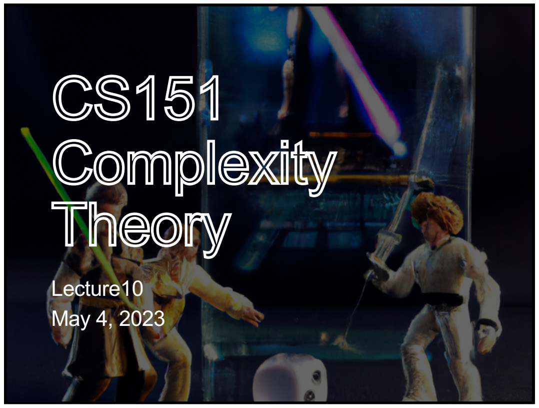


1

Hardness vs. randomness

- BMY pseudo-random generator:
  - one generator fooling all poly-size bounds
  - one-way-permutation is hard function
  - implies hard function in **NP** ∩ **coNP**
- New idea (Nisan-Wigderson):
  - for each poly-size bound, one generator
  - hard function allowed to be in

$$\mathbf{E} = \cup_k \mathbf{DTIME}(2^{kn})$$

2

Comparison

	BMY: $\forall\ \delta > 0$ PRG G^δ	NW: PRG G
seed length	$\mathbf{t} = m^\delta$	$\mathbf{t} = O(\log m)$
running time	$t^c m$	m^c
output length	$\mathbf{m}$ <<	$\mathbf{m}$ >
error	$\boldsymbol{\varepsilon} < 1/m^d$ (all d)	$\boldsymbol{\varepsilon} < 1/m$
fooling size	$\mathbf{s} = m^e$ (all e)	$\mathbf{s} = m$

3

NW PRG

- NW: for fixed constant δ, G = $\{G_n\}$ with

seed length	$\mathbf{t} = O(\log n)$	$\mathbf{t} = O(\log m)$
running time	n^c	m^c
output length	$\mathbf{m} = n^\delta$	$\mathbf{m}$
error	$\boldsymbol{\varepsilon} < 1/m$	
fooling size	$\mathbf{s} = m$	

- Using this PRG we obtain **BPP = P**
  - to fool size n^k use $G_{n^{k/\delta}}$
  - running time $O(n^k + n^{ck/\delta})2^t = \text{poly}(n)$

4

NW PRG

- First attempt: build PRG assuming **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 $\{C_n\}$:

$$\Pr_x[C_n(x) = f_n(x)] \leq ½ + 1/s(n).$$

5

One bit

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

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

- Idea: if not a PRG then exists a predictor that computes $f_{\log n}$ with better than ½ + 1/s(log n) agreement; contradiction.

6

One bit

- Suppose $f = \{f_n\}$ is $s(n)$-unapproximable, for $s(n) = 2^{\delta n}$, and in **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** ≈ s(log n) = n^δ
  - running time n^c
  - error **ε** ≈ 1/s(log n) = $1/n^\delta$

7

Many bits

- Try outputting many evaluations of f:

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

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

8

Many bits

- Try outputting many evaluations of f:

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

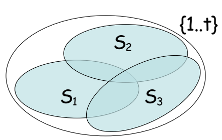
- predictor might notice correlations without having to compute f
- but, more subtle argument works for a specific choice of $b_1 \ldots b_m$

9

Nearly-Disjoint Subsets

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

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



10

Nearly-Disjoint Subsets

Lemma: for every ε > 0 and m < n can in poly(n) time construct an

(h = log n, a = εlog n) design

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

11

Nearly-Disjoint Subsets

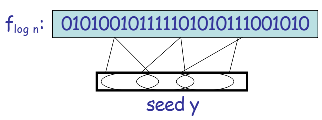
- Proof sketch:
  - pick random (log n)-subset of {1…t}
  - set t = O(log n) so that expected overlap with a fixed S_i is εlog n/2
  - probability overlap with S_i is > εlog n is at most 1/n
  - union bound: some subset has required small overlap with *all* S_i picked so far…
  - find it by exhaustive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

12

The NW generator

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

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



13

The NW generator

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

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

14

The NW generator

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

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

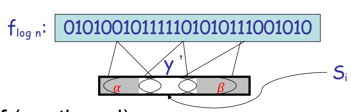
  - can transform this **distinguisher** into a **predictor** P of size $s' = s + O(m)$:

    $\Pr_y[P(G_n(y)_{1\ldots i-1}) = G_n(y)_i] > ½ + \varepsilon/m$

15

The NW generator

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



- Proof (continued):

  $\Pr_y[P(G_n(y)_{1\ldots i-1}) = G_n(y)_i] > ½ + \varepsilon/m$

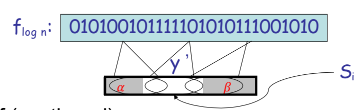
  – fix bits outside of S_i to preserve advantage:

  $\Pr_{y'}[P(G_n(\alpha y'\beta)_{1\ldots i-1}) = G_n(\alpha y'\beta)_i] > ½ + \varepsilon/m$

16

The NW generator

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

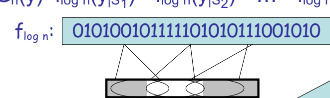


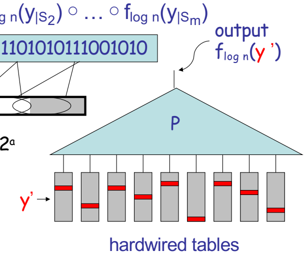
- Proof (continued):
  - $G_n(\alpha y'\beta)_i$ is exactly $f_{\log n}(y')$
  - for $j \neq i$, as vary y', $G_n(\alpha y'\beta)_j$ varies over 2^a values!
  - hard-wire up to $(m-1)$ tables of 2^a values to provide $G_n(\alpha y'\beta)_{1\ldots i-1}$

17

The NW generator

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





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

18

Worst-case vs. Average-case

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

- How reasonable is unapproximability assumption?
- Hope: obtain **BPP** = **P** from worst-case complexity assumption
  - try to fit into existing framework without new notion of "unapproximability"

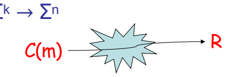
Worst-case vs. Average-case

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

- Proof:
  - main tool: error correcting code

Error-correcting codes

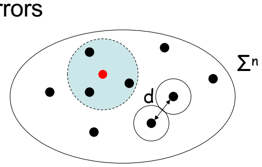
- Error Correcting Code (ECC):
  $C:\Sigma^k \rightarrow \Sigma^n$
- message $m \in \Sigma^k$
- received word R
  - C(m) with some positions corrupted



- 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



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 $(\frac{1}{2} - \delta)n$.

Example: Reed-Solomon

- alphabet $\Sigma = \mathbf{F}_q$: field with q elements
- message $m \in \Sigma^k$
- polynomial of degree at most k-1
  $p_m(x) = \Sigma_{i=0\ldots k-1}\, m_i x^i$
- codeword $C(m) = (p_m(x))_{x \in \mathbf{F}_q}$
- rate = k/q

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 $\mathbf{F}_q$
  - polnomial $p(x) = p_m(x) - p_{m'}(x)$ has more than k-1 zeros
  - but degree at most k-1…
  - contradiction.

Example: Reed-Muller

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

$$p_m(x) = \Sigma_{i=0\ldots k-1}\, m_i M_i$$

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

Example: Reed-Muller

- M_i is monomial of total degree h
  - e.g. $x_1^2x_2x_4^3$
  - need # monomials (h+t choose t) > k
- codeword $C(m) = (p_m(x))_{x \in (\mathbf{F}_q)^t}$
- 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

Codes and hardness

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

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

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

(average-case hard)

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

Codes and Hardness

- if n = poly(k) then

$f \in \mathbf{E}$ implies $f' \in \mathbf{E}$

- Want to be able to prove:

if f' is s'-approximable,

then f is computable by a

size s = poly(s') circuit

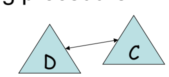
Codes and Hardness

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

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

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

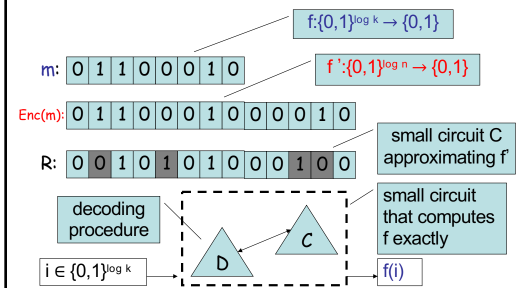
- Decoding procedure D "computes" f exactly



- Requires special notion of efficient decoding

31

Codes and Hardness



32

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 $p_m: \mathbf{F}_q^{\,t} \rightarrow \mathbf{F}_q$ for which $p_m(\mathrm{Emb}(i)) = m_i$ for all i (linear algebra)

so, need $h^t \geq k$

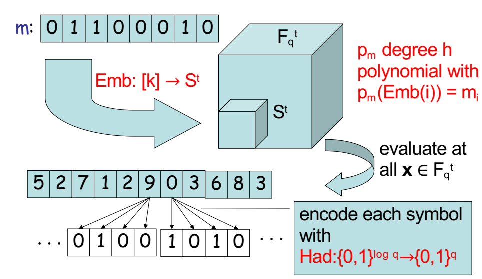
33

Encoding

- **encoding procedure** (continued):
  - Hadamard code Had:$\{0,1\}^{\log q} \rightarrow \{0,1\}^q$
    - = Reed-Muller with field size 2, dim. log q, deg. 1
    - distance ½ by Schwartz-Zippel
  - final codeword: $(\mathrm{Had}(p_m(\mathbf{x})))_{x \in F_q^{\,t}}$
    - evaluate p_m at all points, and encode each evaluation with the Hadamard code

34

Encoding



35

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 $½ + \delta$
- **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 : Had(z) has agreement $½ + \delta/2$ with x-th block}, outputs random z in this set

36

Decoding

- **Decoding step 1** (continued):
  - for at least $\delta/2$ of blocks, agreement in block is at least $½ + \delta/2$
  - 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 poly(q)

37

Decoding

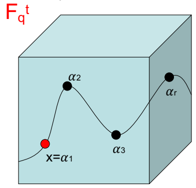
p_m:	5	2	7	1	2	9	0	3	6	8	3
R':	5	9	7	1	6	9	0	3	6	8	1

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

38

Restricting to a curve

- points $x=\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_r \in F_q^t$ specify a degree r curve $L : F_q \rightarrow F_q^t$



- $w_1, w_2, \ldots, w_r$ are distinct elements of F_q
- for each i, $L_i : F_q \rightarrow F_q$

is the degree r poly for which $L_i(w_j) = (\alpha_j)_i$ for all j

- Write $p_m(L(z))$ to mean $p_m(L_1(z), L_2(z), \ldots, L_t(z))$

degree r·h·t *univariate* poly

- $p_m(L(w_1)) = p_m(x)$

39