CS151
Complexity Theory

Lecture 10

April 29, 2004

Outline

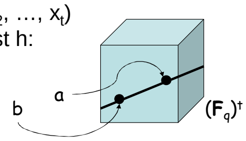
- Decoding Reed-Muller codes
- Transforming worst-case hardness into average-case hardness
- Extractors

Decoding RM

- Main idea: reduce to decoding RS

RM codeword $p(x_1, x_2, \ldots, x_t)$
of total degree at most h:

$L_1(z) = a_1z + b_1(1-z)$
$L_2(z) = a_2z + b_2(1-z)$
…
$L_t(z) = a_tz + b_t(1-z)$

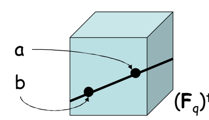


$p_{|L}(z) = p(L_1(z), L_2(z), \ldots, L_t(z))$

"restriction to line L(z) passing though a, b"

Decoding RM

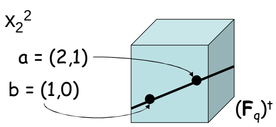
Two key observations:



1. If p has total degree at most h, then $p_{|L}$ has degree at most h
2. $p_{|L}$ is a univariate polynomial

Decoding RM

- Example:
  - $p(x_1, x_2) = x_1^2x_2 + x_2^2$



  - $L_1(z) = 2z + 1(1-z) = z + 1$
  - $L_2(z) = 1z + 0(1-z) = z$
  - $p_{|L}(z) = (z+1)^2z + z^2 = 2z^3 + 2z^2 + z$

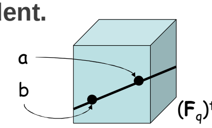
Decoding RM

Key property:

- If pick a, b **randomly** in $(\mathbf{F}_q)^t$ then points in the vector

$$(az + b(1-z))_{z \in \mathbf{F}_q}$$

are **pairwise independent.**



Decoding RM

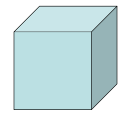
- Meaning of pairwise independent in this context:
- L = az + b(1-z)

  for all w, z $\in$ $\mathbf{F}_q$, α, $\beta \in (\mathbf{F}_q)^t$

  $\Pr_{a,b}[L(w) = \alpha \mid L(z) = \beta] = 1/q^t$
- every pair of points on L behaves just as if it was picked independently

Decoding RM (small error)

- The setup:
  - Codeword is a polynomial $p:(\mathbf{F}_q)^t \to \mathbf{F}_q$ with total degree h
    - k = (h + t choose t)
    - $n = q^t$



  - Given received word $R:(\mathbf{F}_q)^t \to \mathbf{F}_q$

$(\mathbf{F}_q)^t$

  - Suppose $\Pr_a[p(a) = R(a)] > 1 - \delta$
  - Try to recover p by accessing R

Decoding RM (small error)

- To decode one position $a \in (\mathbf{F}_q)^t$:
  - pick b **randomly** in $(\mathbf{F}_q)^t$
  - L is line passing through a, b
  - q pairs (z, $R_{|L}(z)$) for $z \in \mathbf{F}_q$
  - each point L(z) random in $(\mathbf{F}_q)^t$
  - E[# errors hit] < δq
  - $\Pr_b$[# errors hit > 4δq] < ¼ (Markov)
  - try to find degree h univariate poly. r for which

    $\Pr_z[r(z) \neq R_{|L}(z)] \leq 4\delta$

RS decoding!

Decoding RM (small error)

- with probability 3/4 data is close to the univariate polynomial $p_{|L}$, and then

  $r(1) = p_{|L}(1) = p(a)$
- output r(1)
- repeat O(log n) times, choose top vote-getter
- reduces error probability to 1/3n

- union bound: all n positions decoded correctly with probability at least 2/3

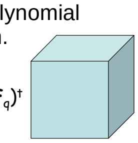
Decoding RM (small error)

- Assume relative distance 1-h/q > ½
- Previous algorithm works when 4δ < 1/4
- requires very small relative error δ < 1/16

List-decoding RM (large error)

- The setup:
  - Given received word $R:(\mathbf{F}_q)^t \to \mathbf{F}_q$
  - each nearby codeword is a polynomial $p:(\mathbf{F}_q)^t \to \mathbf{F}_q$ with total degree h.
    - k = (h + t choose t)
    - $n = q^t$

$(\mathbf{F}_q)^t$



  - By accessing R, try to recover all p such that $\Pr_a[p(a) = R(a)] > \varepsilon$

List-decoding RM (large error)

- Procedure (sketch):
  - pick a and b **randomly** in $(\mathbf{F}_q)^t$
  - L is line passing through a, b
  - q pairs $(z, R_{|L}(z))$ for $z \in \mathbf{F}_q$
  - each point L(z) random in $(\mathbf{F}_q)^t$
  - E[# non-errors hit] > εq
  - $\Pr_{a,b}$[# non-errors hit < $\varepsilon q/2$] < $4/(\varepsilon q)$ (Chebyshev)

List-decoding RM (large error)

- using RS list-decoding, find list of all degree h univariate polynomials $r_1, r_2, \ldots, r_m$ for which
  $$\Pr_z[r_i(z) = R_{|L}(z)] \geq \varepsilon q/2$$
- One r_i is $p_{|L}$ (i.e., $r_i(z) = p_{|L}(z)$) for all z)
- How can we find that one?
- given p(b), can find it with high probability:
  $\Pr_{a,b}[\text{exists } i \neq j \text{ for which } r_j(0) = r_i(0)] < 8m^2h/q$
- find the *unique* r_i for which $r_i(0) = p(b)$; output $r_i(1)$, which should be $p_{|L}(1) = p(a)$

List-decoding RM (large error)

- $\Pr_{a,b}$[# non-errors hit < $\varepsilon q/2$] < $4/(\varepsilon q)$
- given p(b), $\Pr_{a,b}$[fail to output p(a)] < $8m^2h/q$
- $\Pr_{a,b}$[output p(a)] > $1 - 4/(\varepsilon q) - 8m^2h/q$
- Key: try for O(log n) random b values
  - for each b, try all q values for p(b) (one is right!)
  - apply RM decoding from small error
  - each good trial gives small error required for previous decoding algorithm

Decoding RM (large error)

- Requirements on parameters:
  - $\varepsilon q/2 > (2hq)^{1/2} \Rightarrow q > 8h/\varepsilon^2$ (for RS list decoding)
  - $4/(\varepsilon q) < 1/64 \Rightarrow q > 256/\varepsilon$
  - know $m < 2/\varepsilon$ from q-ary Johnson bound
  - $8m^2h/q < 32h/(q\varepsilon^2) < 1/64 \Rightarrow q > 2^{11}h/\varepsilon^2$

conclude: $\Pr_{a,b}$[output p(a)] > 1 – 1/32

Decoding RM (large error)

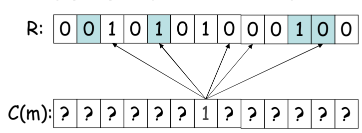
$\Pr_{a,b}$[fail to output p(a)] < 1/32

$\Pr_b[\Pr_a$[fail to output p(a)] > 1/16] < ½

- on trial with correct value for p(b), with probability at least ½ RM decoder from small error is correct on all a
- repetition decreases error exponentially
- union bound: all p with agreement ε included in list

Local decodability

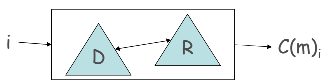
- Amazing property of decoding method:



- Local decodability: each symbol of C(m) decoded by looking at small number of symbols in R

Local decodability

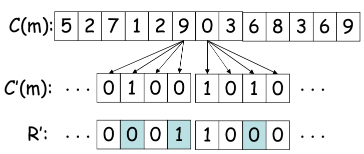
- Local decodability: each symbol of C(m) decoded by looking at small number of symbols in R
  - small decoding circuit D
  - small circuit computing R
  - implies small circuit computing C(m)



Concatenation

- Problem: symbols of $\mathbf{F}_q$ rather than bits
- Solution: encode each symbol with binary code
  - our choice:
    - RM with degree h ≤ 1, field size 2
    - # variables t = log q
    - also called "Hadamard code"
  - Schwartz-Zippel implies distance = ½

Concatenation



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

The concatenated code

- **outer code**: RM with parameters
  - field size q, degree h, dimension t
  - # coefficients (h+t choose t) > k
  - q^t = poly(k)
- **inner code**: RM with parameters
  - field size q' = 2
  - degree h' = 1
  - dimension t' = log q
- block length n = $q^t q$ = poly(k)

Codes and Hardness

- Recall our strategy:

truth table of $f:\{0,1\}^{\log k} \rightarrow \{0,1\}$

(worst-case hard)

m: | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |

truth table of $f':\{0,1\}^{\log n} \rightarrow \{0,1\}$

(average-case hard)

C(m): | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |

Hardness amplification

Claim: $f \in \mathbf{E} \Rightarrow f' \in \mathbf{E}$
- $f \in \mathbf{E} \Rightarrow$ f computable by a TM running in time $2^{c(\log k)} = k^c$
- to write out truth table of f: time kk^c
- to compute truth table of f ': time poly(n, k)
- recall n = poly(k)
- f ' computable by TM running in time $n^{c'} = 2^{c'(\log n)} \Rightarrow f' \in \mathbf{E}$

Hardness amplification

- Need to prove: if f' is s'-approximable by circuit C, then f is computable by a size s = poly(s') circuit.

f: 0 1 1 0 0 0 1 0 "message"

f': 0 1 1 0 0 0 1 0 0 0 0 1 0 "codeword"

C: 0 0 1 0 1 0 1 0 0 0 1 0 0 "received word"

Hardness amplification

- suppose f' is s'-approximable by C
  - $\Pr_x[C(x) = f'(x)] \geq ½ + 1/s'$
  - at least s'/2 "inner" blocks have C, f' agreement ½ + 1/(2s')
  - Johnson Bound: at most $O(s'^2)$ inner codewords with this agreement
  - find by brute force search: time = O(q)
  - pick random mesg. from list for each symbol
  - get "outer" received word with agreement $1/s'^3$

Hardness amplification

- "outer" received word with agreement $\varepsilon = 1/s'^3$
- can only uniquely decode from agreement > ½ (since relative distance < 1)
- our agreement << ½

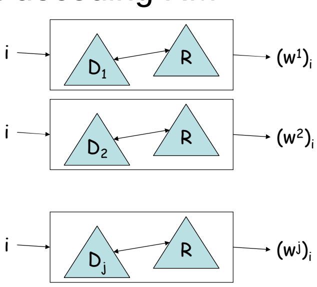
need to use list-decoding of RM for large error

Setting parameters

- have to handle agreement $\varepsilon = 1/s'^3$
- pick q, h, t:
  - need $q > \Omega(h/\varepsilon^2)$ for decoder to work
  - need (h + t choose t) > k (note s' < k)
  - need $q^t = \text{poly}(k)$
- h = s'
- $t \approx (\log k)/(\log s')$
- $q = \Omega(h/\varepsilon^2) = \Omega(s'^3)$

List-decoding RM

- final result: short list of circuits
- size = poly(q)poly(s') =poly(s')
- one computes f exactly!
- circuit for f of size poly(s')



Putting it all together

Theorem 1 (IW, STV): If **E** contains functions that require size $2^{\Omega(n)}$ circuits, then **E** contains $2^{\Omega(n)}$ -unapproximable functions.

- Theorem (NW): if **E** contains $2^{\Omega(n)}$-unapproximable functions then **BPP** = **P**.

Theorem 2 (IW): **E** requires exponential size circuits ⇒ **BPP** = **P**.

Putting it all together

- Proof of Theorem 1:
  - let $f = \{f_n\}$ be such a function that requires size $s = 2^{\delta n}$ circuits
  - define $f' = \{f_n'\}$ be just-described encoding of (truth table of) f
  - just showed: if f' is $s' = 2^{\delta' n}$-approximable, then f is computable by size $s = poly(s') = 2^{\delta n}$ circuit.
  - contradiction.

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

Extractors

- "Hardware" side
  - what physical source?
  - ask the physicists…
- "Software" side
  - what is the minimum we need from the physical source?

Extractors

- imperfect sources:
  - "stuck bits": 

  - "correlation": 
  - "more insidious correlation": perfect squares
- there are specific ways to get independent unbiased random bits from specific imperfect physical sources

Extractors

- want to assume we don't know details of physical source
- **general model** capturing all of these?
  - yes: "min-entropy"
- **universal procedure** for all imperfect sources?
  - yes: "extractors"