

CS151 Complexity Theory

Lecture 10
April 26, 2007

Hardness vs. randomness

- BMY pseudo-random generator:
 - one generator fooling all poly-size bounds
 - one-way-permutation is hard function
 - implies hard function in $\mathbf{NP} \cap \mathbf{coNP}$
- New idea (Nisan-Wigderson):
 - for each poly-size bound, one generator
 - hard function allowed to be in
$$\mathbf{E} = \bigcup_k \mathbf{DTIME}(2^{kn})$$

April 26, 2007

CS151 Lecture 10

2

NW PRG

- First attempt: build PRG assuming \mathbf{E} contains **unapproximable** functions

Definition: The function family

$$f = \{f_n\}, f_n: \{0,1\}^n \rightarrow \{0,1\}$$

is $s(n)$ -unapproximable if for every family of size $s(n)$ circuits $\{C_n\}$:

$$\Pr_x[C_n(x) = f_n(x)] \leq \frac{1}{2} + 1/s(n).$$

April 26, 2007

CS151 Lecture 10

3

One bit

- Suppose $f = \{f_n\}$ is $s(n)$ -unapproximable, for $s(n) = 2^{\delta n}$, and in \mathbf{E}
- a “1-bit” generator family $G = \{G_n\}$:

$$G_n(y) = y \circ f_{\log n}(y)$$

- seed length $t = \log n$
- output length $m = \log n + 1$ (want n^δ)
- fooling size $s \approx s(\log n) = n^\delta$
- running time n^c
- error $\epsilon \approx 1/s(\log n) = 1/n^\delta = 1/m$

April 26, 2007

CS151 Lecture 10

4

Many bits

- Try outputting many evaluations of f :

$$G(y) = f(b_1(y)) \circ f(b_2(y)) \circ \dots \circ f(b_m(y))$$

- Seems that a predictor must evaluate $f(b_i(y))$ to predict i -th bit
- Does this work?

April 26, 2007

CS151 Lecture 10

5

Many bits

- Try outputting many evaluations of f :

$$G(y) = f(b_1(y)) \circ f(b_2(y)) \circ \dots \circ f(b_m(y))$$

- predictor might notice correlations without having to compute f
- but, more subtle argument works for a specific choice of $b_1 \dots b_m$

April 26, 2007

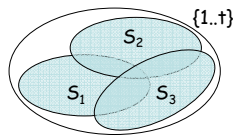
CS151 Lecture 10

6

Nearly-Disjoint Subsets

Definition: $S_1, S_2, \dots, S_m \subset \{1 \dots t\}$ is an (h, a) design if

- for all i , $|S_i| = h$
- for all $i \neq j$, $|S_i \cap S_j| \leq a$



April 26, 2007

CS151 Lecture 10

7

Nearly-Disjoint Subsets

Lemma: for every $\epsilon > 0$ and $m < n$ can in $\text{poly}(n)$ time construct an

$(h = \log n, a = \epsilon \log n)$ design

$S_1, S_2, \dots, S_m \subset \{1 \dots t\}$ with $t = O(\log n)$.

April 26, 2007

CS151 Lecture 10

8

Nearly-Disjoint Subsets

• Proof sketch:

- pick random $(\log n)$ -subset of $\{1 \dots t\}$
- set $t = O(\log n)$ so that expected overlap with a fixed S_i is $\epsilon \log n / 2$
- probability overlap with S_i is $> \epsilon \log n$ is at most $1/n$
- union bound: some subset has required small overlap with *all* S_i picked so far...
- find it by exhaustive search; repeat n times.

April 26, 2007

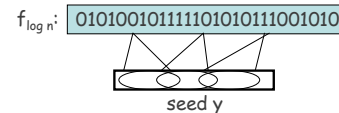
CS151 Lecture 10

9

The NW generator

- $f \in \mathbf{E}$ $s(n)$ -unapproximable, for $s(n) = 2^{\delta n}$
- $S_1, \dots, S_m \subset \{1 \dots t\}$ ($\log n, a = \delta \log n / 3$) design with $t = O(\log n)$

$$G_n(y) = f_{\log n}(y_{|S_1}) \circ f_{\log n}(y_{|S_2}) \circ \dots \circ f_{\log n}(y_{|S_m})$$



April 26, 2007

CS151 Lecture 10

10

The NW generator

Theorem (Nisan-Wigderson): $G = \{G_n\}$ is a pseudo-random generator with:

- seed length $t = O(\log n)$
- output length $m = n^{\delta/3}$
- running time n^c
- fooling size $s = m$
- error $\epsilon = 1/m$

April 26, 2007

CS151 Lecture 10

11

The NW generator

• Proof:

- assume does not ϵ -pass statistical test $C = \{C_m\}$ of size s :

$$|\Pr_x[C(x) = 1] - \Pr_y[C(G_n(y)) = 1]| > \epsilon$$

- can transform this distinguisher into a predictor P of size $s' = s + O(m)$:

$$\Pr_y[P(G_n(y)_1 \dots y_{i-1}) = G_n(y)_i] > 1/2 + \epsilon/m$$

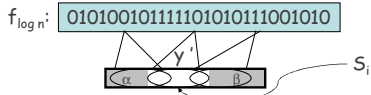
April 26, 2007

CS151 Lecture 10

12

The NW generator

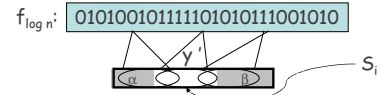
$$G_n(y) = f_{\log n}(y|_{S_1}) \circ f_{\log n}(y|_{S_2}) \circ \dots \circ f_{\log n}(y|_{S_m})$$



- **Proof (continued):**
 - $\Pr_y[P(G_n(y)_{1\dots i-1}) = G_n(y)_i] > \frac{1}{2} + \epsilon/m$
 - fix bits outside of S_i to preserve advantage:
 - $\Pr_y[P(G_n(\alpha y' \beta)_{1\dots i-1}) = G_n(\alpha y' \beta)_i] > \frac{1}{2} + \epsilon/m$

The NW generator

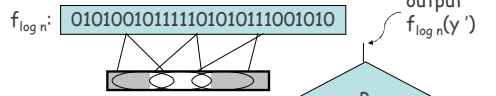
$$G_n(y) = f_{\log n}(y|_{S_1}) \circ f_{\log n}(y|_{S_2}) \circ \dots \circ f_{\log n}(y|_{S_m})$$



- **Proof (continued):**
 - $G_n(\alpha y' \beta)_i$ is exactly $f_{\log n}(y')$
 - for $j \neq i$, as vary y' , $G_n(\alpha y' \beta)_j$ varies over 2^a values!
 - hard-wire up to $(m-1)$ tables of 2^a values to provide $G_n(\alpha y' \beta)_{1\dots i-1}$

The NW generator

$$G_n(y) = f_{\log n}(y|_{S_1}) \circ f_{\log n}(y|_{S_2}) \circ \dots \circ f_{\log n}(y|_{S_m})$$



- size $m + O(m) + (m-1)2^a$
- $< s(\log n) = n^\delta$
- advantage $\epsilon/m = 1/m^2 > 1/s(\log n) = n^{-\delta}$
- contradiction

Worst-case vs. Average-case

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

- How reasonable is unapproximability assumption?
- Hope: obtain $\mathbf{BPP} = \mathbf{P}$ from worst-case complexity assumption
 - try to fit into existing framework without new notion of “unapproximability”

Worst-case vs. Average-case

Theorem (Impagliazzo-Wigderson, Sudan-Trevisan-Vadhan)
 If \mathbf{E} contains functions that require size $2^{\Omega(n)}$ circuits, then \mathbf{E} contains $2^{\Omega(n)}$ -unapproximable functions.

- **Proof:**
 - main tool: error correcting code

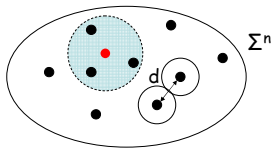
Error-correcting codes

- Error Correcting Code (ECC):
 - $C: \Sigma^k \rightarrow \Sigma^n$
- message $m \in \Sigma^k$
- received word R
 - $C(m)$ with some positions corrupted
- if not too many errors, can decode: $D(R) = m$
- parameters of interest:
 - rate: k/n
 - distance:

$$d = \min_{m \neq m'} \Delta(C(m), C(m'))$$

Distance and error correction

- C is an ECC with distance d
- can **uniquely** decode from up to $\lfloor d/2 \rfloor$ errors



April 26, 2007

CS151 Lecture 10

19

Distance and error correction

- can find **short list** of messages (one correct) after closer to d errors!

Theorem (Johnson): a binary code with distance $(\frac{1}{2} - \delta^2)n$ has at most $O(1/\delta^2)$ codewords in any ball of radius $(\frac{1}{2} - \delta)n$.

April 26, 2007

CS151 Lecture 10

20

Example: Reed-Solomon

- alphabet $\Sigma = \mathbb{F}_q$: field with q elements
- message $m \in \Sigma^k$
- polynomial of degree at most $k-1$

$$p_m(x) = \sum_{i=0}^{k-1} m_i x^i$$
- codeword $C(m) = (p_m(x))_{x \in \mathbb{F}_q}$
- rate = k/q

April 26, 2007

CS151 Lecture 10

21

Example: Reed-Solomon

- Claim: distance $d = q - k + 1$
 - suppose $\Delta(C(m), C(m')) < q - k + 1$
 - then there exist polynomials $p_m(x)$ and $p_{m'}(x)$ that agree on more than $k-1$ points in \mathbb{F}_q
 - polynomial $p(x) = p_m(x) - p_{m'}(x)$ has more than $k-1$ zeros
 - but degree at most $k-1$...
 - contradiction.

April 26, 2007

CS151 Lecture 10

22

Example: Reed-Muller

- Parameters: t (dimension), h (degree)
- alphabet $\Sigma = \mathbb{F}_q$: field with q elements
- message $m \in \Sigma^k$
- multivariate polynomial of total degree at most h :

$$p_m(x) = \sum_{i=0}^{k-1} m_i M_i$$

$\{M_i\}$ are all monomials of degree $\leq h$

April 26, 2007

CS151 Lecture 10

23

Example: Reed-Muller

- M_i is monomial of total degree h
 - e.g. $x_1^2 x_2 x_4^3$
 - need # monomials $(h+t \text{ choose } t) > k$
- codeword $C(m) = (p_m(x))_{x \in (\mathbb{F}_q)^t}$
- rate = k/q^t
- Claim: distance $d = (1 - h/q)q^t$
 - proof: Schwartz-Zippel: polynomial of degree h can have at most h/q fraction of zeros

April 26, 2007

CS151 Lecture 10

24

Codes and hardness

- Reed-Solomon (RS) and Reed-Muller (RM) codes are efficiently encodable
- **efficient** unique decoding?
 - yes (classic result)
- **efficient list**-decoding?
 - yes (recent result. RS on problem set.)

April 26, 2007

CS151 Lecture 10

25

Codes and Hardness

- Use for worst-case to average case:
 - truth table of $f: \{0,1\}^{\log k} \rightarrow \{0,1\}$
(worst-case hard)
 - $m: 01100010$
 - truth table of $f': \{0,1\}^{\log n} \rightarrow \{0,1\}$
(average-case hard)
 - $Enc(m): 0110001000010$

April 26, 2007

CS151 Lecture 10

26

Codes and Hardness

- if $n = \text{poly}(k)$ then
 - $f \in E$ implies $f' \in E$
- Want to be able to prove:
 - if f' is s' -approximable,
 - then f is computable by a
 - size $s = \text{poly}(s')$ circuit

April 26, 2007

CS151 Lecture 10

27

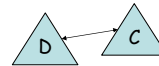
Codes and Hardness

- Key: circuit C that approximates f' implicitly gives received word R

R: 00101010000100

Enc(m): 0110001000010

- Decoding procedure D “computes” f exactly



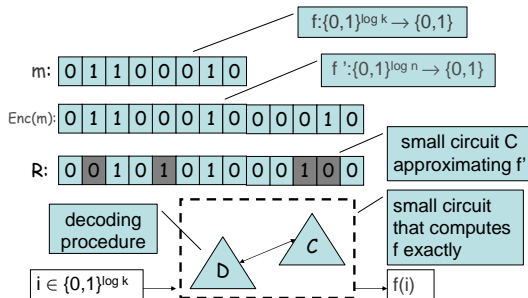
- Requires special notion of efficient decoding

April 26, 2007

CS151 Lecture 10

28

Codes and Hardness



April 26, 2007

CS151 Lecture 10

29

Encoding

- use a (variant of) Reed-Muller code *concatenated* with the Hadamard code
 - q (field size), t (dimension), h (degree)
- **encoding procedure:**
 - message $m \in \{0,1\}^k$
 - subset $S \subseteq F_q$ of size h
 - efficient 1-1 function $Emb: [k] \rightarrow S^t$
 - find coeffs of degree h polynomial $p_m: F_q^t \rightarrow F_q$ for which $p_m(Emb(i)) = m_i$ for all i (linear algebra)

so, need $h^t \geq k$

April 26, 2007

CS151 Lecture 10

30

Encoding

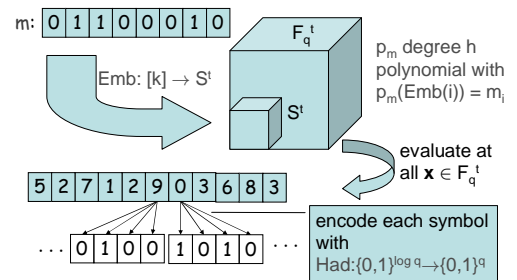
- **encoding procedure (continued):**
 - Hadamard code $\text{Had}: \{0,1\}^{\log q} \rightarrow \{0,1\}^q$
 - = Reed-Muller with field size 2, dim. $\log q$, deg. 1
 - distance $\frac{1}{2}$ by Schwartz-Zippel
 - final codeword: $(\text{Had}(p_m(\mathbf{x})))_{\mathbf{x} \in F_q^t}$
 - evaluate p_m at all points, and encode each evaluation with the Hadamard code

April 26, 2007

CS151 Lecture 10

31

Encoding



April 26, 2007

CS151 Lecture 10

32

Decoding

Enc(m): 0 1 1 0 0 0 1 0 0 0 0 1

R: 0 0 1 0 1 0 1 0 0 0 1 0

- small circuit C computing R, agreement $\frac{1}{2} + \delta$
- **Decoding step 1**
 - produce circuit C' from C
 - given $\mathbf{x} \in F_q^t$ outputs "guess" for $p_m(\mathbf{x})$
 - C' computes $\{z : \text{Had}(z) \text{ has agreement } \frac{1}{2} + \delta/2 \text{ with } x\text{-th block}\}$, outputs random z in this set

April 26, 2007

CS151 Lecture 10

33

Decoding

- **Decoding step 1 (continued):**

- for at least $\delta/2$ of blocks, agreement in block is at least $\frac{1}{2} + \delta/2$
- Johnson Bound: when this happens, list size is $S = O(1/\delta^2)$, so probability C' correct is $1/S$
- altogether:
 - $\Pr_x[C'(x) = p_m(x)] \geq \Omega(\delta^3)$
 - C' makes q queries to C
 - C' runs in time $\text{poly}(q)$

April 26, 2007

CS151 Lecture 10

34

Decoding

p_m : 5 2 7 1 2 9 0 3 6 8 3

R': 5 9 7 1 6 9 0 3 6 8 1

- small circuit C' computing R', agreement $\delta' = \Omega(\delta^3)$
- **Decoding step 2**
 - produce circuit C'' from C'
 - given $\mathbf{x} \in \text{emb}(1,2,\dots,k)$ outputs $p_m(\mathbf{x})$
 - idea: restrict p_m to a random curve; apply efficient R-S list-decoding; fix "good" random choices

April 26, 2007

CS151 Lecture 10

35