Derandomization

— i.e. solving a problem without rnd bits.

Open Qn: BPP = P ?

— We'll show that this question is connected to proving "lower bounds" or "explicit hardness".

BPP=P ⇒

Theorem (Kabanets & Impagliazzo '03): PIT $\in$ P $\Rightarrow$ NEXP $\not\subseteq$ P/poly OR per $\notin$ Ar-P/poly.

[Connects existence of algo. to non-existence!]

⇒ circuit lower bounds.

⇒ non-existence of fast algos!

- Remark: • per is the functional problem of computing the permanent polynomial of $A_{n\times n}$.

• $\mathrm{per}(A) := \sum_{\sigma \in \mathrm{Sym}(n)} A_{1,\sigma(1)} \cdots A_{n,\sigma(n)}$

permutations

• We're mainly interested in $\mathbb{F} = \mathbb{Q}$.

→ We believe in all the three statements that the Theorem connects!

- The proof is involved & depends on several older results.

- Meanwhile, we prove a simpler connection as a detour:

Theorem [Heintz & Schnorr '80]: Blackbox-PIT $\in P$
$\Rightarrow$ $\exists$ E-explicit polynomial family which is exponentially-hard.

Proof:
- Suppose you designed a set of points $p_1, \ldots, p_m \in \mathbb{F}^n$ for blackbox-PIT (of n-variate size-s circuits).
- annihilator $\Rightarrow$ We'll find a polynomial $A(y_1, \ldots, y_\ell)$, where $\ell := 2\lceil \lg s \rceil + 1$ s.t. $\forall i \in [m]$, $A(p_i) = 0$.
- Multilinear $A =: \sum_{\bar{e} \in \{0,1\}^\ell} a_{\bar{e}} \cdot \bar{y}^{\bar{e}}$ has unknown coefficients $a_{\bar{e}}$.

- Constraints are: $\forall i \in [m]$, $A(p_i) = 0$. --- (1)

▷ $m = s^2$ (say) constraints & 2^ℓ unknowns.

▷ Each constraint is <u>linear</u> & <u>homogeneous</u>.

- Since, $2^\ell > m = s^2$, the system has a solution A.

▷ $\{a_{\bar{e}}\}_{\bar{e}}$ can be computed in $\mathrm{poly}(2^\ell) = \mathrm{poly}(s)$-time.

$\Rightarrow$ E-explicit wrt ℓ.

▷ $A(y_1, \ldots, y_\ell)$ cannot have size-s circuit.

[Pf: Use $A(p_i) \neq 0$ for some $i \in [m]$.]

$\Rightarrow$ $A(y_1, \ldots, y_\ell)$ is $2^{\Omega(\ell)}$-hard & $2^{O(\ell)}$-time-explicit. $\square$

– Let's go back to proving Thm [KI'03]:

Lemma 1: $PIT \in P$ & $per \in Ar\text{-}P/poly \Rightarrow P^{per} \subseteq NP$

Proof: • Idea – "Guess" the small circuit for per & "verify" using $PIT \in P$.

• We can expand permanent of an $n \times n$ matrix A, $per_n(A)$, by the first row:

minor

$$per_n(A) = \sum_{i \in [n]} A_{1i} \cdot per_{n-1}(A'_{1i}) .$$

where, A'_{1i} := submatrix of A deleting row=1, col=i.

• Given a circuit $C_{(n-1)^2}$ for per_{n-1}, we can use the following verification protocol:

$$C_{n^2}(A) \stackrel{?}{=} \sum_{i\in[n]} A_{1i} \cdot C_{(n-1)^2}(A'_{1i}) \quad \text{------} \quad (2)$$

▷ This guess-&-verify process will prove $C_{n^2}(A) = \mathrm{per}_n(A)$ by induction on n. [(2) characterizes permanent]

- So, for any $L \in P^{\mathrm{per}}$ [i.e. poly-time TM using oracle per] we can first guess circuit C_{n^2} for per_n; verify it by using PIT on eqn. (2); then use C_{n^2} instead of the oracle.

$$\Rightarrow L \in NP \quad \Rightarrow \quad P^{\mathrm{per}} \subseteq NP$$

$$\Rightarrow P^{\mathrm{per}} = NP. \qquad \square$$

- The "strange" assumption of $\mathrm{per} \in \mathrm{Ar}\text{-}P/\mathrm{poly}$ gives the " conclusion $P^{\mathrm{per}} \subseteq NP$.

- Let's now make another "strange" assumption: $NEXP \subseteq P/\mathrm{poly}$. From this, we want to show that $NEXP \subseteq P^{\mathrm{per}}$ $(\subseteq NP)$; which contradicts the non-deterministic time-hierarchy!

  $\Rightarrow$ one of the strange assumptions is FALSE.

- This proof requires: quantifier-based & interaction-based complexity classes.

- **Definition:** (Quantifiers $\exists, \forall$) • $\Sigma_0 := P$, $\Sigma_1 := NP$
  - $\Sigma_2 := NP^{NP}$, $\Sigma_3 := NP^{\Sigma_2}$, ...
  - I.e. $L \in \Sigma_2$ if $\exists$ poly-time NDTM using oracle SAT.
- It can be shown:

▷ $L \in \Sigma_2 \iff \exists$ poly-time TM N s.t. $\forall x$, $x \in L$ iff
$\exists y_1, \boxed{\forall y_2}, N(x, y_1, y_2) = 1$.

size = poly(|x|)

without this you get $\Sigma_1 = NP$!

$\Rightarrow$ Informally, $\Sigma_1 = \exists \cdot P$ & $\Sigma_2 = \exists \cdot \forall \cdot P$.

- Σ_3 is defined via alternating three quantifiers:
  $\exists y_1 \underline{\forall y_2 \exists y_3}$.

Informally, $\Sigma_3 = \exists \cdot \forall \cdot \exists \cdot P$

· Polynomial Hierarchy PH $:= \bigcup_{i\geq 0} \Sigma_i$.

▷ PH $\subseteq$ Pspace .

Pf. sketch: · Systematically, go over all the possibilities of the quantified strings $y_1, y_2, \ldots, y_c$.

· Requires only poly($|x|$)-space $\square$

OPEN: · $P = \Sigma_0 \subsetneq \Sigma_1 \subsetneq \Sigma_2 \cdots \subsetneq$ PH $\subsetneq$ Pspace ?

· $P \subsetneq$ Pspace ?

- Third quantifier is "$\mathcal{M}$" = "for most strings"

- Defn: • " $\mathsf{M} y \in \{0,1\}^n$, $N(y)=1$ " is True iff
$$\Pr_{y\in\{0,1\}^n}[N(y)=1] \geq 3/4 .$$
Arthur (verifier)
Merlin → (prover)

• k-alternations of M & $\exists$ gives us the class $AM[k]$:
$L \in AM[k]$ if $\forall$ input x, $x\in L$ iff
$$\underbrace{\mathsf{M} y_1 \, \exists y_2 \, \mathsf{M} y_3 \cdots}_{k}, \ [N(x, y_1, \ldots, y_k) = 1]$$

- Eg. $k=1$: $\mathsf{M} y_1 \, [N(x, y_1) = 1]$ gives $L \in BPP = AM[1]$.

• Starting the alternation with "$\exists$" gives the class $MA[k]$.
$MA[1] = NP$.

- We can interpret these classes as k-rounds of interaction:

- Eg. AM[1]:

Arthur $\langle x, y_1 \rangle$ (y_1 ↖ rnd) Merlin (not used)

▷ AM[1] = BPP.

- Eg. MA[1]:

Arthur $\langle x \rangle$ $\xleftarrow{\quad y_1 \quad}$ Merlin

▷ MA[1] = NP.

- Eg. MA[2]: (Also called <u>MA</u>)

Arthur $\langle x \rangle \langle y_2 \rangle$ (y_2 ↖ rnd) $\xleftarrow{\quad y_1 \quad}$ Merlin

▷ MA[2] = randomized-version of NP.

- Eg. AM[2]:

Arthur $\langle x \rangle \langle y_1 \rangle$ (y_1 ↖ rnd) $\xleftarrow{\quad y_2 \quad}$ Merlin

▷ Another rnd. NP version. Called the class <u>AM</u>.

– Defn: • If we make k variable (depending on $|x|$)
then we get IP $:= \bigcup_{c>0} AM[n^c]$.
(interactive protocol)

▷ All these classes are in Pspace.
Pf. sketch: Simulate $\exists$, $\forall$, M in poly(n)-space. $\square$

Theorem (Shamir '90): IP = Pspace.
Pf. sketch: • Pick a hard problem in Pspace & give an interactive protocol.

- We'll use Quantified Boolean Formula (QBF):

  Given a formula $\psi := Q_1 x_1\, Q_2 x_2 \cdots Q_n x_n\; \phi(\bar{x})$, where $Q_i \in \{\forall, \exists\}$ & ϕ is a boolean formula; test whether $\psi = \text{True}$. over $\{0,1\}$

  ▷ QBF is Pspace-complete. [Exercise]

- So we will show $\text{QBF} \in \text{IP}$:

  (the protocol is algebraic, based on PIT!)

- Define an arithmetized version P_ϕ for ϕ,

  eg. For $\phi = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_3 \vee x_4)$, define P_ϕ as $(1-(1-x_1)(1-x_2)) \cdot (1 - x_1(1-x_3)(1-x_4))$

  ▷ $\phi(a_1, \ldots, a_4) = \text{True}$ iff $P_\phi(a_1, \ldots, a_4) = 1$.

- Arithmetization of ϕ is easy to do.
- Extend this to QBF ψ as:

$\forall x_i$ converts to $\prod_{x_i \in \{0,1\}}$ &

$\exists x_i$ converts to $\sum_{x_i \in \{0,1\}}$.

▷ Finally, $\psi := Q_1 x_1 \cdots Q_n x_n \ \phi(\bar{x})$ converts to polynomial

$$P_\psi := \tilde{Q}_1 \tilde{Q}_2 \cdots \tilde{Q}_n \ P_\phi(x_1, \ldots, x_n),$$

where $\tilde{Q}_i \in \{ \sum_{x_i}, \prod_{x_i} \}$.

▷ $\psi = \text{True}$ iff $P_\psi \neq 0$.

integer

- How could Merlin convince Arthur: $P_\psi \neq 0$?

Idea: • Merlin tries to prove to Arthur: $P_\psi = k$, in n rounds of interaction (where $0 \neq k$ is n-bits).

• In i-th round, Merlin un-fixes variables $\{x_1, \ldots, x_i\}$ & sends some partial polynomial to Arthur.

• Arthur does PIT.

Protocol: 0) M sends $k \neq 0$, claiming $P_\psi = k$.

1) If $n=1$ then A accepts iff:

$$\begin{cases} Q_1 = \forall \ \& \ P_\phi(0) \cdot P_\phi(1) = k \\ Q_1 = \exists \ \& \ P_\phi(0) + P_\phi(1) = k. \end{cases}$$

2) If $n>1$ then M sends $\underline{s(x_1)}$, 'claiming' it to be $= \tilde{Q}_2 \tilde{Q}_3 \cdots \tilde{Q}_n P_\phi(x_1, x_2, \ldots, x_n)$

free

3) A tests: $\begin{cases} Q_1 = \forall \ \ \& \ \ s(0) \cdot s(1) = k \\ Q_1 = \exists \ \ \& \ \ s(0) + s(1) = k \end{cases}$

& tests for random $\alpha \in \mathbb{Z}$: $s(\alpha) = \tilde{Q}_2 \cdots \tilde{Q}_n P_\phi(\alpha, x_2, \ldots x_n)$.

is done recursively, by interacting with M.

$\square$

Exercise: Show that if M errs, then A detects it with high probability! (Use P.I. Lemma.)

- Let's prove other lemmas that we need.

<u>Lemma</u> (Babai, Fortnow, Nisan, Wigderson '93):

$$EXP \subseteq P/poly \implies EXP = MA.$$

<u>Pf. sketch</u>:

- Suppose $EXP \subseteq P/poly$. We'll show: $EXP = \Sigma_2$:
- Let $L \in EXP$ & N be its exponential-time TM.
- <u>Idea</u>: Encode the steps of N using $\exists \cdot \forall$.

- The j-th bit in the i-th configuration of $N(x)$ is computable in EXP. $\implies \exists$ poly-size circuit $C(x,i,j)$ that computes this bit.

- So, $x \in L \iff \exists C, \forall (i,j), [C(x,i,j) \to C(x,i+1,j)$ is a valid step of $N]$.
- This means: $L \in \Sigma_2$.

  $\implies EXP \subseteq \Sigma_2 \implies EXP = \Sigma_2$.
- We've $\Sigma_2 \subseteq Pspace = IP \subseteq EXP = \Sigma_2$.

  $\implies Pspace = IP = EXP \subseteq P/poly$.

$\implies$ Merlin can be seen as a Pspace-machine, hence now it can be simulated by a poly-size circuit family $\{C_n\}_{n \geq 1}$.

- This suggests a 1-round protocol to convince Arthur: $x \in L$:

1) Merlin sends circuit C, claiming it to be C_n for $n := |x|$.

2) Arthur runs the protocol on x using C instead of challenging Merlin.

- The protocol $\Rightarrow$ $L \in MA$.

$$\Rightarrow \quad EXP \subseteq MA \quad \Rightarrow \quad EXP = MA. \qquad \square$$

- The final lemma is the most advanced — we'll prove it when we do pseudorandom generators.

<u>Lemma</u> (Impagliazzo, Kabanets, Wigderson 2001):

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

- Let's finish the proof of PIT-LB-theorem :

$PIT \in P \implies NEXP \not\subseteq P/poly$ OR $per \notin Ar\text{-}P/poly$.

Proof: • Suppose $NEXP \subseteq P/poly$.

$\implies NEXP = EXP = MA \subseteq PH$.

- Also, by Toda's theorem, $PH \subseteq P^{per}$

$\implies NEXP \subseteq P^{per}$.

- Assume $per \in Ar\text{-}P/poly$ & $PIT \in P$.

$\implies P^{per} \subseteq NP$.

$\implies NEXP \subseteq NP$; which contradicts non-det. time hierarchy.

- Thus, $PIT \in P \implies$ Either $NEXP \not\subseteq P/poly$

BPP=P

OR $per \notin Ar\text{-}P/poly$. $\square$