Lecture 25

5/11/2009

Note Title

Today: Randomness & Pseudorandomness

- Definitions & Motivation
- Constructions

Broad Questions

- Randomness is useful in computation!
- But does it really exist?
- Is it cheap?
- Is it trustworthy?

Utility Of Randomness

- Algorithms $BPP \supseteq P$ (seemingly)
- Distributed Computing Synchronization $\Leftrightarrow$ Randomization
- Cryptography Randomness $\Leftarrow$ Secrets.

BPP vs. P

Belief: Probably $BPP = P$

Theorem: [Nisan Wigderson + Impagliazzo Wigderson]

if $\exists L \in DTIME(2^{1000 \cdot n})$ s.t.

$L \notin SIZE(2^{n/10000})$

then $BPP = P$

—x—

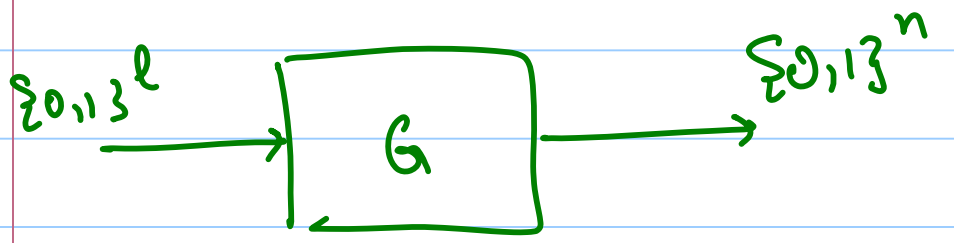
So does such an L exist?

Belief 1: Non-uniform size $\approx$ uniform time, for most problems. So such L should exist.

Belief 2: if it exists, why doesn't $\exists L \in DTIME(2^{cn})$ s.t. $L \notin SIZE(2^n)$?

Pseudorandomness & Derandomization

[Blum, Micali] + [Yao]



G is ϵ-pseudorandom for size s, if

$\forall$ circuits C of size $\leq s$,

$$\Pr_{x \leftarrow U_n}[C(x) = 1] \approx_\epsilon \Pr_{\substack{y = G(z) \\ z \leftarrow \{0,1\}^\ell}}[C(y) = 1]$$

$\longrightarrow$

Ignoring s, ϵ; G is a pseudorandom generator stretching ℓ bits to n bits.

[G polytime computable ...]

Proposition [Yao]: if $\exists$ prg G for linear size circuits stretching $\ell(n)$ bits to n bits (for $\epsilon = \frac{1}{6}$), then $BPP \subseteq \bigcup_{k>0} DTIME(\ell(n^k))$

Corollary: Stretching $O(\log n)$ bits to n bits $\rightarrow$ $BPP = P$.

Initial Definitions / Direction:

- G computable in time, say, n^2 but fools n^{10} size circuits.
- Seems to rely inherently on cryptography/ one-way functions.

[Blum Micali] Construction

$RSA: \{0,1\}^\ell \to \{0,1\}^\ell$

$G_1: \{0,1\}^\ell \to \{0,1\}^{\ell+1}$

$$x \longmapsto (x_1, RSA(x))$$

$x_1 = msb(x)$

$G_k: \{0,1\}^\ell \to \{0,1\}^{\ell+k}$

$$x \longmapsto (x_1, G_{k-1}(RSA(x)))$$

Theorem: RSA is hard to invert for $s = poly$, $\epsilon = \frac{1}{poly}$; implies G_{poly} fools every poly sized circuit with $\epsilon = \frac{1}{poly}$.

Analysis Steps:

Step 1: G_1 is secure.

- Done from scratch

- Depends crucially on MSB, & RSA.

Step 2: G_k is secure.

Hybridization Argument:

Let: $D_i = U_i \times G_{k-i}$

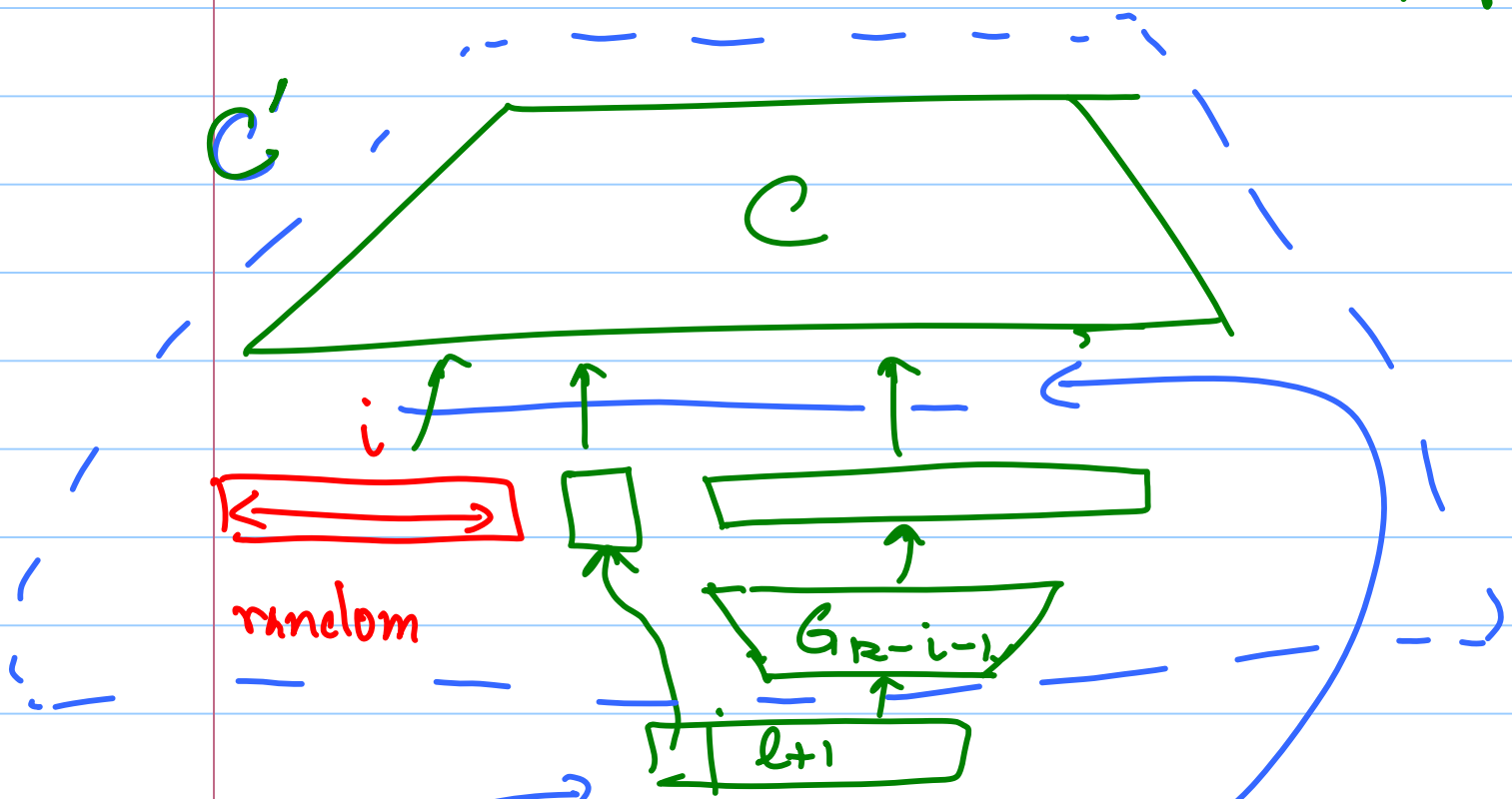
so $D_0 = G_k$; $D_k = U_{k+\ell}$;

if $Pr[C(D_0)=1] > Pr[C(D_k)=1] + \epsilon$

$\Rightarrow$ $\exists$ exists i s.t.

$$Pr[C(D_i)=1] > Pr[C(D_{i+1})=1] + \frac{\epsilon}{k}$$

Will now use C to distinguish $U_{\ell+1}$ from G_1.



if input is $U_{\ell+1}$ then distribution is D_{i+1}

" G_1 " D_i

So C' distinguishes $U_{\ell+1}$ from G_1

Aside: [Goldreich Levin]

Step 1 seems very specific RSA + MSB

(Doesn't work with RSA + LSB e.g.)

More generic construction?

- $f : \{0,1\}^\ell \to \{0,1\}^\ell$

$\Downarrow$

- $\hat{f} : \{0,1\}^\ell \times \{0,1\}^\ell \to \{0,1\}^\ell \times \{0,1\}^\ell$

$(x, r) \longmapsto (f(x), r)$

- f o.w.p. $\iff$ $\hat{f}$ o.w.p.

Clearly $\text{LSB}(x,r)$ insecure (given $\hat{f}(x,r)$)

- But for every f,

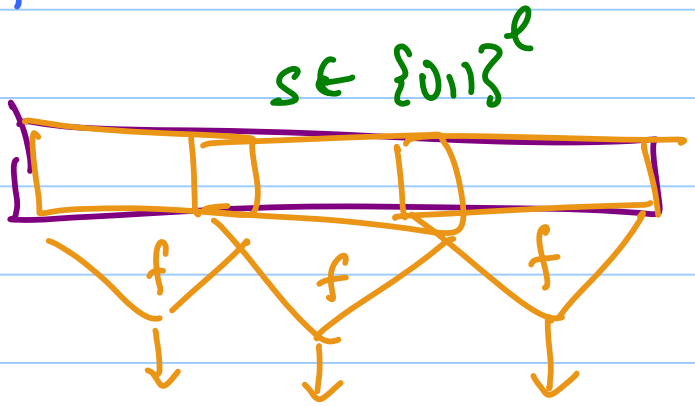
$\langle x, r \rangle = \bigoplus_{i=1}^{\ell} x_i r_i$ secure given $\hat{f}(x,r)$

Back To Derandomizing BPP

- How to do it without crypto assumptions
- [BM] [Y] construction seems to need ability to compute G_{k-i-1} (for some i)
  - So G_k needs to be easy to compute but hard to "spot" (!)?
  - [Nisan Wigderson] Not true definitionally! or for (their) construction!

NW Construction

Basic Idea: Apply hard function f repeatedly to portions of input



But: if $\ell = O(\log n)$

- # application of f exponential in seed length.
- Each bit used in exponentially many applications!
- Analysis of distinguisher still needs to compute $n-1$ instances of f to compute 1 instance of f! How can this be OK?

Key Idea:

- Ensure $(n-1)$ instances of f being computed on small sized instances.
- Small problem have small circuits.

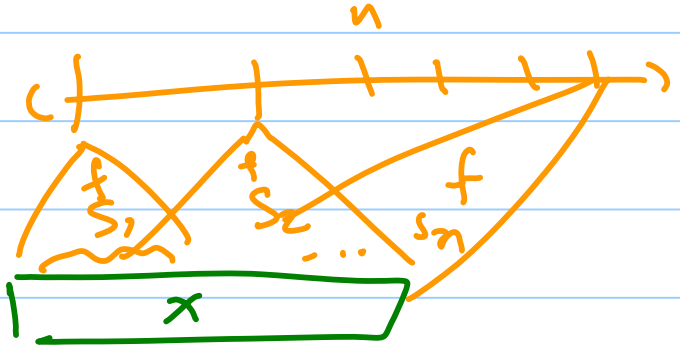
Implementation

- Build sets $S_1, S_2 \ldots S_n \subseteq [\ell]$

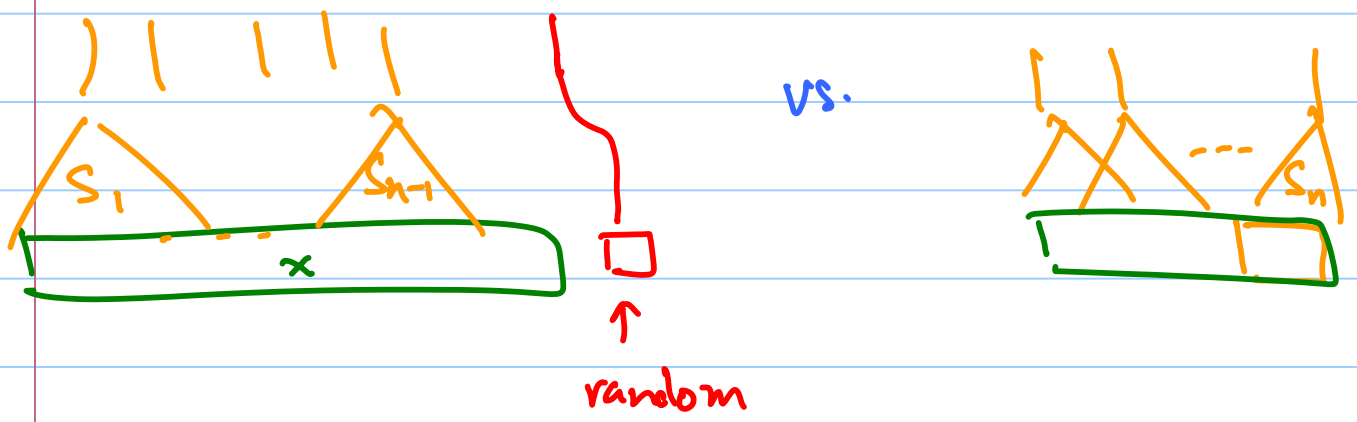
$$|S_i| = m$$

$$|S_i \cap S_j| \leq t$$

- $G(x) = f(x|_{S_1}), f(x|_{S_2}) \ldots f(x|_{S_n})$

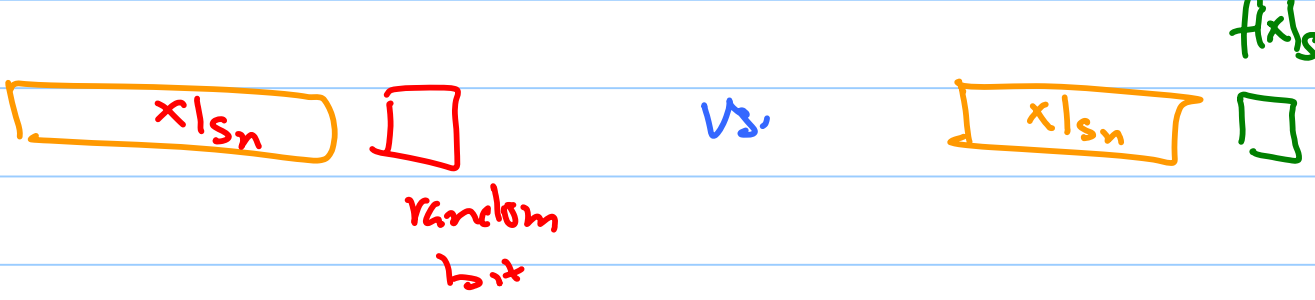


Analysis: Use hybridization & reduce to follow case: $\exists$ C distinguishing



(first n-1 bits from G) (all n bits from G)

will turn this into distinguisher of



Rough sketch

- Fix all bits of x except s_n
- C' can compute $f(x|_{s_i})$ since only t bits are unknown; using circuits of size 2^t.

Final Step: use equivalence between indistinguishability of $(y, f(y))$ & (y, b) $\equiv$ unpredictability of $f(y)$ given y.

Concluding Thoughts

[BM] [Y] + [NW]

Profound Impact on Cryptography & Computational Complexity.

Example: [BM] + [Y]

⇒ Cryptographically strong prg's exist iff one-way functions exist.

[Håstad Impagliazzo Levin Luby].

Example: [Trevisan] NW generator extracts randomness.