Outline

- Decoding Reed-Muller codes
- Turning worst-case hardness into average-case hardness
- Extractors
- Trevisan's extractor
- RL and undirected STCONN

Decoding RM

One key property:

If \( p \) has total degree at most \( h \), then \( p_{\parallel} \) has degree at most \( h \)

advantage: \( p_{\parallel} \) is a univariate polynomial (not multivariate)

Example:
- \( p(x_1, x_2) = x_1^2x_2 + x_2^2 \)
- \( L_1(z) = z + 1 \)
- \( L_2(z) = z \)
- \( p_{\parallel}(z) = (z+1)^2z + z^2 = 2z^3 + 2z^2 + z \)

Decoding RM

Second key property:

If pick \( a, b \) randomly in \( (F_q) \) then points in the vector 

\( (az + b) \) \( z \in F_q \)

are pairwise independent

Meaning of pairwise independent:

for all \( w, z \in F_q, u, v \in (F_q) \)

\( \Pr_{a,b}[L(w) = u \mid L(z) = v] = 1/q^t \)

every pair of points on \( L \) behaves just as if it was picked independently
Decoding RM

- Use random lines
- Given received word $R:(F_q)^t \rightarrow F_q$

- $\frac{\text{fraction of errors } \Delta(C(m), R)}{\text{total deg. } h} < \delta$
- $C(m) = p(x)$ is poly w/ total deg. $h$
- decode one symbol $b \in (F_q)^t$:
  - pick a randomly in $(F_q)^t$
  - $q$ pairs $(z, R(az + b))$ for $z \in F_q$
  - each point $az + b$ random in $(F_q)^t$
  - $E[\text{# errors hit}] < \delta q$
  - $\Pr[\text{# errors hit } > 4\delta q] < \frac{1}{t}$
  - try: find degree $h$ univariate poly $r$
    # $z$ for which $r(z) = R(az + b) \leq 4\delta q$

5/8/2003  CS 151 Lecture 10  5

Local decodability

- Amazing property of decoding method:
  $R: \begin{array}{cccccccc}
  0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
  \end{array}$
  $C(m): \begin{array}{cccccccc}
  \end{array}$

- Local decodability: each symbol of $C(m)$ decoded by looking at small number of symbols in $R$
  - small decoding circuit $D$
  - small circuit computing $R$
  - implies small circuit computing $C(m)$

5/8/2003  CS 151 Lecture 10  7

Concatenation

- Problem: symbols of $F_q$ rather than bits
- Solution: encode each symbol with binary code
  - our choice: RM with degree $h \leq 1$, # variables $t = \log q$
  - Schwartz-Zippel: distance = $\frac{1}{h}$
  $C(m): \begin{array}{cccccccc}
  5 & 2 & 7 & 1 & 2 & 9 & 0 & 3 \\
  \end{array}$
  $R: \begin{array}{cccccccc}
  0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
  \end{array}$

- decoding:
  - whenever would have accessed symbol $i$ of received word, decode binary code first, then proceed

5/8/2003  CS 151 Lecture 10  8
Setting parameters

- Recall we want: if $f'$ is $s'$-approximable, then
  \[ f: (0,1)^{\log k} \rightarrow \{0,1\} \]
  is computable by a size $s = \text{poly}(s')$ circuit
  - view (truth table of) $f'$ as encoding of (truth table of) $f$
  - polynomial blow-up: $f \in E \Rightarrow f' \in E$

- outer code: RM with parameters
  - field size $q = \log^2 k$
  - degree $h = \log^2 k$
  - dimension $t = \log k/(\log \log k)$

- inner code: RM with parameters
  - field size $q' = 2$
  - degree $h' = 1$
  - dimension $t' = \log q$

- Verify:
  - # outer coefficients $(h+t \text{ choose } t) > k$
  - block length $n = q'q = \text{poly}(k)$

- Conclude: $f \in E \Rightarrow f' \in E$

Decoding

- suppose $f'$ is $s'$-approximable
  - circuit of size $s'$ computes received word with agreement $\frac{1}{2} + 1/s'$
  - at least $s'/2$ "inner" recvd' words have agreement $\frac{1}{2} + 1/(2s')$
  - Johnson Bound: at most $O(s^2)$ inner codewords with this agreement
  - find by brute force: time $= q$
  - pick random one from list for each symbol
  - result is "outer" recvd word with agreement $1/s'^3$

- $f: (0,1)^{\log k} \rightarrow \{0,1\}$
  \[
  \begin{bmatrix}
  0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
  \end{bmatrix}
  \]
  \[
  f': (0,1)^{\log n} \rightarrow \{0,1\}
  \]
  \[
  \begin{bmatrix}
  0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
  \end{bmatrix}
  \]

- "outer" recvd word with agreement $1/s'^3$
  - can only uniquely decode from $< \frac{1}{2}$
    agreement (since rel. distance $< 1$)
  - our agreement $< \frac{1}{2}$
  - analog of Johnson Bound for q-ary codes: small number of codewords with agreement $1/s'^3$
  - can efficiently find this list!
  - same decoding strategy: reduce to RS list-decoding
  - list-decoding of RS: homework prob.
Decoding

- Final result: short list of circuits

\[ D_1 \xrightarrow{i} R \xrightarrow{(w_1)} \]

\[ D_2 \xrightarrow{i} R \xrightarrow{(w_2)} \]

\[ D_3 \xrightarrow{i} R \xrightarrow{(w_3)} \]

- Size: \( \text{poly}(q) \cdot \text{poly}(s') = \text{poly}(\log k, s') \)
- One computes \( f \) exactly!
- Conclude: if \( f' \) is \( s' \)-approximable, then \( f \) is computable by a size \( s = \text{poly}(s') \) circuit

Putting it all together

- Theorem (IW, STV): If \( E \) contains functions that require size \( 2^{\Omega(n)} \) circuits, then \( E \) contains \( 2^{\Omega(n)} \)-unapproximable functions.
  - Proof: let \( f = (f_i) \) be such a function that requires size \( s = 2^{2n} \) circuits
  - Define \( f' = (f_i^*) \) be just-described encoding of (truth table of) \( f \)
  - Just showed: if \( f' \) is \( s' = 2^{3n} \)-approximable, then \( f \) is computable by size \( s = \text{poly}(s') = 2^{kn} \) circuit, contradiction.
- Theorem (NW): if \( E \) contains \( 2^{\Omega(n)} \)-unapproximable functions then \( \text{BPP} = \text{P} \).
- Theorem (IW): \( E \) requires exponential circuits \( \Rightarrow \text{BPP} = \text{P} \).

Extractors

- PRGs: can remove randomness from algorithms
  - Based on unproven assumption
  - Polynomial slow-down
  - Not applicable in other settings
- Question: can we use "real" randomness?
  - Physical source
  - Imperfect - biased, correlated
- "Hardware" side
  - What physical source? (ask physicists...)
- "Software" side
  - What is the minimum we need from the physical source?

Extractors

- Imperfect sources:
  - "Stuck bits": \( \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array} \)
  - "Correlation": \( \begin{array}{cccc} * & * & * & * \end{array} \)
  - "More insidious correlation": \( \begin{array}{cccc} \text{perfect squares} \end{array} \)
- Specific ways to get independent unbiased random bits
- Assume don't know details of physical source
- General model capturing all of these? (yes: min-entropy)
- Universal procedure for all imperfect sources? (yes: extractors)
### Min-entropy

- General model of physical source with \( k < n \) bits of hidden randomness

- Definition: random variable \( X \) on \( \{0,1\}^n \) has **min-entropy**
  \[
  \min_x -\log(\Pr[X = x])
  \]
- min-entropy \( k \) implies no string has weight more than \( 2^{-k} \)
- convex combination of sources in picture

---

### Extractor

- Extractor: universal procedure for “purifying” imperfect source

- \( E \) is efficiently computable
- truly random seed as catalyst
- output fools all circuits \( C \):
  \[
  |\Pr_Z[C(z) = 1] - \Pr_{x,y}(C(E(x, y)) = 1)| \leq \epsilon
  \]
- \( E(X, U_1), U_m \) \( \epsilon \)-close \( (L_1 \text{ dist} \leq 2\epsilon) \)

---

### Extractors

- Notice similarity to PRGs
  - output of PRG fools all efficient tests
  - Output of extractor fools all tests

- Using extractors
  - use output in place of randomness in any application
  - alters probability of any outcome by at most \( \epsilon \)

- Main motivation:
  - use output in place of randomness in algorithm
  - how to get truly random seed?
  - enumerate all seeds, take majority

- Goals: good: best:
  - short seed \( O(\log n) \) \( \log n + O(1) \)
  - long output \( m = k^{(1)} \) \( m = k + O(1) \)
  - many \( k \)’s \( k = n^{(1)} \) any \( k = k(n) \)
- random function for \( E \) achieves best!
  - but we need explicit constructions
  - usually complex + technical
  - optimal extractors still open
Trevisan extractor

- Insight: use NW generator with source string in place of hard fn.
  - this works (!)
  - proof slightly different, easier

- Ingredients:
  - error-correcting code
    \[ C: \{0,1\}^n \rightarrow \{0,1\}^t \]
  - distance \(d = \frac{1}{2}m \cdot n'\), \(n' = \text{poly}(n)\)
  - \( (\log n, \alpha = \delta \log n/3) \) design:
    \[ S_1, S_2, \ldots, S_m \subseteq \{1, \ldots, t = O(\log n)\} \]

\[ E(x, y) = C(x)[y_{|S_1}] \cdot C(x)[y_{|S_2}] \cdot \ldots \cdot C(x)[y_{|S_m}] \]

\[ C(x) = \begin{bmatrix} 01010010111101011011001010 \end{bmatrix} \]

\[ \text{seed y} \]

- Theorem (T): \( E \) is an extractor for min-entropy \( k = n^3 \), with
  - output length \( m = k^{1/3} \)
  - seed length \( t = O(\log n) \)
  - error \( \epsilon \leq 1/m \)

- Proof (assumed \( X \subseteq \{0,1\}^n \))
  - assume fails to \( \epsilon \)-pass statistic test \( C \)
  - \( |Pr_{z}[C(z) = 1]; \text{Pr}\_{x,y}[C(E(x, y))] = 1]| > \epsilon \)
    - distinguisher \( C \Rightarrow \text{predictor} P \):
      \[ \text{Pr}_{x,y}[P(E(x, y)) = E(x, y)] > \frac{1}{3} + \epsilon/(2m) \]

Trevisan Extractor

\[ E(x, y) = C(x)[y_{|S_1}] \cdot C(x)[y_{|S_2}] \cdot \ldots \cdot C(x)[y_{|S_m}] \]

\[ C(x) = \begin{bmatrix} 01010010111101011011001010 \end{bmatrix} \]

\[ \text{seed y} \]

- Proof (continued):
  - for at least \( \epsilon/2 \) of \( x \in X \) we have:
    \[ Pr_{y}[P(E(x, y)_{1 \ldots t}) = E(x, y)] > \frac{1}{3} + \epsilon/(2m) \]
  - fix bits outside of \( S_i \), to preserve advantage
    \[ Pr_{y}[P(E(x; w, y_{1 \ldots t}) = C(x)[y_{1 \ldots t}] > \frac{1}{3} + \epsilon/m \]
  - as vary \( y' \), for \( j \neq i \) \( j \)-th bit varies over only \( 2^t \) values
    - build up to (m-1) tables of \( 2^t \) values to supply \( E(x; w, y')_{1 \ldots t-1} \)

Trevisan Extractor

\[ E(x, y) = C(x)[y_{|S_1}] \cdot C(x)[y_{|S_2}] \cdot \ldots \cdot C(x)[y_{|S_m}] \]

\[ C(x) = \begin{bmatrix} 01010010111101011011001010 \end{bmatrix} \]

\[ \text{seed y} \]

- Proof (continued):
  - (m-1) tables of size \( 2^t \) constitute a description of a string that has
    \[ \frac{1}{3} + \epsilon/(2m) \] agreement with \( C(x) \)
  - \( \# x \) with such a description?
    - \[ \exp((m-1)2^n) = \exp(k^{2/3}) \] strings
    - Johnson Bound: each string accounts for at most \( O(m^3) x \)'s
    - total \( \# : O(m^3) \exp(k^{2/3}) \ll 2^k(\epsilon/2) \)
    - contradiction.
**Strong error reduction**

- **L ∈ BPP if there is a p.p.t. TM M:**
  - \( x \in L \Rightarrow \Pr[M(x,y) \text{ accepts}] \geq 2/3 \)
  - \( x \notin L \Rightarrow \Pr[M(x,y) \text{ rejects}] \geq 2/3 \)

- **Want:**
  \( x \in L \Rightarrow \Pr[M(x,y) \text{ accepts}] \geq 1 - 2^{-k} \)
  \( x \notin L \Rightarrow \Pr[M(x,y) \text{ rejects}] \geq 1 - 2^{-k} \)

- **We saw:** repeat \( O(k) \) times
  - \( n = O(k) \cdot |y| \) random bits; \( 2^{-k} \) bad

- **Better:**
  - \( E \text{ ext. for } k = |y|^3 \cdot n^2, \varepsilon < 1/6 \)
  - \( \text{pick } w \in \{0,1\}^n \), run \( M(x, E(w, z)) \) for all \( z \in \{0,1\}^n \), take majority
  - \( w \) "bad" if \( \text{maj}_M(x, E(w, z)) \) wrong
  \( |\Pr[M(x,E(w,z)=b)-\Pr[M(x,y)=b]| \geq 1/6 \)
  - extractor property: \( \leq 2^{-k} \) bad \( w \)
  - \( n \) random bits; \( 2^{-k} \) bad

---

**RL**

- **Recall: probabilistic Turing Machine**
  - deterministic TM with additional tape for "coin flips"

- **RL (Random Logspace)**
  - \( L \in RL \) if there is a probabilistic logspace TM \( M \):
    - \( x \in L \Rightarrow \Pr[M(x,y) \text{ accepts}] \geq 1/2 \)
    - \( x \notin L \Rightarrow \Pr[M(x,y) \text{ rejects}] \geq 1 \)
  - important detail #1: only allow one-way access to coin-flip tape
  - important detail #2: explicitly require to run in polynomial time

- \( L \subseteq RL \subseteq NL \subseteq \text{TIME}(\log^2 n) \)
- **Theorem (SZ):** \( RL \subseteq \text{TIME}(\log^{3/2} n) \)

---

**Undirected STCONN**

- **Recall:** STCONN is NL-complete.
- **Undirected STCONN:** given an **undirected** graph \( G = (V, E) \), nodes \( s, t \), is there a path \( s \rightarrow t \)
- **Theorem:** \( \text{USTCONN} \in RL \)

- **Proof sketch:** (in Papadimitriou)
  - add self-loop to each vertex (technical reasons)
  - start at \( s \), take a random walk for \( 2|V||E| \) steps, accept if see \( t \)
  - Lemma: expected return time for any node \( i \) is \( 2|E|/d_i \)
  - suppose \( s = v_1, v_2, ..., v_n = t \) is a path
  - expected time from \( v_i \) to \( v_{i+1} \)
    \( (d/2)(2|E|/d) = |E| \)
  - expected time to reach \( v_n \) is \( |V||E| \)
  - \( \Pr[\text{fail reach } t \text{ in } 2|V||E| \text{ steps}] \leq 1/2 \)