Pseudorandom generators (prg)

- Expanders helped in derandomizing a specific problem in RL.

- Prgs are objects to derandomize more general randomized algorithms.

Definition: • A distribution R over $\{0,1\}^m$ is (s, ε)-pseudorandom if $\forall$ circuits C of size $\leq s$,

$$\left| \Pr_{x \in R}[C(x)=1] - \Pr_{x \in U_m}[C(x)=1] \right| < \varepsilon.$$

(U_m ← uniform distribution)

(This measures how well can C distinguish R from U_m. It has cryptographic origins.)

• Let $S: \mathbb{N} \to \mathbb{N}$ be a function. A $2^{O(n)}$-time computable function $G: \{0,1\}^* \to \{0,1\}^*$ is an → S-prg if $\forall \ell \in \mathbb{N}$,

$G: \{0,1\}^\ell \to \{0,1\}^{S(\ell)}$ &

$G(U_\ell)$ is $(S(\ell)^3, 0.1)$-pseudorandom.

S is the stretch

Prgs derandomize classes

$S, \ell : \mathbb{N} \to \mathbb{N}$ are poly-time computable & nondecreasing.

Lemma: If there exists an S-prg then $\forall$ function ℓ, $\mathrm{BPtime}(S \circ \ell(n)) \subseteq \mathrm{Dtime}(2^{O(\ell(n))} \cdot S \circ \ell(n))$.

Proof:

- Idea is to use an S-prg G as the source of (pseudo-) random bits in the randomized algorithm.

- A language $L \in \mathrm{BPtime}(S \circ \ell(n))$ if $\exists$ algorithm M that on input $x \in \{0,1\}^n$ uses $m = O(S \circ \ell(n))$ random bits r & runs for time $O(S \circ \ell(n))$ s.t.

$$\Pr_{r \in U_m}[M(x,r) = L(x)] \geq 3/4.$$

- The derandomization idea is to use an S-prg G to produce r's:
- On input x, our deterministic algorithm B will go over all $z \in \{0,1\}^{\ell(n)}$, compute $M(x, G(z))$ & output the majority vote.

- We claim that $\Pr_{z \in U_{\ell(n)}}[M(x, G(z)) = L(x)] \geq$

$\frac{3}{4} - 0.1 > 1/2$, thus, B correctly solves L.

- Suppose not, then $\Pr_{z \in U_{\ell(n)}}[M(x, G(z)) = L(x)]$

$< \frac{3}{4} - 0.1$.

$\Rightarrow \left| \Pr_{z \in U_{\ell(n)}}[M(x, G(z)) = L(x)] - \Pr_{r \in U_m}[M(x, r) = L(x)] \right|$

$> \left| \left(\frac{3}{4} - 0.1\right) - \left(\frac{3}{4}\right) \right| = 0.1$

- Now consider the circuit C_x that on input $y \in \{0,1\}^{s_0\ell(n)}$ outputs 1 iff $M(x, y) = L(x)$.
- Since M is $O(s_0\ell(n))$-time we get a (lazy) size bound of $O(s_0\ell(n))^2$ for C_x. (Exercise)

$\Rightarrow$ C_x distinguishes $G(U_{\ell(n)})$ from $U_{s_0\ell(n)}$ well, contradicting the fact that G is a S-prg.

$\Rightarrow$ B is a det. algo. solving L in $O(2^{\ell(n)} \cdot s_0\ell(n))$-time

□

- By picking various stretch fns S, we get the following <u>conditional</u> derandomizations:

<u>Corollary:</u> (i) If $\exists\ 2^{\varepsilon \ell}$-prg, for some $\varepsilon > 0$, then BPP = P.
↑ exponential stretch

(ii) If $\exists\ 2^{\ell^\varepsilon}$-prg, for some $\varepsilon > 0$, then
↑ subexponential stretch
$BPP \subseteq Dtime(2^{poly\text{-}lg(n)}) =: QuasiP.$

(iii) If $\forall c > 1$, $\exists\ \ell^c$-prg then $BPP \subseteq \bigcap_{\varepsilon > 0} Dtime(2^{n^\varepsilon}) =: Subexp.$
↑ polynomial stretch

<u>Proof:</u>

(i) Apply the lemma on $S: \mathbb{N} \to \mathbb{N}$; $n \mapsto 2^{\varepsilon n}$ & $\ell: \mathbb{N} \to \mathbb{N}$; $n \mapsto c \cdot \lg n$, for $c \in \mathbb{N}$.

(ii) On $S: \mathbb{N} \to \mathbb{N}$, $n \mapsto 2^{n^\varepsilon}$ & $\ell: \mathbb{N} \to \mathbb{N}$, $n \mapsto c \cdot (\lg n)^{1/\varepsilon}$, $c \in \mathbb{N}$.

(iii) On $S: \mathbb{N} \to \mathbb{N}$, $n \mapsto n^c$ & $\ell: \mathbb{N} \to \mathbb{N}$, $n \mapsto n^\varepsilon$ for $c \in \mathbb{N}$ & $\varepsilon \in (0,1)$.

□

- How do we construct these prgs?

The only known way is to exploit the hardness (conjectured) of problems!

Hardness & prgs

- We define two types of hardness of boolean functions.

Definition: • For $f: \{0,1\}^* \to \{0,1\}$, the **average-case hardness** $H_{avg}(f)$ is the largest $S(n)$ s.t. $\forall$ circuit $C_n \in Size(S(n))$,

$$\Pr_{x \in U_n}[C(x) = f(x)] < 1/2 + 1/S(n).$$

hard ⇑ large size & less advantage

• **Worst-case hardness** $H_{wrs}(f)$ is the largest $S(n)$ s.t. $\forall$ circuit $C_n \in Size(S(n))$,

$$\Pr_{x \in U_n}[C(x) = f(x)] < 1.$$

▷ $H_{avg}(f) \leq H_{wrs}(f) < 2^n$.

- Counting methods show that "usually" $H_{wrs}(f) = 2^{\Omega(n)}$.

But we do not know of "natural" f with super-polynomial hardness! or explicit

- The conjectured f, of cryptographic significance, are:

(1) $H_{wrs}(3SAT) = 2^{\Omega(n)}$?

(2) $H_{avg}(Int\text{-}Fact) = n^{\omega(1)}$?

bits in a complete factorization

R for n-bit input

- We will later prove that a worst-case hard function gives rise to an average-case one.

The tool would be local list decoding of linear error-correcting codes.

- For now, we relate average-case hardness to derandomization.

Hardness ⇒ Prg

Theorem (Nisan, Wigderson, 1988): If $\exists$ $f \in E$ with $H_{avg}(f) \geq S(n)$ then $\exists$ an $S'(\ell)$-prg where

$S' \approx S$ { $S'(\ell) := S(n)^{0.01}$ for $\frac{100n^2}{\log S(n)} < \ell \leq \frac{100(n+1)^2}{\log S(n+1)}$.

Proof:

- Idea — We stretch a seed $z \in \{0,1\}^\ell$ to $\{0,1\}^{S'(\ell)}$ by choosing n-sized subsets $I_1, \ldots, I_m \subseteq [\ell]$ & considering $f(z_{I_1}) \circ f(z_{I_2}) \circ \cdots \circ f(z_{I_m})$.

little overlap → / hard to guess the next bit →

- **Definition**: Let $\mathcal{I} := \{I_1, \ldots, I_m\}$ be a family of n-sized subsets of $[\ell]$ & let $f: \{0,1\}^n \to \{0,1\}$.

The **$(\mathcal{I}, f)$-NW generator** is the function $NW_f^{\mathcal{I}} : \{0,1\}^\ell \to \{0,1\}^m$ s.t. $\forall z \in \{0,1\}^\ell$,

$$NW_f^{\mathcal{I}}(z) := f(z_{I_1}) \circ \cdots \circ f(z_{I_m})$$

where z_I is the restriction to the coordinates I.

- For an $(\mathcal{I}, f)$-NW generator to be pseudo-random, it suffices that, f should be hard & $\mathcal{I}$ should be a certain <u>design</u>:

<u>Definition</u>: Let $\ell > n > d$. A collection $\mathcal{I} = \{I_1, \ldots, I_m\}$ of n-sized subsets of $[\ell]$ is an <u>(ℓ, n, d)-design</u> if $|I_j \cap I_k| \leq d$ for all $j \neq k \in [m]$.

<u>Lemma 1</u> (designs): $\exists$ algorithm A that on input (ℓ, n, d), where $\ell > 10n^2/d$, outputs an (ℓ, n, d)-design $\mathcal{I}$ having $m \geq 2^{d/10}$ subsets, in time $2^{O(\ell)}$.

<u>Proof</u>:

- Idea - Greedily build $\mathcal{I}$.
- Initialize $\mathcal{I} = \phi$.

(1) Say, $\mathcal{I} = \{I_1, \ldots, I_m\}$ with $m < 2^{d/10}$.
Find an $I \in \binom{[\ell]}{n}$ s.t. $\forall j \in [m]$, $|I \cap I_j| \leq d$.

(2) $\mathcal{I} \leftarrow \mathcal{I} \cup \{I\}$ & goto (1).

- Clearly this takes time $< (2^{d/10})^2 \times 2^{\ell} \cdot n = 2^{O(\ell)}$.

- Can it get stuck at $m < 2^{d/10}$?

We show the existence of I by the probabilistic method.

- Build I by picking each $x \in [\ell]$ with probability $2n/\ell$.

$\Rightarrow$

$$E[\#I] = \sum_{x \in [\ell]} \frac{2n}{\ell} = 2n.$$

& $\forall j \in [m]$, $E[|I \cap I_j|] = \sum_{x \in I_j} \frac{2n}{\ell} = \frac{2n^2}{\ell} < \frac{d}{5}$.

- Thus, by Chernoff bounds: $\Pr_I[|I| < n] < 2 \cdot e^{-n/8}$

$\Pr[|\sum X_i - \mu| \geq c\mu]$
$\leq 2 \cdot \exp(-\mu \cdot \min(\frac{c}{2}, \frac{c^2}{4}))$.

& $\forall j$, $\Pr_I[|I \cap I_j| > d] < 2 \cdot e^{-2d/5}$.

$\Rightarrow \Pr[|I| < n \ \vee \ \exists j, |I \cap I_j| > d]$
$< 2e^{-n/8} + 2e^{-4d/10 + d/10} < 1$

$\Rightarrow \Pr_I[|I| \geq n \wedge \forall j, |I \cap I_j| \leq d] > 0$.

$\Rightarrow$ There exists an I in Step-(1).
(If it is larger than n then we can drop the extra elements.) □

- Let us use this design now.

Lemma 2 (NW-generator): If $\mathcal{I}$ is an (ℓ, n, d)-design with $|\mathcal{I}| = 2^{d/10} =: m$, $f: \{0,1\}^n \to \{0,1\}$ & $H_{avg}(f) > 2^{2d}$, then $NW_f^{\mathcal{I}}(U_\ell)$ is $(H_{avg}(f)/10, 0.1)$-pseudorandom.

Proof:

- Let $S := H_{avg}(f)$.
- Suppose $\exists$ a circuit C of size $\leq S/10$ s.t.
  $|Pr[C(NW_f^{\mathcal{I}}(U_\ell)) = 1] - Pr[C(U_m) = 1]| \geq 0.1$.
  (I.e. $NW_f^{\mathcal{I}}$ is not pseudornd →)
- Wlog assume,
  $Pr[C(NW_f^{\mathcal{I}}(U_\ell)) = 1] - Pr[C(U_m) = 1] \geq 0.1$.
- We will now devise a bit-predictor for $NW_f^{\mathcal{I}}$.

- For that let us define distributions $\mathcal{D}_0, \ldots, \mathcal{D}_m$ over $\{0,1\}^m$ s.t. $\forall i$,

  $\underline{\mathcal{D}_i}$: choose $x \in_R \{0,1\}^\ell$; $z_{i+1}, \ldots, z_m \in_R \{0,1\}$,
  compute $y = NW_f^x(x)$
  output $\langle y_1, \ldots, y_i, z_{i+1}, \ldots, z_m \rangle$.

hybrid distribution

▷ $\mathcal{D}_0 \equiv U_m$ & $\mathcal{D}_m \equiv NW_f^x(U_\ell)$.

- Define $p_i := Pr[C(\mathcal{D}_i) = 1]$.

- Since $p_m - p_0 \geq 0.1$, averaging gives us:
  $\exists i_0 \in [m]$, $p_{i_0} - p_{i_0-1} \geq 0.1/m$

- We will use this advantage to predict the i_0-th bit of $NW_f^x(U_\ell)$ given the preceding (i_0-1) bits.

- Define circuit $\underline{C'}$: on input $y_1, \ldots, y_{i_0-1}$,
  pick $z_{i_0}, \ldots, z_m \in_R \{0,1\}$,
  output $\begin{cases} z_{i_0}, & \text{if } C(y_1, \ldots, y_{i_0-1}, z_{i_0}, \ldots, z_m) = 1 \\ 1 - z_{i_0}, & \text{else.} \end{cases}$

• How well does C' predict?

$$\Pr_{y \in NW(U_\ell),\, \bar{z} \in U_m}\left[C'(y_1,\dots,y_{i_0-1}) = y_{i_0}\right] =$$

$$\Pr[z_{i_0} = y_{i_0}] \cdot \Pr\left[C(y_1,\dots,y_{i_0-1}, z_{i_0},\dots,z_m) = 1 \,\middle|\, z_{i_0} = y_{i_0}\right]$$
$$+ \Pr[z_{i_0} \neq y_{i_0}] \cdot \Pr\left[\quad\quad\quad\quad\quad\quad\quad\quad \neq 1 \,\middle|\, z_{i_0} \neq y_{i_0}\right]$$

$$= \frac{1}{2} \cdot \Pr[C(D_{i_0}) = 1] + \frac{1}{2} \cdot \left(1 - \Pr\left[C(y_1,\dots,y_{i_0-1}, \bar{y}_{i_0}, z_{i_0+1},\dots) = 1\right]\right)$$

$$= p_{i_0} + \frac{1}{2} - \frac{1}{2}\left(p_{i_0} + \Pr[C(y_1,\dots,y_{i_0-1}, \bar{y}_{i_0}, z_{i_0+1},\dots) = 1]\right)$$

union bound ↘

$$= p_{i_0} + \frac{1}{2} - \frac{1}{2} \cdot (2p_{i_0-1}) \geqslant \left(\frac{1}{2} + \frac{0.1}{m}\right).$$

• To make C' deterministic, we could fix $z_{i_0},\dots,z_{i_m}$ & get a circuit C'' s.t.

$$\Pr_{y \in NW(U_\ell)}\left[C''(y_1,\dots,y_{i_0-1}) = y_{i_0}\right] \geqslant \left(\frac{1}{2} + \frac{0.1}{m}\right).$$

z_{i_0} C $\vee$ $\bar{z}_{i_0}$ $\bar{C}$

• Clearly, $size(C'') < 2 \cdot size(C) \leq 5/5$.

• Now we plug the definition of NW_g^f, to get:

$$\Pr_{Z \in U_\ell}\left[C''(f(Z_{I_1}), \ldots, f(Z_{I_{i_0-1}})) = f(Z_{I_{i_0}})\right] \geq \left(\frac{1}{2} + \frac{0.1}{m}\right)$$

- Let us fix $Z_{[\ell]\setminus I_{i_0}}$ s.t. the above probability advantage is retained.

- Note that this leaves only $|I_j \cap I_{i_0}|$ many variables free in Z_{I_j}, $j \in [i_0-1]$.
$\Rightarrow f(Z_{I_1}), \ldots, f(Z_{I_{i_0-1}})$ are d-variate.
$\Rightarrow$ " " can be computed (trivially) by circuits of size $O(d \cdot 2^d)$.

$\Rightarrow \exists$ a circuit B of size $< \frac{S}{5} + O(d2^d) \cdot m$
$= S/5 + O(d\,2^{d + d/10}) < S$ $(\because S > 2^{2d})$ s.t.

$$\Pr_{Z_{I_{i_0}} \in U_n}\left[B(Z_{I_{i_0}}) = f(Z_{I_{i_0}})\right] \geq \frac{1}{2} + \frac{0.1}{m} > \frac{1}{2} + \frac{1}{S}.$$

- This contradicts the assumption that $H_{avg}(f) = S$.
$\Rightarrow NW^f_\ell(U_\ell)$ is $(S/10, 0.1)$-pseudorandom. $\square$

Proof (NW theorem):

- Let $f \in \mathrm{Dtime}(2^{O(n)})$ s.t. $H^{avg}(f) \geq S(n)$.
- We will define an $S'(\ell)$-prg G:

On input $z \in \{0,1\}^\ell$,

1) Pick n s.t. $\frac{100n^2}{\ell g S(n)} < \ell \leq \frac{100(n+1)^2}{\ell g S(n+1)} \leq \frac{200n^2}{\ell g S(n)}$.

2) Set $d = \ell g S(n)/10$.

3) Compute an (ℓ, n, d)-design $\mathcal{I} = \{I_1, \ldots, I_m\}$ with $m = 2^{d/10}$.

4) Output $NW_f^{\mathcal{I}}(z)$.

- This takes time: $2^{O(\ell)} + 2^{O(n)} \cdot 2^{d/10} = 2^{O(\ell)}$.
- Since $H^{avg}(f) \geq S(n) = 2^{10d}$, by Lemma 2 we get: $NW_f^{\mathcal{I}}(U_\ell)$ is $(S(n)/10, 0.1)$-pseudorandom.
- Finally, the stretch is $2^{d/10} = S(n)^{1/100} =: S'(\ell)$.
- Clearly, G is an $S'(\ell)$-prg. □

($\because S'(\ell)^3 < S(n)/10$.)

- Thus, "hardness ⇒ prg".
  Is there a converse?

↙ $s(\ell) > \ell$

Claim: If $\exists$ $s(\ell)$-prg then $\exists f \in E$ s.t. $H_{wrs}(f_n) > n^3$.

Proof:

- Let $G : \{0,1\}^\ell \to \{0,1\}^n$ be an $s(\ell)$-prg.
- Consider the function $f_n : \{0,1\}^n \to \{0,1\}$ s.t. $f_n(x) = 1$ iff $x \in im(G)$. Clearly, $f \in E$.
- Let C_n be the smallest circuit computing f_n.

- Also, $Pr[C_n(G(U_\ell)) = 1] = 1$
  while $Pr[C_n(U_n) = 1] \leq 2^\ell / 2^n \leq 1/2$
  $\Rightarrow$ C_n distinguishes $G(U_\ell)$ from U_n well.
  $\Rightarrow$ $size(C_n) > s(\ell)^3 = n^3$. □

- We will now see more impressive applications of prg in complexity:

Partial derandomization from $H^{wrs}(per)$.

Theorem (Impagliazzo, Wigderson 1998): If $BPP \neq EXP$ then $\forall L \in BPP$, $\exists$ subexponential-time algorithm A s.t. for only-many n's:

$$\Pr_{x \in \{0,1\}^n}[A(x) = L(x)] \geq 1 - \frac{1}{n}.$$

↑ the det. algo. A is right on average.

Proof sketch:

- If $EXP \not\subseteq P/poly$ then $\exists f \in EXP$ with $H^{wrs}(f) = n^{\omega(1)}$.

  Later we will see how to amplify this to get an $f' \in EXP$ with $H^{avg}(f') = n^{\omega(1)}$. NW-theorem then implies $BPP \subseteq$ Subexp.

- So, assume $EXP \subseteq P/poly$.

  EXP⊆MA ⊆PH⊆EXP →

  Then (recall the initial lectures), $EXP = PH$. This, with Toda's theorem ($PH \subseteq P^{\#P}$) means that $P^{\#P} = EXP$.

  $\Rightarrow$ $P^{\#P} \not\subseteq BPP$.

- This, essentially, says that #P is hard & we will use it to define $G := NW_f^{\#P} : \{0,1\}^{\ell} \to \{0,1\}^n$,

with a superpoly- stretch.

- For an $L \in BPP$, if $B(x,r)$ is the randomized algorithm solving L, then we define the promised A as:

$$A(x) := \text{majority}\{ B(x, G(U_\ell)) \}.$$

- Suppose the Thm. statement is false. Then, for almost all n's: $\Pr_{x \in U_n}[A(x) = L(x)] < 1 - \frac{1}{n}$.

ie. all except finitely many

$$\Rightarrow \Pr_{x \in U_n}\left[\text{maj}\{B(x, G(U_\ell))\} \neq \text{maj}\{B(x, U_n)\} \right] > 1/n.$$

$\Rightarrow$ We can fix $x = s_n \in \{0,1\}^n$ s.t. the circuit family $\{D_n := B(s_n, \cdot) \mid n\}$ can distinguish, $G(U_\ell)$ from U_n, well.

- In fact, the circuit D_n can be constructed by a randomized poly-time algorithm (whp).

- Recalling the properties of $G = NW_g^{per}$, we can deduce that $\exists$ randomized poly-time algorithm T that can "learn" per_N, i.e.
  Given <u>oracle access</u> to per_N, T runs in poly(N)-time & produces a poly(N)-sized circuit computing per_N.

- Now we can remove the need for the oracle because per_N is <u>self-reducible</u>:

$$per_N(M) = \sum_{i \in [N]} M_{1i} \cdot per_{N-1}(minor_{1i}(M)).$$

$\Rightarrow$ T can build $per_1, per_2, \ldots, per_N$ recursively.

$\Rightarrow$ $P^{per} \subseteq BPP$, which is a contradiction.
$\Rightarrow$ $A(x)$ is "mostly" correct. $\square$

- The next prg application completes the proof of "PIT $\in$ P $\Rightarrow$ lower bounds".

Theorem (Impagliazzo, Kabanets, Wigderson 2001):

$NEXP \subseteq P/poly \Rightarrow NEXP = EXP$.

Proof sketch:

- Let us assume that $EXP \subsetneq NEXP \subseteq P/poly$.
- We will derive a contradiction to the time-hierarchy theorem.
- Idea – $\exists L \in NEXP \setminus EXP$, which can be used to get a "hard" function. By the "worst-case vs. prg" we get a poly-stretch prg that can "derandomize" $EXP \subseteq MA$.

- Pick an $L \in NEXP \setminus EXP$. $\exists c > 0$ & a relation $R(x, y)$ testable in $exp(|x|^{10c})$-time s.t.

$$x \in L \text{ iff } \exists y \in \{0,1\}^{exp(|x|^c)},\ R(x,y) = 1.$$

- We now consider the complexity of y given x.
- For $D > 0$, let M_D be the following machine:

On input $x \in \{0,1\}^n$,

1) enumerate boolean circuits of size n^{100D} that

take n^c-bit input & output 1-bit.

2) For each such circuit C, let $tt(C)$ be the 2^{n^c}-long string that corresponds to the truth-table of C.

3) If $\exists$ such C, $R(x, tt(C)) = 1$ then OUTPUT 1.

4) Else OUTPUT 0.

▷ M_D runs in time $\exp(n^{10D} + n^{10c})$.

• $\therefore L \notin EXP$, M_D cannot solve L. Thus, $\forall D$, $\exists$ infinite sequence of inputs $X_D := \{s_i | i\}$ on which $M_D(s_i) = 0$ even though $s_i \in L$.

$\Rightarrow \forall x \in X_D$, the y, for which $R(x,y)=1$, represents the truth-table of a "hard" function that cannot be computed in $Size(n^{100D})$.

• By worst-case-hardness based prg, we can use y to get a ℓ^D-prg G_D.

- We know that $EXP \subseteq P/poly \Rightarrow EXP \subseteq MA$.
- Thus, $\forall L' \in EXP$, Merlin proves $x' \in L'$ by sending a proof, which Arthur can verify by a randomized algorithm in, say, n^D steps ($n := |x'|$).
- Here, Arthur can use the prg G_D.

  Let $x'' \in X_D$, $|x''| =: n$. Arthur guesses a string $y \in \{0,1\}^{exp(n^c)}$ s.t. $R(x'', y) = 1$ & uses y to design G_D.

  Using G_D, Arthur reduces the random n^D-bits to n-bits.

this saves us from testing $x'' \in X_D$

▷ Arthur needs $poly(n^D) \cdot 2^{n^{10c}}$-time, n random bits, n advice bits (for x''), 2^{n^c}-bit guess (for y).
$\Rightarrow \exists c' > 0$ s.t.

▷ $EXP \subseteq$ i.o.$-NTime(2^{n^{c'}})/2n$.

i.o. = infinitely-often; $2n$ = advice-bits

[For a class $\mathcal{C}$, $L \in$ i.o.$-\mathcal{C}$ if $\exists M \in \mathcal{C}$ s.t. $L \cap \{0,1\}^n = M \cap \{0,1\}^n$ for ∞-ly many n.]

• ∴ NEXP ⊆ P/poly, we can further write:
$\exists c'' > 0$, $EXP \subseteq$ i.o.-Size($n^{c''}$).

• By standard diagonalization this can be ruled out. (Exercise.)

• This contradiction means:
$NEXP \neq EXP \Rightarrow NEXP \not\subseteq P/poly$. □

– This leads to the result:

Theorem (Impagliazzo & Kabanets, 2003):
$PIT \in P \Rightarrow NEXP \not\subseteq P/poly$ or
per $\not\in$ AlgP/poly.