Sunday, April 29, 2018 8:54 PM

BPP : $L \in BPP$ if probabilistic poly-time TM M:

if $x \in L$ then $\text{Prob}_y[M(x,y) \text{ accepts}] \geq 2/3$

if $x \notin L$ then $\text{Prob}_y[M(x,y) \text{ accepts}] \leq 1/3$

↳ random string.

We know $P \subseteq BPP$

Question: $P = BPP$?

We will show: If one-way ~~functions~~ permutations exist then

$$BPP \subseteq \bigcap_{\delta} TIME(2^{n^\delta})$$

Family $\{f_n\}$ of one-to-one functions $f_n : \{0,1\}^n \to \{0,1\}^n$ is $(s(n), \epsilon(n))$ - one way if

f_n is poly-time computable.

for every circuit family $\{C_n\}$ such that $|C_n| \leq s(n)$

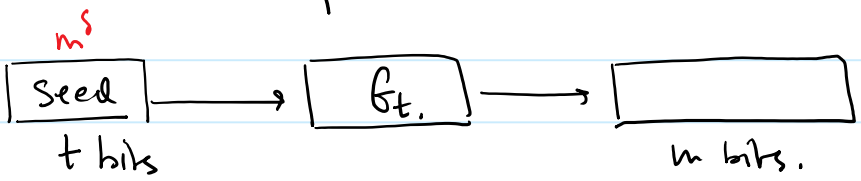
$$Pr_y[C_n(y) = f_n^{-1}(y)] \leq \epsilon(n)$$

Will assume a family of one-to-one functions that are $(p(n), 1/q(n))$ one way for any polynomials $p(n)$, $q(n)$.

x $\quad f(x) = y$. $\quad f^{-1}(y)$.

Monday, May 7, 2018 8:21 AM

Pseudo-random number generator:



A PRG is $(s(t), \epsilon(t))$ indistinguishable from a random string if for any circuit family $\{C_t\}$ $|C_t| \le s(t)$.

$$\left| Pr_y [C_t(y) = 1] - Pr_x [C_t(G_t(x)) = 1] \right| < \epsilon(t)$$

Suppose that for any $0 < \delta < 1$, there is a PRG that stretches m^δ bits to m bits and is $(p(n), 1/6)$ indistinguishable from random for any polynomial $p(n)$

then $BPP \subseteq \bigcap_\gamma TIME(2^{n^\gamma})$

Suppose $L \in BPP$ then TM M

$x \in L \Rightarrow Pr_y [M(x,y) \text{ accepts}] \ge 2/3$

$x \notin L \Rightarrow Pr_y [M(x,y) \text{ accepts}] \le 1/3.$

Input x: estimate $Pr_y [M(x,y) \text{ accepts}]$ by

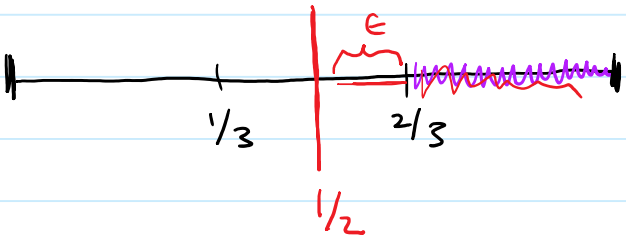
$Pr_z [M(x, G(z)) \text{ accepts}].$ ← Compute this quantity by brute force $TIME(2^{n^\gamma})$.

Monday, May 7, 2018 8:35 AM

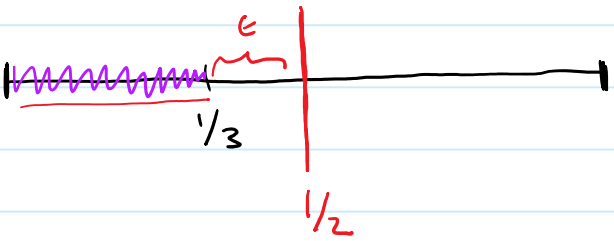
Want:

$$\left| Pr_y[M(x,y) \text{ accepts}] - Pr_z[M(x, G(z)) \text{ accepts}] \right| < 1/6.$$

$x \in L$ $Pr_y[M(x,y) \text{ accepts}] \geq 2/3.$



$x \notin L$ $Pr_y[M(x,y) \text{ accepts}] \leq 1/3.$



Suppose

$$\left| Pr_y[M(x,y) \text{ accepts}] - Pr_z[M(x, G(z)) \text{ accepts}] \right| \geq 1/6.$$

Construct poly-size circuit $C_x(y)$ output $= 1$ iff $M(x,y)$ accepts.

$$\left| Pr_y[C(x,y) = 1] - Pr_z[C_x(G(z)) = 1] \right| \geq 1/6$$

C_x can distinguish between output of G & a random string.

Saturday, April 28, 2018 8:10 PM

We have shown PRG → OWF

For simulating BPP, we would like to go the other way:
Assume a one-way function exists.
Show that a pseudo-random number generator exists.

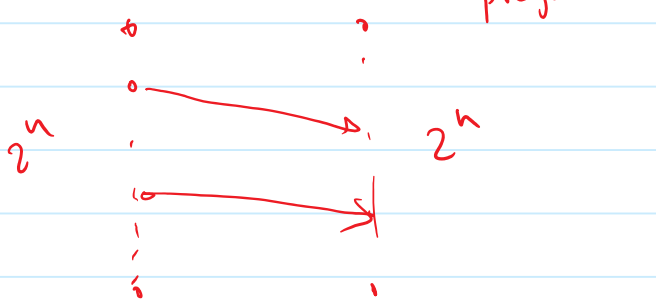
→ This is true but we will show an easier result due to Blum-Micali-Yao that uses a stronger assumption: the existence of a one-way permutation.

Definition: A one-way permutation is a one-way function that is one-to-one.

$$f_n : \{0,1\}^n \to \{0,1\}^n$$

if f_n is one-to-one then $f_n^{-1}(y)$ is unique.

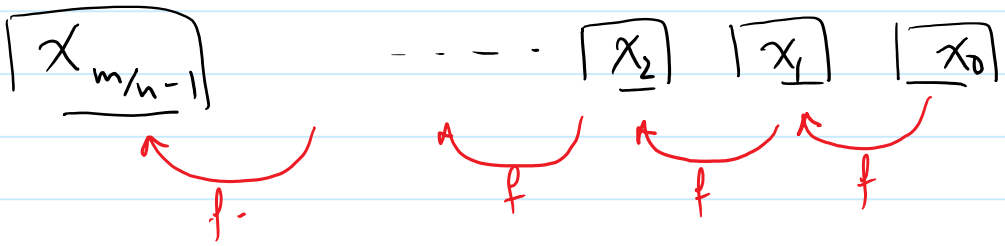
Will assume the existence of a family of functions $\{f_n\}$ $f_n : \{0,1\}^n \to \{0,1\}^n$ that are one-to-one and $(s(n), \epsilon(n))$-one-way for $s(n)$ - super polynomial or for any $s(n)$ polynomial and $\epsilon(n) < 1/\text{poly}(n)$.



Sunday, April 29, 2018 3:42 PM

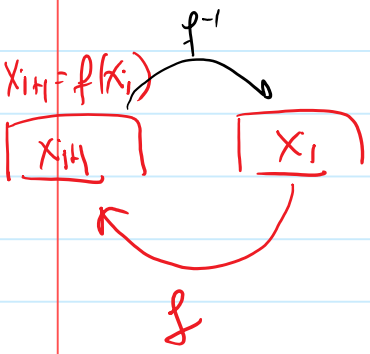
[Want to use one-way perm fn to expand n bits into m bits]

Some intuition: One property of random strings is that given a prefix, it is hard to predict the next bit.



Pick x_0 at random (seed)

Output $x_{m/n-1}\, x_{m/n-2} \cdots x_2\, x_1\, x_0$



$x_{i+1} = f(x_i)$

$f^{-1}(x_{i+1}) = x_i$

x_i is hard to predict from x_{i+1}.

⇓

It's hard to produce <u>all</u> of x_i from x_{i+1}

Perhaps it's easy to produce a few of the bits?

Goal: extract a single bit from x_i that is <u>hard to compute</u> given x_{i+1}.

↳ w.p. bounded away from $\frac{1}{2}$.

Saturday, April 28, 2018 8:10 PM

Hard bits:

If $\{f_n\}$ is a family of functions that are 1-1 and $(s(n), \epsilon(n))$ one-way, then no circuit family $\{C_n\}$ of size $\leq s(n)$ can achieve

$$\Pr_y [C_n(y) = f_n^{-1}(y)] \leq \epsilon(n).$$

Want to identify a single bit of $f^{-1}(y)$ that is hard to compute

no circuit of size $s(n)$ can compute the j^{th} bit of $f^{-1}(y)$ w/ non-negligible advantage over a coin flip.

$$\Pr_y [C_n(y) = (f_n^{-1}(y))_j] \leq 1/2 + \epsilon(n)$$

For some specific functions, we know a bit position j that is hard to compute.

Would like a more generic construction.

$\{h_n\}$ $h_n : \{0,1\}^n \to \{0,1\}$

If $\{f_n\}$ if a family of one-way permutations then

$h_n(f_n(x))$ is hard to compute (instead of the j^{th} bit of x).

$h_n(y)$ is hard to compute (instead of the j^{th} bit of $f^{-1}(x)$).

Saturday, April 28, 2018 8:10 PM

We will be applying this to $g_n = f_n^{-1}$

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

$$\Pr_y[C_n(y) = h_n(g_n(y))] \geq 1/2 + \epsilon(n)$$

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

$$\Pr_y[C_n'(y) = g_n(y)] \geq \epsilon'(n)$$

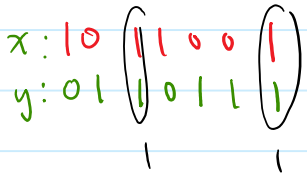
$$\epsilon'(n) = \left(\frac{\epsilon(n)}{n}\right)^{O(1)} \qquad s'(n) = \left(\frac{s(n)\,n}{\epsilon(n)}\right)^{O(1)}$$

In order to get a generic hard bit, we need to modify our one-way permutation

Let $f_n: \{0,1\}^n \to \{0,1\}^n$

Define $f_n': \{0,1\}^{2n} \to \{0,1\}^{2n}$

$$f_n'(x,y) = (f_n(x), y).$$



(1) f_n' is 1-1 iff f_n is 1-1.
(2) f_n' is one-way iff f_n is one-way

Goldreich-Levin function:

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

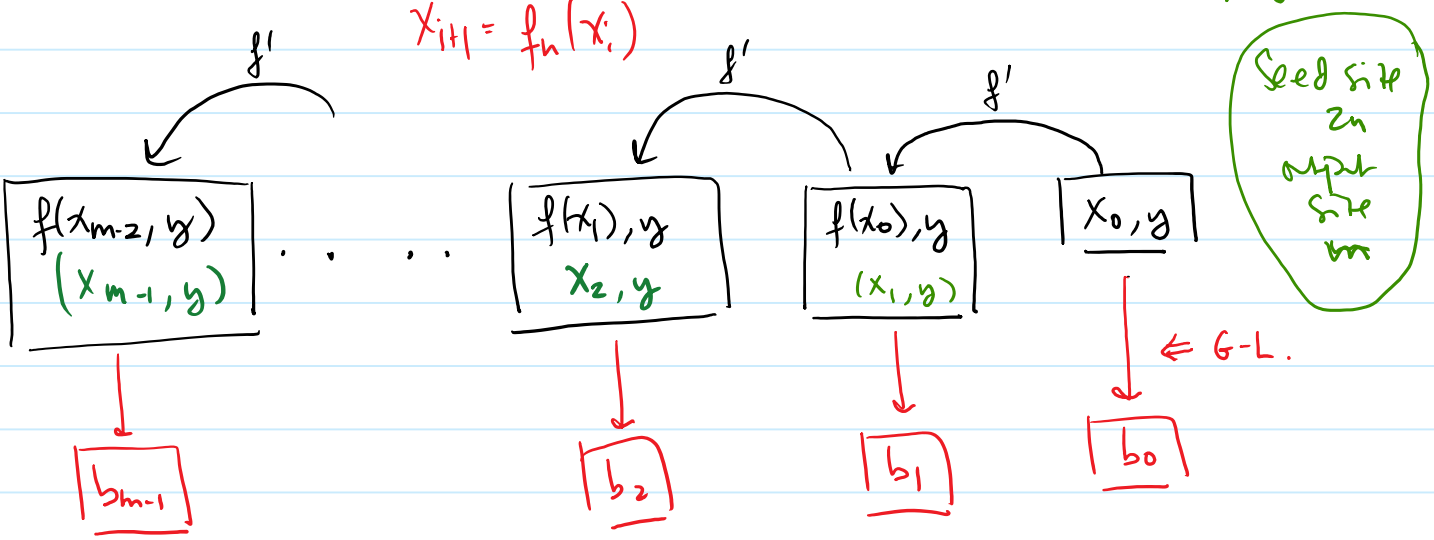
$$GL_{2n}(x,y) = \bigoplus_{j=1}^{n} x_i \wedge y_j$$

inner product of $\vec{x} \cdot \vec{y}$.

Saturday, April 28, 2018 8:10 PM

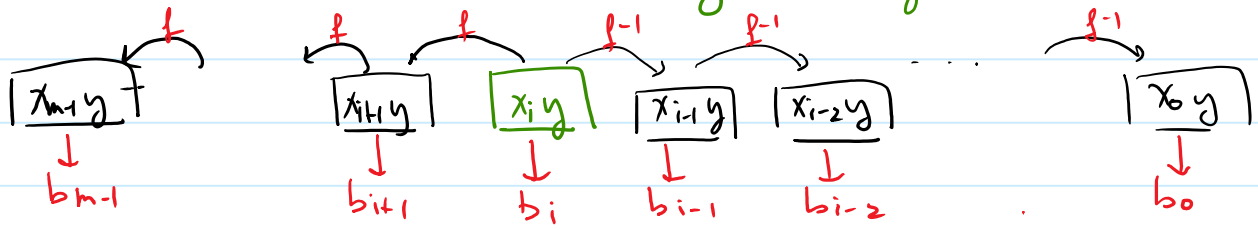
Theorem: For every function f, GL is a hard-bit for f'.

(1) Start w/ {fn} Family of 1-1 functions.
$(s(n), \epsilon(n))$ one-way $s(n)$ super-poly.
$\epsilon(n) < 1/\text{poly}(n)$



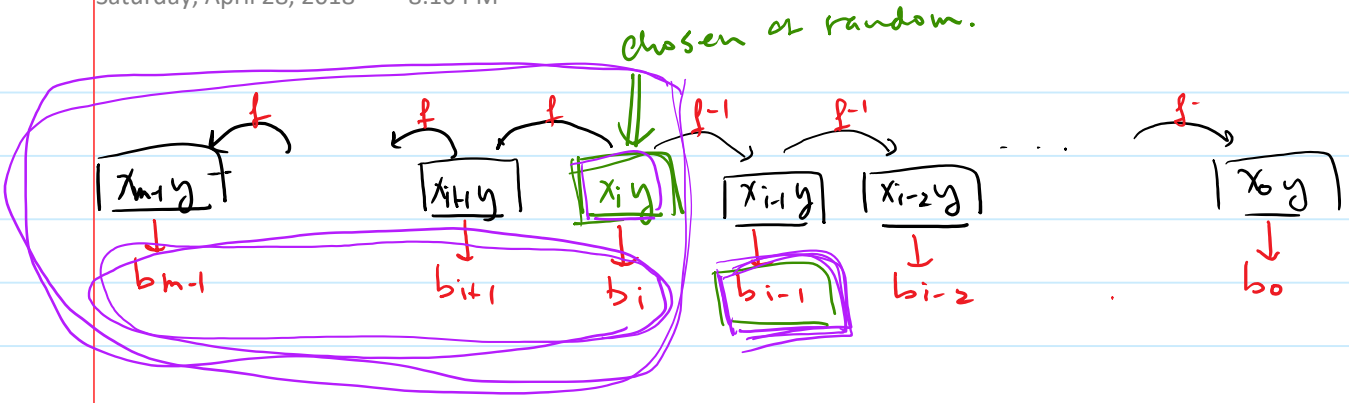
PRG: Seed $(x_0, y) \longrightarrow b_{m-1}\, b_{m-2} \cdots b_1\, b_0$

Since f_n & f_n^{-1} are permutations we could start in the middle & work our way back from either end:



This would be hard to compute but it is well defined.
Distribution over $b_{m-1}\, b_{m-2} \cdots b_0$ is the same.

Saturday, April 28, 2018 8:10 PM



Given (x_i, y) it is hard to compute (x_{i-1}, y) OWF.
Given (x_i, y) it is hard to compute b_{i-1} (Hard bit)

$$\Pr_{(x_i,y)}\left[C(x_i,y) = b_{i-1}\right] \leq \frac{1}{2} + \epsilon.$$

Given (x_i, y) and $b_{m-1} \cdots b_i$ it's still hard to compute b_{i-1}
(because $b_{m-1}, \ldots, b_i$) are easy to compute from (x_i, y).

Given $b_{m-1}, \ldots, b_i$ it's hard to compute b_{i-1}

$b_{m-1} b_{m-2} \cdots b_1$ behaves like a random string
Given a prefix of the string, hard to compute the next bit.

Need to relate predictability to distinguishability.

Want to say no circuit can distinguish $b_{m-1} \cdots b_0$ from a random string.

$$\left| \Pr_{(x_0,y)}\left[C(b_{m-1} \cdots b_1) = 1\right] - \Pr_{z \in \{0,1\}^m}\left[C(z) = 1\right] \right| \leq \epsilon.$$