= \mathrm{poly}\log n$.
- Pretty good. Almost best known.
- Error-tolerance not so good. Will do better next time.

Improving error-correction

- Idea 1:
  - Sample more points $\alpha_i, i \in [10D]$ from L.
  - Now get $\beta_i, i \in [10D]$. Find h of degree D agreeing with many pairs α_i, β_i (just RS decoding!) and output $h(0)$.
  - Analysis: Use Markov's inequality to bound too many errors.
  - Can get error close to $\frac{1}{4}$.
- More sophisticated algorithm + analysis corrects error close to $\frac{1}{2}$.

List-decoding?

- What is implicit list-decoding?
  - Main issue: First think about list-decoding; then about implicit representation of the output.
  - Technically easier to do it the other way, but that may be pointless.
  - Specifically, if $p_1, \ldots, p_c$ are the nearby polynomials, then easier to come up with an algorithm that produces $\{p_1(x), \ldots, p_c(x)\}$. But how do you produce an algorithm that only outputs, say, $p_1(x)$?
  - How does the algorithm distinguish p_1 from the rest?
  - Solution: Give it some advice (non-

uniform) to allow it to distinguish p_1 from the rest.
  - Example $p_1(z) = \gamma$.

Implicit "List-Decoding" Algorithm

- Given: Oracle r, Advice z, γ, input x.
- Algorithm:
  - Let $L = L_{x,z-x}$, so $L(0) = x, L(1) = z$.
  - Compute a list of all polynomials $h_1, \ldots, h_c$ of deg. D s.t. $h_i(\alpha) = r(\alpha)$ for $\delta/2$ fraction of $j \in \mathbb{F}$'s.
  - If $\exists$ unique i s.t. $h_i(1) = \gamma$, then output $h_i(0)$, else "BLAH".

Analysis

- No randomness? !
- Can't do it - right? Right!
- Will only show correct for
  - Random z.
  - Random x.
  - W.h.p. assuming $p_1(z) = \gamma$.

Analysis (contd.)

- Bad events:
  - A : (x, z) s.t. $p(L(\alpha)) = r(L(\alpha))$ for less than $\epsilon/2$ fraction of $\alpha \in \mathbb{F}$.
  - B : z s.t. some $h_j! = p|_L$ satisfies $h_j(1) = p|_L(1)$.
  - $\Pr[A]$ bounded by Chebychev.
  - $\Pr[B]$ more subtle. Think of L being picked first, and z later. Then $\Pr_{z|L}[B] \leq cD/q$.

- If neither A nor B occur, then printer outputs correct response.