Partial derandomization from $H_{wrs}(\text{per})$:

Theorem (Impagliazzo, Wigderson '98): $BPP \neq EXP \Rightarrow$
$\forall L \in BPP$, $\exists$ subexp. time algorithm A solving L on "average", i.e. for infinitely-many n's:

(& infinitely often)

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

Pf: • If $EXP \not\subseteq P/poly$ then $\exists f \in EXP$ with $H_{wrs}(f) > n^{\omega(1)}$. Later we'll see how to amplify this to get $f' \in EXP$ with $H_{avg}(f') > n^{\omega(1)}$.
$\Rightarrow$ NW-theorem gives $BPP \subseteq Subexp$.

• Now, assume $EXP \subseteq P/poly$.

- Then, $EXP = PH$. [Recall $EXP \subseteq MA \subseteq PH \subseteq EXP$]

$\Rightarrow EXP = P^{per}$. [Also, $PH \subseteq P^{per} \subseteq EXP$]

$\Rightarrow P^{per} \nsubseteq BPP \subseteq P/poly$.

- So, per is "hard" & we'll use it to define $G := NW_g^{per} : \{0,1\}^{\ell} \to \{0,1\}^n$ with super-poly-stretch.

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

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

- Suppose the theorem-statement is false. Then, for all except finitely many n: $\Pr_{x \in U_n}[A(x) = L(x)] < 1 - \frac{1}{n}$.

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

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

- In fact D_n is constructible by a randomized poly-time algorithm.

- Recall the properties of $NW_\ell^{per} =: G$. Deduce:

$\exists$ randomized poly-time algo. T that can learn per.

$\Rightarrow$ bit-predictor for NW!

- Given oracle access to per_N, T runs in poly(N)-time. Produces poly(N)-size circuit computing per_N.

- Eliminate the oracle access by using <u>self-reducibility</u> of per_N: $\text{per}_N(M) = \sum_{i\in[N]} M_{1i}\cdot \text{per}_{N-1}(\text{minor}_{1i}(M))$.

$\Rightarrow$ T builds $\text{per}_1, \text{per}_2, \dots \to \text{per}_N$ recursively (giving "small" circuits).

$\Rightarrow$ $\text{Pper} \subseteq \text{BPP}$; which is a contradiction.

$\Rightarrow$ $A(x)$ is "mostly" correct. $\square$

- Now, let us move to the earlier unproved theorem: (whose statement has no mention of prg!)

Theorem (Impagliazzo, Kabanets, Wigderson 2001):

$\mathsf{NEXP} \subseteq \mathsf{P/poly} \implies \mathsf{NEXP} = \mathsf{EXP}$.

Pf: • Assume $\mathsf{EXP} \subsetneq \mathsf{NEXP} \subseteq \mathsf{P/poly}$. ($\mathsf{EXP} \subseteq \mathsf{P/poly}$)

Idea – $\exists L \in \mathsf{NEXP} \setminus \mathsf{EXP}$, which can be used to get a "hard" function. By "hardness vs. prg", we get a poly-stretch prg that derandomizes Arthur in $\mathsf{EXP} = \mathsf{MA}$.

Finally, contradict a hierarchy theorem!

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

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

- What's the circuit-complexity of certificate y?
  - View y as a truth-table of a function!
- For $D>0$, let M_D be the following TM to search y (as t.t.):

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

  1) Enumerate circuits of size n^{100D}, with n^c-bit input.
  2) For each such C: Let $tt(C)$ be the 2^{n^c}-bit long truth-table of C.
  3) If $\exists$ such a C: $R(x, tt(C))=1$, then OUTPUT YES.
  4) Else OUTPUT NO.

▷ M_D runs in time $\exp(n^{101D}+n^{10c})$.

- Since $L \notin EXP$, M_D cannot solve L. (i.e. unable to find y)

$\Rightarrow \forall D$, $\exists$ infinite-sequence of (hard) inputs $\mathcal{X}_D := \{s_i \mid i\}$ on which $M_D(s_i) = 0$ but $s_i \in L$.

$\Rightarrow \forall x \in \mathcal{X}_D$, certificate y (for which $R(x,y)=1$) is a tt of a hard function that is not in $\mathrm{size}(n^{100D})$.

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

super-poly-stretch

- Recall $EXP \subseteq P/poly \Rightarrow EXP$ has MA-protocol.
- Thus, $\forall \underline{L'} \in EXP$, Merlin proves $x' \in ? L'$ by sending a (short) proof.

  Arthur verifies by a randomized algorithm in, say, $\underline{n^D}$ steps ($\underline{n} := |x'|$).

- Arthur could now use prg G_D:
  - Let $x'' \in X_D$, $|x''| =: n$. Arthur guesses $y \in \{0,1\}^{exp(n^{4})}$: $R(x'', y) = \underline{1}$. Use y to get G_D.

$\triangleright$ For Arthur, G_D reduces random bits from n^D to n.

▷ Arthur needs $\text{poly}(n^D) \cdot 2^{n^{10c}}$ -time, n rnd bits, n-bit advice (x''), 2^{n^c}-bit guess (y).

[for infinitely-many length n.]

▷ $\exists c' > 0$ s.t. $\text{EXP} \subseteq \text{i.o.-Ntime}(2^{n^{c'}})/2n$.

i.o. ← infinitely-often

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

- By hypothesis, $\text{NEXP} \subseteq \text{P/poly}$. We can further deduce:

▷ $\exists c'' > 0$ s.t. $\text{EXP} \subseteq \text{i.o.-Size}(n^{c''})$.

Exercise: This is ruled out by standard diagonalization based proof. (Note: c'' is fixed!)

$\Rightarrow$ NEXP $\neq$ EXP $\Rightarrow$ NEXP $\not\subseteq$ P/poly. $\square$

- As we saw in the first few classes, this leads to the result:

<u>Theorem</u> (Impagliazzo & Kabanets '03): $PIT \in P$

$\Rightarrow \begin{cases} \text{NEXP} \not\subseteq \text{P/poly} & \text{OR} \\ \text{per} \notin \text{Ar P/poly}. \end{cases}$

- The only detail that remains is to convert $H_{wrs}(\cdot)$ to $H_{avg}(\cdot)$.