Today: Hardness vs. Randomness
An interesting combinatorial lemma:

\[ \mathcal{I} = \{ I_1, \ldots, I_m \} \subseteq [l] \] is an \((m, l, n, d)\)-design \((l > n > d)\)

if

1) \(|I_j| = n \quad \forall j\)

2) \(|I_j \cap I_k| = d \quad \forall j \neq k\)

Thm 3: Algorithm running in \(O(\log n)\) time \(\forall l\)

for \(n > d, l > 10n^2\)

which outputs \((m, l, n, d)\)-design s.t. \(m = 2\)

Pf. Idea: Greedy algorithm

Show can always make progress after have picked \(I_1, \ldots, I_l\) for \(l < \frac{d}{10}\)

Search all subsets to find \(I^*\)

s.t.: \(|I^* \cap I_j| = d \quad \forall j \in [l]\)

Runtime: \(\text{poly}(m) \cdot 2^l\)
why doesn't it get stuck? probabilistic proof

if pick $I^*$ randomly,

st. prob $x \in U$ gets chosen with prob

\[
\frac{2n}{l}
\]

(if $I^*$ too big, truncate to

size $n$ later on)

\[
E[I^*] = l \cdot \frac{2n}{l} = 2n
\]

\( \Rightarrow \) \( Pr[|I^*| = n] \geq 0.9 \) \( \implies \) \( \text{Chernoff} \)

\[
E[|I^* \cap I_j|] = n \cdot \frac{2n}{l} \cdot \frac{2n}{l} < \frac{d}{5}
\]

by \( l \geq \frac{10n^2}{d} \)

\( \implies \) \( Pr[|I^* \cap I_j| = d] \leq \frac{1}{2} \cdot 2^{-d/10} \) \( \text{Chernoff} \)

\[
Pr[\forall j \text{ } |I^* \cap I_j| = d + \text{ } |I^*| = n]
\]

\( \geq 1 - (0.1 + m \cdot \frac{1}{2} \cdot 2^{-d/10}) \geq 0.4
\]

\( \implies \) \( Pr[I^* \text{ good}] \geq 0.4 \)
Derandomization

\[ \text{X} \xrightarrow{\text{PRG}} \text{Random bits} \xrightarrow{\text{Program}} \text{Output} \]

Previously: derandomized programs using $k$-wise independent random bits
Today: general programs

PRG outputs bits that look random to any time $t$ algorithm

More generally, hard on average $f(n)$
(for programs running in time $t(n)$ get advantage $\leq \frac{1}{\sqrt{t(n)}}$)

$\Rightarrow$ PR on programs run in time $\leq t(n)$
adv $\leq \frac{1}{10}$ or some other parameters?
Some definitions:

**Def: "Pseudorandom"**

Let $X_n$ be a sequence of r.u.'s on $\{0,1\}^n$

$X_n$ is $(\epsilon,\delta)$- p.r. if $A$ probabilistic

TM's running in time $\leq t(n)$

$|Pr \left[ T(X_n)=1 \right] - Pr \left[ T(U_n)=1 \right]| \leq \delta(n)$

**Def: $f: \{0,1\}^d \rightarrow \{0,1\}$ is $(\epsilon,\delta)$- average case hard if $A$ in time $t(l)$

$Pr_{x \in \{0,1\}^* \text{ of } A} \left[ A(x) = f(x) \right] \leq \frac{1}{2} + \epsilon(\delta)$ for large enough $l$

$\leq \frac{1}{2} + \frac{1}{t(l)}$

**Def: $f$ is $t$-average case hard if $\text{adv}(A)$ is $\frac{1}{t(l)}$**

for nonuniform $A$ in time $t(l)$

ckt complexity of $A$
Warmup:

**Theorem**

If $f: \mathbb{Z}^* \rightarrow \mathbb{Z}$ is $(t, \varepsilon)$-average case hard,

then $G(y) = y \circ f(y)$ is PRG

$l$ bits $\rightarrow$ $l+1$ bits

extends by 1 bit

Unpredictability:

**Definition**

$X = x_1 \cdots x_n$ is "next bit unpredictable" (nbu) with parameters $(t(n), \varepsilon(n))$ if

$\forall P$ using $\leq t(n)$ time

$\Pr_{x, x_i \in \{0,1\}} \left[ P(x_1 \cdots x_{i-1}) = x_i \right] \leq \frac{1}{2} + \varepsilon(n)$

Cool theorem:

nbu $\iff$ pr are equivalent (up to parameters)

more specifically:

1) if next bit $i$ can be predicted

$\Pr_{x, x_i \in \{0,1\}} \left[ P(x_1 \cdots x_{i-1}) = x_i \right] \geq \frac{1}{2} + \frac{1}{n}k$

then $\exists$ statistical test $T$ which distinguishes $X$ from $U$ within $\frac{1}{n}k$
2) if $\mathcal{F}$ distinguishing test for $X$ from $U$ with advantage $\frac{1}{n^k}$
then can predict with advantage $\frac{1}{n^{k+1}}$ in $t+O(n)$ steps

Proof "idea" for $G(y^* yof(y))$ is a PRG-

$y \in U \Rightarrow y$ is n.b.u.

since $f(y)$ is hard for any
$P$ using $\leq t(n)$ steps

$\Rightarrow$ any program on $t(n)$ steps
predicts $f(y)$ with
adv $\leq \frac{1}{t(n)}$

$\Rightarrow yof(y)$ is n.b.u
with parameter $\frac{1}{t(n)}$

$\Rightarrow yof(y)$ is p.r.

How do you get $\geq 1$ bit stretch?
Nisan Wigderson Generator: (Given $f$)
\[
\begin{aligned}
|I_j| &= n \\
|I_j \cap I_k| &\leq d \\
|I_j| &\leq n = d
\end{aligned}
\]

Given $(l,n,d)$ design $Y = \{I_1, \ldots, I_m\} \subseteq \{l\}$

$G : \{0,1\}^l \to \{0,1\}^m$

is $G(x) = f(x|_{I_1}) \circ f(x|_{I_2}) \circ \cdots \circ f(x|_{I_m})$

String of length $n$
Selecting bits indexed by $I_i$

$x$:

New notation:

$f(x) = f(x|_{I_x})$
Thm. If \( \exists f : \{0,1\}^n \rightarrow \{0,1\} \) s.t.

\[ T \in \text{TIME}(2^n) \]

\[ f \in \text{is } \text{t-arc case hard} \]

(2) \( \{l, m, d\} \) design with \( m \) sets

constructable in time \( 2^{O(l)} \leq t(l)/2 \)

\[ m = \frac{3}{10} \]

\[ l > 10n^2 \]

\[ n > d \]

\[ = t(l)^c \]

\[ \text{e.g. } c = 2 \]

then \( \mathcal{G} \) is \( \frac{\epsilon}{m} \)-PRG against nonuniform time \( m \)

\[ \text{think of } \epsilon = \frac{1}{10} \]

Pf.
If $G$ not $\frac{1}{n}$-PRG against time $m$

\exists n.b. predictor $P$ s.t,

\[
Pr_{i,x}\left[P\left(f_1(x), f_2(x), \ldots, f_{m-1}(x)\right) = f_m(x)\right] \geq \frac{1}{2} + \frac{\epsilon}{m}
\]

\[\uparrow \text{time prog predictor}\]

will use to compute $f$

with $\frac{\epsilon}{m}$ advantage in $O(t(m))$ time

where $m \propto t^{1/c}$

$t \propto m^c$

As usual:

\begin{align*}
\text{averaging} & \implies \exists x^* \text{ s.t. achieve expectation} \\
\text{averaging} & \implies \exists \text{ choice of bits of } x^* \\
\text{call it } z & \text{ not in } \overline{I_{x^*}} \text{ achieving expectation} \\
\overline{I_{x^*}} & \sim \overline{I_{x^*}}
\end{align*}

notation: $Y \subset X$ with bits in $\overline{I_{x^*}}$ set to $z$

and other bits $I_{x^*}$ set randomly
\[ \Pr_y \left[ P(f_1(y), f_2(y), \ldots, f_{\lambda-1}(y)) = f_{\lambda^*}(y) \right] \geq \frac{1}{2} + \frac{\varepsilon}{m} \]

- Each depend on \( \leq d \) bits of \( y \)
- Since \( |I_{\lambda^*} \cap I_j| \leq d \) for all \( j \)
- Since \( f \in \mathcal{E} \) can compute each \( f_j \) in time \( \leq 2^d \)

\[ A(y) = P(f_1(y), f_2(y), \ldots, f_{\lambda^*}(y)) \]

Predicts \( f_{\lambda^*}(y) \) with adv \( \geq \frac{\varepsilon}{m} = \frac{1}{10} \cdot 2^{d/10} \)

Runtime \( \tilde{O}(2^d) \times O(m) + \frac{t(n)}{2} \)

Set \( d \approx \log t \)

\[ 2^d \approx \tilde{O}(t^{10}) \]

Total time: \( 2^d \cdot O(m) + O(m) = \tilde{O}(t^{10}) \cdot O(t^{10}) + \varepsilon \)

Contradicts hardness of \( f \)