Polynomial Identity Testing and
Circuit Lower Bounds

Robert Špalek, CWI
based on papers by
Nisan & Wigderson, 1994
Kabanets & Impagliazzo, 2003
Randomised algorithms

- For some problems (polynomial identity testing) we know an efficient randomised algorithm, but not a deterministic one.

- However nobody proved $P \subset BPP$ yet. It is possible that $P = BPP$.

- There is a connection between *hardness* and *randomness*: if we have a hard function, we can use it to *derandomize* BPP.

- Until recently, it was not known whether the converse holds. Kabanets & Impagliazzo showed that it does.

- This is bad, since non-trivial circuit lower-bounds are a long-standing open problem.
Pseudo-random generators

\[ G : \{0,1\}^{\ell(n)} \to \{0,1\}^n \] is a **pseudo-random generator**
iff for any circuit \( C \) of size \( n \):

\[
|P[C(r) = 1] - P[C(G(x)) = 1]| < \frac{1}{n'}
\]

where \( x, r \) are chosen uniformly.

Having a pseudo-random generator, we can **derandomize** BPP:

- instead of \( n \) random bits, plug a pseudo-random sequence
  (acceptance prob. changed only slightly)
- check all \( 2^{\ell(n)} \) random seeds
Hard functions

\( f_n : \{0,1\}^n \rightarrow \{0,1\} \) has hardness \( h \) iff for any circuit \( C \) of size \( h \):

\[
\left| P[C(x) = f(x)] - \frac{1}{2} \right| < \frac{1}{2h},
\]

where \( x \) is chosen uniformly.

Hard functions can be used to build pseudo-random generators:

- take \( \ell(n) \) truly random bits
- evaluate \( f \) on \( n \) subsets of them
- if these subsets have small intersection, then the results are hardly correlated
Nearly disjoint sets

System of sets \( \{S_1, \ldots, S_n\} \), where \( S_i \subseteq \{1, \ldots, \ell\} \) is a \((k, m)\)-design if:

- \( |S_i| = m \)
- \( |S_i \cap S_j| \leq k \)

For every \( m \in \{\log n, \ldots, n\} \), there exists an \( n \times \ell \) matrix which is a \((\log n, m)\)-design, where \( \ell = O(m^2) \).

(If \( m = O(\log n) \), then even \( \ell = O(m) \) is enough.)

Assume \( m \) is a prime power. Take \( S_q = \{ \langle x, q(x) \rangle \mid x \in GF(m) \} \), where \( q \) has degree at most \( \log n \). Can be computed in log-space.
Let $f$ have hardness $\geq n^2$ and $S$ be a $(\log n, m)$-design. Then $G : \{0, 1\}^\ell \rightarrow \{0, 1\}^n$ given by $G(x) = f_S(x)$ is a pseudo-random generator.

1. Assume a circuit $C$ distinguishes random $r$ and $y = G(x)$ w.p. $> \frac{1}{n}$. Let $p_i = P[C(z) = 1]$, where $z = y_1 \ldots y_ir_{i+1} \ldots r_n$. There must be $i$ such that $p_{i-1} - p_i > \frac{1}{n^2}$.

2. Build a circuit $D$ that predicts $y_i$ from $y_1 \ldots y_{i-1}$ w.p. $\geq \frac{1}{2} + \frac{1}{n^2}$. $D$ evaluates $C(y_1 \ldots y_{i-1}, r_i \ldots r_n)$ and returns $r_i$ iff $C = 1$. 


3. Assume w.l.o.g. $S_i = \{1, \ldots, m\}$, then $y_i = f(x_1 \ldots x_m)$. Since $y_i$ does not depend on other bits, there exists some assignment of $x_{m+1} \ldots x_\ell$ preserving the prediction prob.

4. After fixing, every $y_1 \ldots y_{i-1}$ depends only on $\log n$ variables, hence can be computed from $x$ as a CNF of size $O(n)$.

5. Plug computed $y_1 \ldots y_{i-1}$ into $D$ and obtain a circuit predicting $y_i$ from $x$ w.p. $\geq \frac{1}{2} + \frac{1}{n^2}$.

This contradicts that $f$ has hardness $\geq n^2$. 
Hardness-randomness tradeoff

If there exists a function computable in $E = \text{DTIME}(2^{O(n)})$ that cannot be approximated by

1. polynomial-size circuits, then
   $\text{BPP} \subset \bigcap_{\varepsilon > 0} \text{DTIME}(2^{n^\varepsilon})$.

2. circuits of size $2^{n^\varepsilon}$ for some $\varepsilon > 0$, then
   $\text{BPP} \subset \text{DTIME}(2^{(\log n)^c})$ for some constant $c$.

3. circuits of size $2^{\varepsilon n}$ for some $\varepsilon > 0$, then
   $\text{BPP} = \text{P}$.
   (We need to use $(\log n, m)$-design with $\ell = O(m)$.)
If some function in $E$ has circuit complexity $2^\Omega(n)$, then BPP = P.

- Similar claim as NW.3, but assuming hardness in the worst-case. NW needed hardness on the average.

- Convert mildly hard function $f$ to almost unpredictable function. Yao’s XOR-Lemma: $f(x_1) \oplus \cdots \oplus f(x_k)$ is hard to predict, when $x_i$ are independent.

- Use expanders to reduce the need for random bits.
Is circuit lower bound needed?

- $f$ is in BPP, if there is a randomised algorithm with error $\leq \frac{1}{3}$ on every input.

- $f$ is in promise-BPP, if there is a randomised algorithm with error $\leq \frac{1}{3}$ on some subset of inputs, and we do not care the acceptance prob. on other inputs.

[Impagliazzo & Kabanets & Wigderson, 2002]
Promise-BPP = P implies NEXP $\not\subset$ P/poly (circuit lower bound!).

[Kabanets & Impagliazzo, 2003]
BPP = P implies super-polynomial arithmetical circuit lower bound for NEXP.
Prerequisites of [KI03]

- [Valiant, 1979] Perm is $\#P$-complete
  - $\text{Perm}(A) = \sum \sigma \prod_{i=1}^{n} a_{i,\sigma(i)}$
  - $\#P$ is a class counting the number of solutions
- [Toda, 1991] $\text{PH} \subset P^{\#P}$
- [Impagliazzo & Kabanets & Wigderson, 2002]
  - $\text{NEXP} \subset P^{\text{poly}} \implies \text{NEXP} = \text{MA}$

If $\text{NEXP} \subset P^{\text{poly}}$, then

1. $\text{NEXP} = \text{MA} \subset \text{PH} \subset P^{\#P} \implies \text{Perm is NEXP-hard}$
2. $\text{Perm} \in \text{EXP} \subset \text{NEXP} \implies \text{Perm is NEXP-complete}$
Polynomial identity testing

- is testing whether a given polynomial is identically zero
- is in co-RP: take a random point and evaluate the polynomial. If the field is big enough, we get nonzero with high prob.

Can test whether a given *arithmetical circuit* $p_n$ computes Perm:
Input: $p_n$ on $n \times n$ variables, let $p_i$ be its restriction to $i \times i$ variables.

- test $p_1(x) = x$ (by the method above)
- for $i \in \{2, \ldots, n\}$, test $p_i(X) = \sum_{j=1}^{i} x_{1,j} p_{i-1}(X_j)$, where $X_j$ is the $j$-th minor

If all tests pass, then $p_n = \text{Perm}$. 
Circuit lower bounds from derandomization

- Suppose that *polynomial identity testing is in P*.

- If *Perm is computable by polynomial-size arithmetic circuits*, then *Perm ∈ NP*:
  1. guess the circuit for Perm
  2. verify its validity
  3. compute the result

- If *NEXP ⊂ P/poly*, then *Perm is NEXP-complete*.

Contradiction with nondeterministic time hierarchy theorem!
Main result of KI03

If $BPP = P$, or even $BPP \subset NSUBEXP = \bigcap_{\epsilon > 0} \text{NTIME}(2^{n^\epsilon})$, then

1. Perm does not have polynomial-size arithmetical circuits, or
2. $\text{NEXP} \not\subset P/poly$
Summary

- [NW94] Average circuit lower bounds imply derandomization
- [IW97] Worst-case circuit lower bounds imply derandomization
- [KI03] Derandomization implies circuit lower bounds