Today

Conclude: [Parvaresh - Varsham]

[Guruswami + Rudra]

Rate-optimal list-decoding over large alphabets.

Review of last lecture

Parvaresh-Varsham Codes: Correlated RS Codes.

Message: Degree $< k$, poly $P_i$ over $\mathbb{F}_2$

Alphabet: $\mathbb{F}_2^*$

Encoding: let $P_2 = P_0 \ (mod \ h(x))$

Transmit $\sum_{i=1}^{n} (P_i(x_i), P_2(x_i))$
**Code Spec:** \( F_q, \alpha, \ldots, \alpha_n, k \), \( D > \left( \frac{n}{k} \right)^{\frac{3}{2}}, h(x), \text{monic, \hspace{1cm} New \ Stuff} \) \hspace{1cm} \text{rs stuff}

**New Stuff**

\( \text{weights, deg. } k \)

**Decoding Problem:**

**Given:** Code Spec + \( \mathcal{Z}_{i=1}^n (\alpha_i, \beta_i, \gamma_i) \)

**Find:** All poly \( p_1 \) s.t.

\[ \left| \left\{ i \mid p_1(\alpha_i) = \beta_i \land p_2(\alpha_i) = \gamma_i \right\} \right| > t \]

for \( p_2 = p_1^d \mod h(x) \)
Algorithm

Step 1: find \( Q(x, y, z) \neq 0 \) s.t.

- \( Q(ax, bx, cx) = 0 \) \( \forall i \)
- \( \deg_x Q \leq r^{2/3} n^{1/3} \)
- \( \deg_y Q, \deg_z Q \leq \left( \frac{n}{r} \right)^{1/3} \)

Step 1.5 - while \( h(x) \) divides \( Q(x, y, z) \)

\[ Q \leftarrow Q / h(x) \]

Step 2: let \( Q_x(y^2) = Q(x, y, z) \mod h(x) \)

let \( P_x(y) = Q_x(y, y^2) \)

Output all roots of \( P_x \) in \( \mathbb{F}_q[x] / h(x) \)
Analysis: (Won't repeat):

- $P_i$ has agreement $t > 3^{2/3} n^{1/3}$
  $\Rightarrow \ P_i$ root of $P_x(y)$.

- $P_x \neq 0$, deg $P_x$ not too large

Conclude:

- get code of rate $R = \frac{k}{2n}$

- corrects $\frac{n - 3^{2/3} n^{1/3}}{n} = 1 - O(R^{2/3})$ fraction errors.

- Beats $1 - \sqrt{R}$ as $R \to 0$
Two Improvements

1. Multiplicities: Can use multiplicities trick to get rid of the \(3\) in \(3 \cdot 2^{3n} n^{1/2}\)

More precise result:

code of rate \( R = \frac{R'}{2} \)

correcting \( 1 - (R')^{2/3} \) errors

\[ = 1 - (2R)^{2/3} \] errors.

2. \(m\)-correlated Polynomials

- Can have \( P_1, \ldots, P_m \)

\[ P_{in} = P_i^D \pmod{h(x)} \]
Get code of rate $R$

Correcting $1 - (mR)^{m/n}$ fraction errors

Converges to $1 - O(R \log \frac{1}{R})$ fraction errors.

Great for small rate!

Big rate = ?

[Elurusan - Rudra] Get rid of $m$ in $1 - (mR)^{m/n}$ by Algebraic Magic.
Idea: (Back in 2 polynomial setting)

- We're losing factor 2 in rate by being $P_1(x)$, then $P_2(x)$.

Could recover this if $P_2(x) = P_1(-x)$ for instance!

Then we'd be transmitting

But this is same as
Where $i^m$ symbol has same info as $(\frac{n}{2} + i)^m$ symbol.

- Don't need to send second half!
- Recover 2 in rate 1!!
- Can we implement our wish

$$P'_2(x) = P_1(x) \overset{D}{\equiv} (\mod h(x)) = P_1(-x) \overset{(\mod h(x))}{\equiv}$$

for every $P_i$?

- Parameters under our control:

$$h(x), \quad D : \text{can choose them as we please.}$$
Some Basic Magic

Issue: How can we guarantee some identity of the form

\[ P_i(x)^D = P_i(-x) \] for all \( P_i \)?

Insight: Using \( D = q \) simplifies the question...

\[ P_i(x)^D = \left( \sum c_i x^i \right)^D \]

\[ = \sum c_i^D x^{iD} \]

\[ = \sum c_i x^{iD} \]

\[ = P_i(x^D) \]
So suffice to have
\[ x^D = -x \pmod{h(x)} \]
\[ \implies \text{need } h(x) \mid x^D + x \]

**Bad News**

Unfortunately ... this ties our hands.

\( x^t \times x \) is fixed. Can't have an irreducible factor for most \( x \).

**[GRJ] Recovery:**

1. Let consider \( x^q \eta x \) for some \( \eta \neq -1 \).

2. Don't have to insist \( h(x) \) of degree \( k \); Magic takes care of degrees.
**Fact:** Let $\alpha \in \mathbb{F}_q^*$ be primitive.

Then $x^{q-1} - \alpha$ is irreducible.

Furthermore, for $p_i$,

$$p_i(x)^q \equiv p_i(\alpha x) \pmod{x^{q-1} - \alpha}$$

But we no longer have

$$\{p_1(x), p_2(x)^q\} = \{p_1(\alpha x), p_2(\alpha x)^q\}$$

**Fix:** Imperfect covers ....
GR code:

**Sieve:** \( F_2, k, \alpha, (m = 2), C = 3 \)

**Message:** \( p_i \) of \( \deg < k \)

**Encoding:**

\( p_1(\alpha), p_1(\alpha^2), \ldots, p_i(\alpha^c) \)

**Decoding**

\( y_1, \ldots, y_c \)

\( Y_0, \ldots, Y_m \)

\( Y_0, \ldots, Y_{m+1} \)

\( Y_{c-m+1}, \ldots, y_c \)

**PV decode this with \( D = 2 \), \( h(x) = x^{2^m} - \alpha \)**
Performance

Rate \( R = \frac{k}{2^{-1}} \)

Error correction: suppose \( t \) out of \( 2^{-1} \) agreements

- translate to \( (C - m + 1)t = t' \) out of \( n' \)

\( n' = (C - m + 1) \left( \frac{2^{-1}}{2} \right) \) symbols of PV code in agreement.

- PV decoder needs

\[ t' > n' \left( \frac{k}{n'} \right)^{\frac{m}{m+1}} \] agreements
\[ R = \frac{R}{q-1} \quad q-1 = \frac{c \cdot n'}{c-m+1} \]

\[ \Rightarrow \frac{R}{n'} = (q-1) \cdot R = \frac{c \cdot R}{c-m+1} \]

[GR] Code is decodable from

\[ \left( \left( \frac{c}{c-m+1} \right) \cdot R \right)^{\frac{m}{m-1}} \] fraction agreements

Let \( m = \frac{1}{c} \) and \( c = \frac{1}{c^2} \)

Then \( \text{error} = 1 - (1+\varepsilon) \cdot R^{\frac{1-\varepsilon}{c}} \)

\[ \Rightarrow 1 - R - f(\varepsilon) \]
Conclusions:

- Have found essentially best possible code + decoder over large alphabets.

- Drawbacks: Runtime = \( \text{poly}(n^{1/4}) \) ... not so nii. Can we do better?

- Questions: - Can RC codes be decoded better?
  - What other forms of folding will work well?