How do we construct PRG's

want to use "computational hardness"
will start with 1-way functions.

<u>def</u>. f is 1-way if

1) f computable in deterministic ptime
2) $\forall$ ppt A, $\exists$ negligible $\varepsilon(n)$
   st. $\forall n$ big enough

$$\Pr_{x, \text{ coins of } A}\Big[A(f(x)) \in \underbrace{f^{-1}}_{\text{any inverse of } f(x)}(f(x)) \Big] \le \varepsilon(n)$$

<u>notes</u>:

1) A ptime in n, not just $|f(x)|$ (which might be small)
2) don't need to find "right" inverse – just <u>some</u> inverse

Why is this good? We have candidates!

1) $f(x,y) = x \cdot y$ — factoring
2) $f_{m,e}(x) = x^e \bmod m$ — RSA $m = pq$
3) $f_m(x) = x^2 \bmod m$ — Rabin's fctn (square roots mod m)
4) $f_{p,g}(x) = g^x \bmod p$ — discrete log

For PRGs to exist, 1-way fctns must exist:

Claim $f: \{0,1\}^n \to \{0,1\}^{2n}$ PRG $\Rightarrow f$ is o.w.

idea if f not o.w.

$\exists$ inverter algorithm which sometimes works

use it to give statistical test on PRG output

(i.e. output "1" if finds inverse)

- sometimes works on PRG output
- almost never works on truly random bits

But do 1-way fctns imply existence of PRG's?

Yes, but much harder to prove...

Thm [HILL] 1-way fctn exist $\Rightarrow$ PRG's exist ↙ so PRGs exist $\Updownarrow$ 1-w.f.'s exist!

here, a weaker result

Thm 1-way $\underbrace{\text{permutations}}_{\text{fctn that is 1-1 + onto so preimages are unique}}$ exist $\Rightarrow$ PRG's exist

Main Idea

1-way permutation gives "stretch"?

Can we stretch via:

1) $f(x) \circ x$?

No: $T((y, x)) = 1$ iff $y = f(x)$

passes $(f(x), x)$ + fails uniform (z, x)

2) $f(x) \circ x_1$?

Not for all f, what if

$f(x_1 \cdots x_n) = x_1 \circ f'(x_2 \cdots x_n)$?

↖ reveals x_1

so $f(x) \circ x_1 = x_1 \circ f'(x_1 \cdots x_n) \circ x_1$

predictable

3) other?

A candidate way to stretch:

assume there is a hardcore bit

def. $b: \{0,1\}^* \to \{0,1\}$ is hardcore bit for OWF f
if $\forall$ PPT A, $\exists$ negligible $\varepsilon(\ell)$ s.t. } hard to guess it

$$\Pr_{x \in \{0,1\}^\ell}[A(f(x)) = b(x)] \leq \frac{1}{2} + \varepsilon(\ell)$$

computation time?
given x, poly
given $f(x)$ hard

Use hcb to get 1-bit of stretch: Prg maps n bits to $n+1$ bits

Thm if b is hcb for a OWP f
then $G = f(x) \circ b(x)$ is a PRG

An example: here is candidate hcb:

assume factoring hard (even products of 2 primes)
p, q prime
$p = q = 3 \mod 4$
$N = pq$
$x \in Z_N^*$
$f(x) = x^2 \mod N \leftarrow$ is 1-way?

Candidate hcb of f: $\mathrm{lsb}(x)$
PRG: $(f(x), \mathrm{lsb}(x))$
$(x^2 \mod N, \mathrm{lsb}(x))$

Proof idea for theorem:

$x \in_r U \Rightarrow f(x) \in_r U$ so bits of f are i.i.d + unpredictable

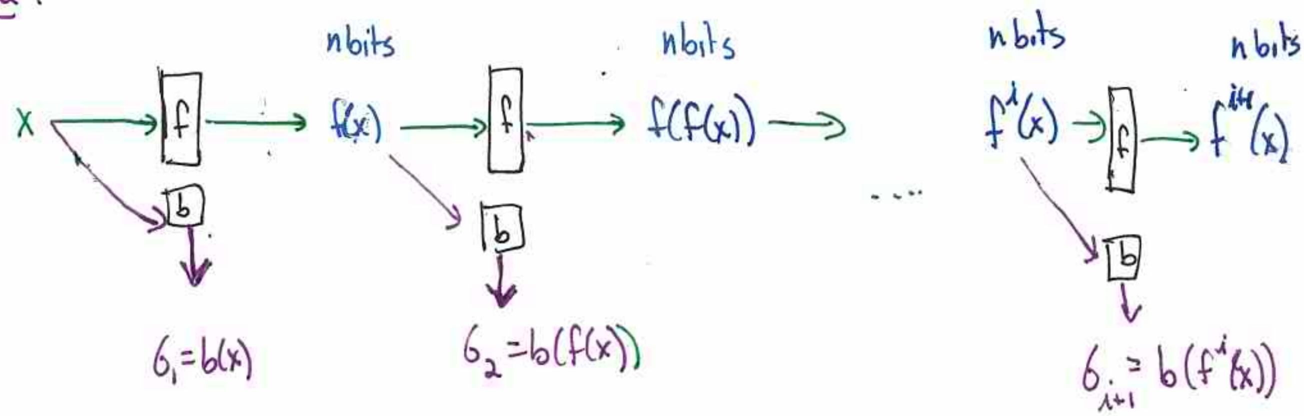
use that f is permutation

$b(U_l)$ unpredictable from $f(U_l)$ since b is h.c.
$\le \frac{1}{2} + \varepsilon(l)$

$\therefore$ $(f(x), b(x))$ is n.b.u $\Rightarrow$ P.R. ∎

How do we get more "stretch"?
i.e. map m bits to n bits for $n = m+k$?

idea:



idea: output $\sigma_k \cdots \sigma_1$

since if see $f^{(i)}$ can't guess $f^{(i-1)}$, much less its hcb

so hard to predict

even if predict $f^{(i-1)}$, can't guess $f^{(i-2)}$ (+ its hcb) ...

actually, this argument uses "order", but we know order of bits doesn't matter! e.g. can output $\sigma_1 \cdots \sigma_k$

<u>Thm</u> if $f: \{0,1\}^\ell \to \{0,1\}^\ell$ is OWP with efficiently computable hcb b then

$$G(x) = b(f^{(n-1)}(x)) \circ \ldots \circ b(f(f(x))) \circ b(f(x)) \circ b(x)$$

is PRG $\forall\ n = poly(\ell)$

<u>Pf</u>

Assume not PRG $\Rightarrow$ not n.b.u.

i.e. $\exists$ ppt P st.

$$\underbrace{\Pr_{x,i}\left[P(b(f^{(n-1)}(x)) \ldots b(f^{(n-i+1)}(x))) = b(f^{n-i}(x))\right]}_{*} - \frac{1}{2} \geq 1/n^k$$

note! can we remove i here by averaging?

$*$ depends on r.v. x, i

$$\boxed{\text{set } y = f^{(n-i)}(x)}$$

since $x \in_r \{0,1\}^\ell$ & f is permutation $\Rightarrow y \in_r \{0,1\}^\ell$

$$\therefore \quad \overbrace{\Pr_{y,i}\left[P(b(f^{(i-1)}(y)) \ldots b(f^{(1)}(y)) = b(y)\right] - \frac{1}{2}}^{*} \geq \frac{1}{n^k}$$

"change of basis"

Define $A(z)$ (computes when $z = f(y)$)

1. $i \in_r \{1..n\}$
2. output $P(\underbrace{b(f^{(i-2)}(z))\, b(f^{(i-3)}(z)) \ldots b(z)}_{\text{requires many ptime computations of } b, f, \text{ z}})$

$$\text{So} \quad \underbrace{\Pr_{i,y}[A(f(y)) = b(y)] - 1/2 \geq 1/n^k}_{*}$$

use that y is distributed uniformly since f is permutation.

($\Rightarrow \exists i$ s.t. $\Pr[A(f(y)) = b(y)] - \frac{1}{2} \geq \frac{1}{n^k}$)

$\Rightarrow$ b not hardcore!

$\blacksquare$

Where do we get a hard core bit?

Thm [Goldreich Levin]

If f is a OWF then $b(x,r) = \langle x, r\rangle$ ← 2 input fctn sometimes convenient to think of it as r fixed, x varies other times r varies, x fixed

is hcb for OWF $\boxed{f'(x,r) = (f(x), r)}$ ← r is 2nd input of 1-way fctn but just "passed" through + "announced"

$|x| = |r|$

Previously, specific 1-way fctns with some hcb

i.e. "discrete log is very discreet"

this works with <u>any</u> 1-way fctn

<u>Pf.</u> [of [GL] for OWP + f s.t. $f: \{\pm 1\}^n \to \{\pm 1\}$]

assume $b(x,r)$ not h.c.

then $\exists A$ s.t. $\Pr_{x, r, S = \underbrace{A's}_{\substack{\text{coins} \\ = \phi}}} [A_s(f(x), r) = \langle x, r\rangle] \geq \frac{1}{2} + \varepsilon$ (*)

↑ deterministic ckt (randomness doesn't help in ckt model)

$h_x(r) \equiv A(f(x), r)$

<u>Plan</u> show can recover many x's ⇒ f' not 1-way $\overset{\text{trivial}}{\Rightarrow}$ f not 1-way

"good" x : x s.t. $\Pr_r [h_x(r) = \langle x, r\rangle] \geq \frac{1}{2} + \frac{\varepsilon}{2}$

how many good x ? $\geq \varepsilon/2$

why so many good x?

if not

$$\Pr_{x,r}[A(f(x),r)=\langle x,r\rangle] < \underbrace{\frac{\varepsilon}{2}}_{\text{upper bnd on fraction good } x} \cdot \underbrace{1}_{\text{trivial bnd on how good they can be}} + \underbrace{1}_{\text{trivial upper bnd on fraction \# bad } x} \cdot \underbrace{\left(\frac{1}{2}+\frac{\varepsilon}{2}\right)}_{\text{upper bnd on bad } x}$$

$$< \frac{1}{2}+\varepsilon$$

contradicts (*)

$\rightarrow\leftarrow$

So we have a fctn:

$$r \longrightarrow \boxed{h \text{ with unknown } x} \longrightarrow h(r)$$

$= \underbrace{\langle x,r\rangle}$ for $\geq \frac{1}{2}+\frac{\varepsilon}{2}$ fraction of r's

has 1 nonzero Fourier coeff ~~others~~ + it has value 1

i.e. h agrees with linear fctn $\frac{1}{2}+\frac{\varepsilon}{2}$ inputs

so h has Fourier coeff corresponding to r which is large! (i.e. $\geq \varepsilon$)

- Can query $\underset{\equiv A(f(x),r)}{h}$ on any r w/o knowing x

  have access to $A_{f(x)}^{r}$ (ptime)

- find all z s.t. : $\Pr[h(r)=\langle z,r\rangle] \geq \frac{1}{2}+\underbrace{\frac{\varepsilon}{2}}_{\text{use } \varepsilon/4}$

  for each, check via f if it works!

$\blacksquare$

A lemma for next time:

def. $\mathcal{I} = \{I_1 \cdots I_m\} \subseteq [\ell]$ is (ℓ, n, d)-design $(\ell > n > d)$

if 1) $|I_j| = n \quad \forall j$

2) $|I_j \cap I_k| \leq d \quad \forall j \neq k$

Thm. $\exists$ algorithm running in $2^{O(\ell)}$ time s.t. for $n > d$, $\ell > 20n^2/d$ which outputs (ℓ, n, d)-design s.t. $m = 2^{d/10}$

Pf.

Greedy - best parameters, use prob method to show can progress:

GreedyAlg: after have $I_1 \cdots I_\ell$ for $\ell < 2^{d/10}$ search all subsets to find I^* s.t. $|I^* \cap I_j| \leq d \quad \forall j \in [\ell]$

runtime: $\text{poly}(m) \cdot 2^{\ell}$

Why doesn't it get stuck?

if pick I^* randomly: prob $x \in [\ell]$ gets chosen $= \frac{2n}{\ell}$ (and truncate later)

$E[|I^*|] = 2n \longrightarrow \Pr[|I^*| \geq n] \geq 0.9$ } chernoff

$E[|I^* \cap I_j|] = \frac{4n^2}{\ell} < \frac{d}{5} \longrightarrow \Pr[|I^* \cap I_j| \geq d] \leq \frac{1}{2} \cdot 2^{-d/10}$

Since $m < 2^{d/10}$, via union bnd, with prob ≥ 0.4 I^* will be good. ∎