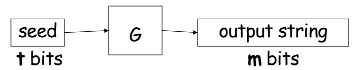
CS151 Complexity Theory

Lecture 9
May 2, 2023

1

Derandomization

- Pseudo-Random Generator (PRG):



  - G is efficiently computable
  - "stretches" **t** bits into **m** bits
  - "fools" small circuits: for all circuits C of size at most **s**:

$$|Pr_y[C(y) = 1] – Pr_z[C(G(z)) = 1]| \le \boldsymbol{\varepsilon}$$

2

Blum-Micali-Yao PRG

- Initial goal: for all $1 > \delta > 0$, we will build a family of PRGs $\{G_m\}$ with:

  output length **m** fooling size **s** = m
  seed length **t** = m^δ running time m^c
  error $\boldsymbol{\varepsilon} < 1/6$

- implies: $\mathbf{BPP} \subseteq \cap_{\delta>0} \mathbf{TIME}(2^{n^\delta}) \subsetneq \mathbf{EXP}$
- Why? simulation runs in time
  $O(m+m^c)(2^{m^\delta}) = O(2^{m^{2\delta}}) = O(2^{n^{2k\delta}})$

3

Blum-Micali-Yao PRG

- PRGs of this type imply existence of one-way-functions
  - we'll use widely believed cryptographic assumptions

Definition: One Way Function (OWF): function family $f = \{f_n\}$, $f_n:\{0,1\}^n \to \{0,1\}^n$
  - f_n computable in poly(n) time
  - for every family of poly-size circuits $\{C_n\}$
    $Pr_x[C_n(f_n(x)) \in f_n^{-1}(f_n(x))] \le \varepsilon(n)$
  - $\varepsilon(n) = o(n^{-c})$ for all c

4

Blum-Micali-Yao PRG

- believe one-way functions exist
  - e.g. integer multiplication, discrete log, RSA (w/ minor modifications)

Definition: One Way Permutation: OWF in which f_n is 1-1
  - can simplify "$Pr_x[C_n(f_n(x)) \in f_n^{-1}(f_n(x))] \le \varepsilon(n)$" to
    $Pr_y[C_n(y) = f_n^{-1}(y)] \le \varepsilon(n)$

5

First attempt

- attempt at PRG from OWP f:
  - $t = m^\delta$
  - $y_0 \in \{0,1\}^t$
  - $y_i = f_t(y_{i-1})$
  - $G(y_0) = y_{k-1}y_{k-2}y_{k-3}\ldots y_0$
  - $k = m/t$
- computable in time at most
  $kt^c < mt^{c-1} = m^c$

6

First attempt

- output is "**unpredictable**":
  - no poly-size circuit C can output y_{i-1} given $y_{k-1}y_{k-2}y_{k-3}\ldots y_i$ with non-negl. success prob.
  - if C could, then given y_i can compute $y_{k-1}, y_{k-2}, \ldots, y_{i+2}, y_{i+1}$ and feed to C
  - result is poly-size circuit to compute
    $y_{i-1} = f_t^{-1}(y_i)$ from y_i
  - note: we're using that f_t is 1-1

May 2, 2023 CS151 Lecture 9 7

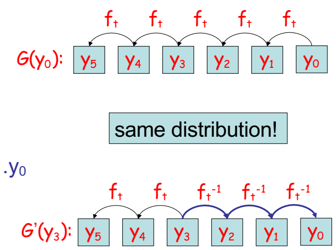
7

First attempt

attempt:

- $y_0 \in \{0,1\}^t$
- $y_i = f_t(y_{i-1})$

- $G(y_0) =$
  $y_{k-1}y_{k-2}y_{k-3}\ldots y_0$



May 2, 2023 CS151 Lecture 9 8

8

First attempt

- one problem:
  - hard to compute y_{i-1} from y_i
  - but might be easy to compute single bit (or several bits) of y_{i-1} from y_i
  - could use to build small circuit C that distinguishes G's output from uniform distribution on $\{0,1\}^m$

May 2, 2023 CS151 Lecture 9 9

9

First attempt

- second problem
  - we don't know if "unpredictability" given a prefix is sufficient to meet fooling requirement:
    $|Pr_y[C(y) = 1] – Pr_z[C(G(z)) = 1]| \leq \boldsymbol{\varepsilon}$

May 2, 2023 CS151 Lecture 9 10

10

Hard bits

- If $\{f_n\}$ is one-way permutation we know:
  - no poly-size circuit can compute $f_n^{-1}(y)$ from y with non-negligible success probability
    $Pr_y[C_n(y) = f_n^{-1}(y)] \leq \varepsilon'(n)$
- We want to identify a single bit position j for which:
  - no poly-size circuit can compute $(f_n^{-1}(x))_j$ from x with non-negligible advantage over a coin flip
    $Pr_y[C_n(y) = (f_n^{-1}(y))_j] \leq ½ + \varepsilon(n)$

May 2, 2023 CS151 Lecture 9 11

11

Hard bits

- For some specific functions f we know of such a bit position j

- More general:
  function $h_n:\{0,1\}^n \rightarrow \{0,1\}$
  rather than just a bit position j.

May 2, 2023 CS151 Lecture 9 12

12

Hard bits

Definition: hard bit for $g = \{g_n\}$ is family $h = \{h_n\}$, $h_n:\{0,1\}^n \rightarrow \{0,1\}$ such that if circuit family $\{C_n\}$ of size $s(n)$ achieves:

$$\Pr_x[C_n(x) = h_n(g_n(x))] \geq ½ + \varepsilon(n)$$

then there is a circuit family $\{C'_n\}$ of size $s'(n)$ that achieves:

$$\Pr_x[C'_n(x) = g_n(x)] \geq \varepsilon'(n)$$

with:
- $\varepsilon'(n) = (\varepsilon(n)/n)^{O(1)}$
- $s'(n) = (s(n)n/\varepsilon(n))^{O(1)}$

Goldreich-Levin

- To get a generic hard bit, first need to modify our one-way permutation
- Define $f'_n:\{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^{2n}$ as:

$$f'_n(x,y) = (f_n(x), y)$$

Goldreich-Levin

$f'_n(x,y) = (f_n(x), y)$

- Two observations:
  - f' is a permutation if f is
  - if circuit C_n achieves

    $$\Pr_{x,y}[C_n(x,y) = f'^{-1}_n(x,y)] \geq \varepsilon(n)$$

    then for some y^*

    $$\Pr_x[C_n(x,y^*)=f'^{-1}_n(x,y^*)=(f_n^{-1}(x), y^*)] \geq \varepsilon(n)$$

    and so f' is a one-way permutation if f is.

Goldreich-Levin

- The Goldreich-Levin function:

  $$GL_{2n} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$$

  is defined by:

  $$GL_{2n}(x,y) = \oplus_{i:y_i=1} x_i$$

  - parity of subset of bits of x selected by 1's of y
  - inner-product of n-vectors x and y in GF(2)

Theorem (G-L): for every function f, GL is a hard bit for f'. (proof: problem set)

Distinguishers and predictors

- Distribution D on $\{0,1\}^n$
- D **ε**-passes statistical tests of size s if for all circuits of size s:

  $$|\Pr_{y\leftarrow U_n}[C(y) = 1] - \Pr_{y\leftarrow D}[C(y) = 1]| \leq \boldsymbol{\varepsilon}$$

  - circuit violating this is sometimes called an efficient "distinguisher"

Distinguishers and predictors

- D **ε**-passes prediction tests of size s if for all circuits of size s:

  $$\Pr_{y\leftarrow D}[C(y_{1,2,\ldots,i-1}) = y_i] \leq ½ + \boldsymbol{\varepsilon}$$

  - circuit violating this is sometimes called an efficient "predictor"
- predictor seems stronger
- Yao showed essentially the same!
  - important result and proof ("hybrid argument")

Distinguishers and predictors

Theorem (Yao): if a distribution D on $\{0,1\}^n$ (ε/n)-passes all prediction tests of size s, then it ε-passes all statistical tests of size $s' = s - O(n)$.

May 2, 2023 CS151 Lecture 9 19

19

Distinguishers and predictors

- Proof:
  - idea: proof by contradiction
  - given a size s' distinguisher C:

$$|\Pr_{y\leftarrow U_n}[C(y) = 1] - \Pr_{y\leftarrow D}[C(y) = 1]| > \varepsilon$$

  - produce size s predictor P:

$$\Pr_{y\leftarrow D}[P(y_{1,2,\ldots,i-1}) = y_i] > \tfrac{1}{2} + \varepsilon/n$$

  - work with distributions that are "hybrids" of the uniform distribution U_n and D

May 2, 2023 CS151 Lecture 9 20

20

Distinguishers and predictors

- given a size s' distinguisher C:

$$|\Pr_{y\leftarrow U_n}[C(y) = 1] - \Pr_{y\leftarrow D}[C(y) = 1]| > \varepsilon$$

- define n+1 hybrid distributions
- hybrid distribution D_i:
  - sample $b = b_1b_2\ldots b_n$ from D
  - sample $r = r_1r_2\ldots r_n$ from U_n
  - output:

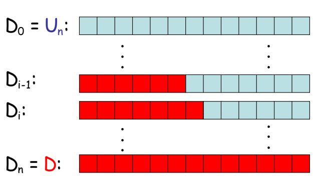
$$b_1b_2\ldots b_i\, r_{i+1}r_{i+2}\ldots r_n$$

May 2, 2023 CS151 Lecture 9 21

21

Distinguishers and predictors

- Hybrid distributions:



May 2, 2023 CS151 Lecture 9 22

22

Distinguishers and predictors

- Define: $p_i = \Pr_{y\leftarrow D_i}[C(y) = 1]$
- Note: $p_0=\Pr_{y\leftarrow U_n}[C(y)=1]$; $p_n=\Pr_{y\leftarrow D}[C(y)=1]$
- by assumption: $\varepsilon < |p_n - p_0|$
- triangle inequality: $|p_n - p_0| \le \Sigma_{1 \le i \le n}|p_i - p_{i-1}|$
- there must be some i for which

$$|p_i - p_{i-1}| > \varepsilon/n$$

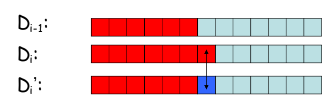
- WLOG assume $p_i - p_{i-1} > \varepsilon/n$
  - can invert output of C if necessary

May 2, 2023 CS151 Lecture 9 23

23

Distinguishers and predictors

- define distribution D_i' to be D_i with i-th bit flipped
- $p_i' = \Pr_{y\leftarrow D_i'}[C(y) = 1]$



- notice:

$$D_{i-1} = (D_i + D_i')/2 \qquad p_{i-1} = (p_i + p_i')/2$$

May 2, 2023 CS151 Lecture 9 24

24

Distinguishers and predictors

- randomized predictor P' for i^{th} bit:
  - input: $u = y_1y_2\ldots y_{i-1}$ (which comes from D)
  - flip a coin: **d** $\in\{0,1\}$
  - $w = w_{i+1}w_{i+2}\ldots w_n \leftarrow U_{n-i}$
  - evaluate C(u**d**w)
  - if 1, output d; if 0, output ¬d

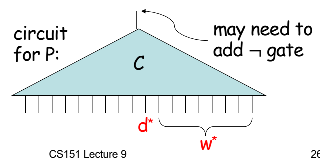
Claim:

$\Pr_{y\leftarrow D, d, w\leftarrow U_{n-i}}[P'(y_{1\ldots i-1}) = y_i] > ½ + \varepsilon/n$.

Distinguishers and predictors

- P' is randomized procedure
- there must be some fixing of its random bits d, w that preserves the success prob.
- final predictor P has d^* and w^* hardwired:

Size is $s' + O(n) = s$ as promised



Distinguishers and predictors

$u = y_1y_2\ldots y_{i-1}$

- Proof of claim:

$\Pr_{y\leftarrow D,d,w\leftarrow U_{n-i}}[P'(y_{1\ldots i-1}) = y_i] =$

$\Pr[y_i = d \mid C(u,d,w) = 1]\Pr[C(u,d,w) = 1]$

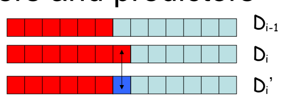
$+ \Pr[y_i = \neg d \mid C(u,d,w) = 0]\Pr[C(u,d,w) = 0]$

$= \Pr[y_i = d \mid C(u,d,w) = 1](p_{i-1})$

$+ \Pr[y_i = \neg d \mid C(u,d,w) = 0](1 - p_{i-1})$

Distinguishers and predictors

$u = y_1y_2\ldots y_{i-1}$



- Observe:

$\Pr[y_i = d \mid C(u,d,w) = 1]$

$= \Pr[C(u,d,w) = 1 \mid y_i = d]\Pr[y_i=d] / \Pr[C(u,d,w) = 1]$

$= p_i/(2p_{i-1})$

$\Pr[y_i = \neg d \mid C(u,d,w) = 0]$

$= \Pr[C(u,d,w)=0 \mid y_i= \neg d]\Pr[y_i=\neg d] / \Pr[C(u,d,w) = 0]$

$= (1 - p_i') / 2(1 - p_{i-1})$

Distinguishers and predictors

- Success probability:

  $\Pr[y_i=d|C(u,d,w)=1](p_{i-1}) + \Pr[y_i=\neg d|C(u,d,w)=0](1-p_{i-1})$
- We know:
  - $\Pr[y_i = d \mid C(u,d,w) = 1] = p_i/(2p_{i-1})$
  - $\Pr[y_i = \neg d \mid C(u,d,w) = 0] = (1 - p_i')/2(1 - p_{i-1})$
  - $p_{i-1} = (p_i + p_i')/2$
  - $p_i - p_{i-1} > \varepsilon/n$

  $p_i'/2 = p_{i-1} - p_i/2$
- Conclude:

  $\Pr[P'(y_{1\ldots i-1}) = y_i] = ½ + (p_i - p_i')/2$

  $= ½ + p_i/2 - (p_{i-1} - p_i/2) = ½ + p_i - p_{i-1} > ½ + \varepsilon/n$.

The BMY Generator

- Recall goal: for all $1 > \delta > 0$, family of PRGs $\{G_m\}$ with
  - output length **m**
  - seed length $\mathbf{t} = m^\delta$
  - error $\boldsymbol{\varepsilon} < 1/6$
  - fooling size $\mathbf{s} = m$
  - running time m^c
- If one way permutations exist then WLOG there is OWP $f = \{f_n\}$ with hard bit $h = \{h_n\}$

The BMY Generator

- Generator $G^\delta = \{G^\delta_m\}$:
  - $t = m^\delta$
  - $y_0 \in \{0,1\}^t$
  - $y_i = f_t(y_{i-1})$
  - $b_i = h_t(y_i)$
  - $G^\delta(y_0) = b_{m-1}b_{m-2}b_{m-3}\ldots b_0$

31

The BMY Generator

Theorem (BMY): for every $\delta > 0$, there is a constant c s.t. for all d, e, G^δ is a PRG with

error $\boldsymbol{\varepsilon} < 1/m^d$

fooling size $\mathbf{s} = m^e$

running time m^c

- Note: stronger than we needed
  - sufficient to have $\boldsymbol{\varepsilon} < 1/6$; $\mathbf{s} = m$

32

The BMY Generator

Generator $G^\delta = \{G^\delta_m\}$:
- $t = m^\delta$; $y_0 \in \{0,1\}^t$; $y_i = f_t(y_{i-1})$; $b_i = h_t(y_i)$
- $G^\delta_m(y_0) = b_{m-1}b_{m-2}b_{m-3}\ldots b_0$

- Proof:
  - computable in time at most

    $mt^c < m^{c+1}$
  - assume G^δ does not $(1/m^d)$-pass statistical test $C = \{C_m\}$ of size m^e:

    $|\Pr_{y \leftarrow U_m}[C(y) = 1] - \Pr_{z \leftarrow D}[C(z) = 1]| > 1/m^d$

33

The BMY Generator

Generator $G^\delta = \{G^\delta_m\}$:
- $t = m^\delta$; $y_0 \in \{0,1\}^t$; $y_i = f_t(y_{i-1})$; $b_i = h_t(y_i)$
- $G^\delta_m(y_0) = b_{m-1}b_{m-2}b_{m-3}\ldots b_0$

- transform this **distinguisher** into a **predictor** P of size $m^e + O(m)$:

  $\Pr_y[P(b_{m-1}\ldots b_{m-i}) = b_{m-i-1}] > ½ + 1/m^{d+1}$

34

The BMY Generator

Generator $G^\delta = \{G^\delta_m\}$:
- $t = m^\delta$; $y_0 \in \{0,1\}^t$; $y_i = f_t(y_{i-1})$; $b_i = h_t(y_i)$
- $G^\delta_m(y_0) = b_{m-1}b_{m-2}b_{m-3}\ldots b_0$

- a procedure to compute $h_t(f_t^{-1}(y))$
  - set $y_{m-i} = y$; $b_{m-i} = h_t(y_{m-i})$
  - compute y_j, b_j for j = m-i+1, m-i+2…, m-1 as above
  - evaluate $P(b_{m-1}b_{m-2}\ldots b_{m-i})$
  - f a permutation implies $b_{m-1}b_{m-2}\ldots b_{m-i}$ distributed as (prefix of) output of generator:

    $\Pr_y[P(b_{m-1}b_{m-2}\ldots b_{m-i}) = b_{m-i-1}] > ½ + 1/m^{d+1}$

35

The BMY Generator

Generator $G^\delta = \{G^\delta_m\}$:
- $t = m^\delta$; $y_0 \in \{0,1\}^t$; $y_i = f_t(y_{i-1})$; $b_i = h_t(y_i)$
- $G^\delta_m(y_0) = b_{m-1}b_{m-2}b_{m-3}\ldots b_0$

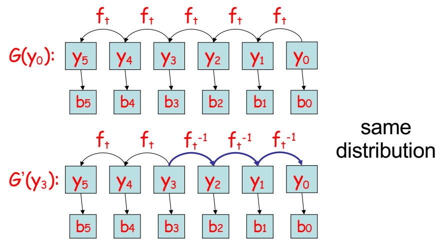
$\Pr_y[P(b_{m-1}b_{m-2}\ldots b_{m-i}) = b_{m-i-1}] > ½ + 1/m^{d+1}$

- What is b_{m-i-1}?

  $b_{m-i-1} = h_t(y_{m-i-1}) = h_t(f_t^{-1}(y_{m-i})) = h_t(f_t^{-1}(y))$
- We have described a family of polynomial-size circuits that computes $h_t(f_t^{-1}(y))$ from y with success greater than ½ + 1/poly(m)
- Contradiction.

36

The BMY Generator



37

Hardness vs. randomness

- We have shown:

  If one-way permutations exist then

  BPP $\subseteq \bigcap_{\delta>0}$ **TIME**(2^{n^δ}) $\subsetneq$ **EXP**

- simulation is better than brute force, but just barely
- stronger assumptions on difficulty of inverting OWF lead to better simulations…

38

Hardness vs. randomness

- Next, we will show:

  If **E** requires exponential size circuits then **BPP = P**

  by building a different generator from different assumptions.

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

39

Hardness vs. randomness

- BMY: for every $\delta > 0$, G^δ is a PRG with

  seed length **t** = m^δ

  output length **m**

  error **ε** < $1/m^d$ (all d)

  fooling size **s** = m^e (all e)

  running time m^c

- running time of simulation dominated by 2^t

40

Hardness vs. randomness

- To get BPP = P, would need t = O(log m)
- BMY building block is one-way-permutation:

  $f:\{0,1\}^t \rightarrow \{0,1\}^t$

- required to fool circuits of size m^e (all e)
- with these settings a circuit has time to invert f by brute force!

  can't get BPP = P with this type of PRG

41

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

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

42