Parameterized Algorithms and Circuit Lower Bounds

Ryan Williams  Stanford
Two Important Areas of Research

Faster FPT Algorithms for NP

**Given:** Verifier $V(x, y)$ that reads a $k$-bit witness $y$, and runs in $(k + |x|)^{O(1)}$ time.

**Find:** a deterministic algorithm which
1. Runs in *less than* $2^k \cdot |x|^{O(1)}$ time
2. Given any input $x$, finds a $y$ so that $V(x, y)$ accepts (or concludes there is no $y$)

Circuit Lower Bounds

**Given:** Any NP problem (or any EXP<sup>NP</sup> problem!)

**Find:** Sequence of algorithms $\{A_n\}$ such that:
1. $|A_n| \leq n^k + k$
2. On all inputs $x$ of length $n$, $A_n(x)$ correctly solves the problem in $O(n^k)$ time.

*(Alternatively, prove that no such algorithms exist!)*
One May Look “Easier” Than The Other...

**Faster FPT Algorithms for NP**

Given: **Verifier** $V(x, y)$ that reads a $k$-bit witness $y$, and runs in $(k+|x|)^{O(1)}$ time.

Find: a deterministic algorithm which
1. Runs in *less than* $2^k \cdot |x|^{O(1)}$ time
2. Given any input $x$, finds a $y$ so that $V(x, y)$ accepts (or concludes there is no $y$)

**Circuit Lower Bounds**

Given: Any NP problem (or any EXP^NP problem!)

Find: Sequence of algorithms $\{A_n\}$ such that:
1. $|A_n| \leq n^k + k$
2. On all inputs $x$ of length $n$, $A_n(x)$ correctly solves the problem in $O(n^k)$ time.

(Alternatively, prove that no such algorithms exist!)
One May Look “Easier” Than The Other...

**Faster FPT Algorithms for NP**

- 3SAT: $O^*(1.308^n)$ time \[H12\]
- k-Path: $O^*(1.66^k)$ \[BHKP11\]
- Min-VC: $O^*(1.28^k)$ \[CKX06\]
  - degree-3: $O^*(1.17^k)$ \[M11\]
- 3-Coloring: $O^*(1.33^n)$ \[E04\]
- k-Coloring: $O^*(2^n)$ \[BHKP08\]
- ... many, many more!

**Circuit Lower Bounds**

*Given:* Any NP problem (or any $\text{EXP}^{\text{NP}}$ problem!)

*Find:* Sequence of algorithms $\{A_n\}$ such that:
1. $|A_n| \leq n^k + k$
2. On all inputs $x$ of length $n$, $A_n(x)$ correctly solves the problem in $O(n^k)$ time.

*(Alternatively, prove that no such algorithms exist!)*
One May Look “Easier” Than The Other...

<table>
<thead>
<tr>
<th>Faster FPT Algorithms for NP</th>
<th>Circuit Lower Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>- 3SAT: $O^*(1.308^n)$ time [H12]</td>
<td>- We don’t know how to get non-uniform</td>
</tr>
<tr>
<td>- k-Path: $O^*(1.66^k)$ [BHKP11]</td>
<td>algorithms that outperform these</td>
</tr>
<tr>
<td>- Min-VC: $O^*(1.28^k)$ [CKX06]</td>
<td><em>uniform</em> ones</td>
</tr>
<tr>
<td>degree-3: $O^*(1.17^k)$ [X10]</td>
<td>- Best lower bound known:</td>
</tr>
<tr>
<td>- 3-Coloring: $O^*(1.33^n)$ [BE05]</td>
<td>There is a function in NP that</td>
</tr>
<tr>
<td>- Max-2-SAT: $O^*(1.8^n)$ [W05]</td>
<td>requires circuits of size $5n + o(n)$</td>
</tr>
<tr>
<td>- ... many, many more!</td>
<td>- It is still open whether $\text{EXP}^{\text{NP}}$ has <em>polynomial-size circuits</em>!</td>
</tr>
</tbody>
</table>
Faster Algorithms $\implies$ Lower Bounds!

**Faster FPT Algorithms**

Deterministic algorithm for:
- CircuitSAT in $n^{O(1)} \frac{2^k}{k \log k}$ time (circuits with $k$ inputs, $n$ gates)
- FormSAT in $n^{O(1)} \frac{2^k}{k \log k}$ time
- ACC SAT in $n^{O(1)} \frac{2^k}{k \log k}$ time
- Given a circuit that’s either unsatisfiable, or has at least $2^{k-1}$ satisfying assignments, determine which is the case in $n^{O(1)} \frac{2^k}{k \log k}$ time
  (This problem is in BPP!)

**Circuit Lower Bounds**

Would imply:
- $\text{NEXP } \not\subset P/poly$ [W’10]
- $\text{NEXP } \not\subset \text{non-uniform NC}^1$
- $\text{NEXP } \not\subset \text{non-uniform ACC}$ [W’11]
- $\text{NEXP } \not\subset P/poly$
Circuit Satisfiability

Let $\mathbf{C}$ be a class of Boolean circuits

$\mathbf{C} = \{\text{Arbitrary Boolean formulas over AND and OR}\}$,
$\mathbf{C} = \{\text{Constant-depth circs}\}$, $\mathbf{C} = \{\text{Arbitrary Boolean circuits}\}$

**The C-SAT Problem:** Given a circuit $K(x_1, \ldots, x_k) \in \mathbf{C}$ with $k$ inputs and $n$ gates, is there an assignment $(a_1, \ldots, a_k) \in \{0,1\}^k$ such that $K(a_1, \ldots, a_k) = 1$?

$\mathbf{C}$-SAT is $\mathbf{NP}$-complete, for essentially all interesting $\mathbf{C}$

$\mathbf{C}$-SAT is solvable in $2^k \cdot n^{O(1)}$ time
**Theorem:** For many natural circuit classes $C$, 
**IF** $C$-SAT has a slightly faster parameterized algorithm, 
**THEN** $\text{NEXP}$ can’t be efficiently simulated by $C$-circuits.

**Proof Plan:** Assume we have two kinds of “good algorithms”

1. Slightly faster $C$-SAT algorithm

2. Every problem in $\text{NEXP}$ has a small $C$-circuit family

$$\forall \Pi \in \text{NEXP} \Rightarrow \Pi \text{ is solved by a family } \{C_n\}$$

Use them to simulate every $2^n$ time algorithm in $<< 2^n$ time

*False* by the time hierarchy theorem!
**Assume** (for an appropriate circuit class $C$)

- $C$-SAT with $n$ inputs and $n^{O(1)}$ size is in $O(2^n/n^{10})$ time
- $\text{NEXP}$ has polynomial-size circuits from class $C$

**Karp-Lipton, Meyer ‘80:** $P = NP \Rightarrow \text{EXP} \not\subset P/poly$

Assume $P = NP$ and $\text{EXP} \subset P/poly$

$\text{EXP} \subset P/poly \Rightarrow \exists$ polysize circuits $C$ encoding tableaus:

For every exponential-time machine $M$ and every string $x$, $C(M,x,i,j)$ prints the content of the $j$th cell of $M(x)$ in step $i$

The behavior of $M(x)$ can be simulated in $\Sigma_2 P$:

$(\exists C)(\forall i, j) [C \text{ makes consistent claims of cells } j-1, j, j+1 \text{ in steps } i-1, i, i+1]$

$\Rightarrow (\exists C)R(x,C)$, where $R(x,C)$ is a poly-time computable predicate

$\Rightarrow M(x)$ is in $P$. But then $\text{EXP} = P$, contradicting the time hierarchy.
Assume (for an appropriate circuit class $C$)

- $C$-SAT with $n$ inputs and $n^{O(1)}$ size is in $O(2^n/n^{10})$ time
- $\text{NEXP}$ has polynomial-size circuits from class $C$

**Impagliazzo-Kabanets-Wigderson ’01:**

$\text{NEXP} \subseteq \text{poly size } C \Rightarrow \exists$ circuit $D$ from class $C$ encoding tableaus:

For every non-deterministic $2^n$ time machine $M$ and every string $x$, $D(M,x,i,j)$ prints the content of the jth cell of $M(x)$ in step $i$.

The behavior of $M(x)$ can be simulated in $\Sigma_2 \text{P}$:

$(\exists D)(\forall i,j) \left[D \text{ makes consistent claims of cells } j-1, j, j+1 \text{ in steps } i-1, i, i+1\right]$ Express this efficiently as an $C$-SAT instance??

$\Rightarrow (\exists D)R(x,D)$, where $R(x,D)$ is an $O(2^n/n^{10})$ time predicate

$\Rightarrow M(x)$ is in non-deterministic $O(2^n/n^{10})$ time.

But then $\text{NTIME}[2^n] \subseteq \text{NTIME}[2^n/n^{10}]$, contradicting the non-deterministic time hierarchy!
Definition: The Circuit Class ACC

An ACC circuit family \( \{ C_n \} \) has the properties:

- Every \( C_n \) takes \( n \) bits of input and outputs a bit.
- There is a fixed \( d \) such that every \( C_n \) has depth \( d \).
- There is a fixed \( m \) such that the gates of \( C_n \) are AND, OR, NOT, MODm (unbounded fan-in).

\[ \text{MODm}(x_1, \ldots, x_t) = 1 \quad \text{iff} \quad \sum_i x_i \text{ is divisible by } m \]

\[ n = 11 \]
\[ \text{Size} = 5 \]
\[ \text{Depth} = 3 \]
**Definition: The Circuit Class ACC**

An **ACC** circuit family \( \{ C_n \} \) has the properties:

- Every \( C_n \) takes \( n \) bits of input and outputs a bit
- There is a fixed \( d \) such that every \( C_n \) has depth \( d \)
- There is a fixed \( m \) such that the gates of \( C_n \) are AND, OR, NOT, \( \text{MOD}_m \) (unbounded fan-in)

\[ \text{MOD}_m(x_1, \ldots, x_t) = 1 \iff \sum_i x_i \text{ is divisible by } m \]

**Remarks**

1. The default size of \( n^{\text{th}} \) circuit: **polynomial in** \( n \)
2. This is a **non-uniform** model of computation
   (Can compute some undecidable languages)
3. ACC circuits can be efficiently simulated by **constant-layer neural networks**
Where does ACC come from?

Sipser’s Program: Prove $P \neq NP$ by proving $NP \not\subset P/poly$.
The simple combinatorial nature of circuits should make it easier to prove impossibility results.

**Ajtai, Furst-Saxe-Sipser, Håstad (early 80’s)**
- $MOD2 \not\in AC0$ [i.e., $n^{O(1)}$ size $ACC$ with only AND, OR, NOT]

**Razborov, Smolensky (late 80’s)**
- $MOD3 \not\in (AC0$ with $MOD2$ gates)
- For $p \neq q$ prime, $MODp \not\in (AC0$ with $MODq$ gates)

**Barrington (late 80’s)** Suggested $ACC$ as the next step
- **Conjecture** Majority $\not\in ACC$

No real progress since then
Proof Strategy for ACC Lower Bounds

1. Show that faster ACC-SAT algorithms imply lower bounds against ACC

2. Design faster ACC-SAT algorithms!

**Theorem** For all $d, m$ there’s an $\varepsilon > 0$ such that ACC-SAT on circuits with $n$ inputs, depth $d$, MOD$m$ gates, and $2^n\varepsilon$ size can be solved in $2^n - \Omega(n^\varepsilon)$ time
Ingredients for Solving ACC SAT

Ingredients:

1. **A known representation of ACC**
   [Yao ’90, Beigel-Tarui’94] Every ACC function $f : \{0,1\}^n \rightarrow \{0,1\}$ can be expressed in the form
   $$f(x_1,\ldots,x_n) = g(h(x_1,\ldots,x_n))$$
   - $h$ is a multilinear polynomial with $K$ monomials and for all $(x_1,\ldots,x_n) \in \{0,1\}^n$, $h(x_1,\ldots,x_n) \in \{0,\ldots,K\}$
   - $K$ is not “too large” (*quasipolynomial in circuit size*)
   - $g : \{0,\ldots,K\} \rightarrow \{0,1\}$ can be an arbitrary function

2. **“Fast Fourier Transform” for multilinear polynomials:**
   Given a multilinear polynomial $h$ in its coefficient representation, the value $h(x)$ can be computed over all points $x \in \{0,1\}^n$ in $2^n \text{ poly}(n)$ time.
Theorem. For all $d$, $m$ there's an $\varepsilon > 0$ such that $\text{ACC}[m]$ SAT with depth $d$, $n$ inputs, $2^{n^\varepsilon}$ size can be solved in $2^n - \Omega(n^\varepsilon)$ time.

Proof:

Take the OR of all possible assignments to the first $n^\varepsilon$ inputs of $C$. For small $\varepsilon > 0$, evaluate $h$ on all $2^n - n^\varepsilon$ assignments in $2^n - n^\varepsilon$ poly$(n)$ time.
Fast Multipoint Circuit Evaluation Suffices for Circuit Lower Bounds!

**Theorem** If Multipoint Evaluation of $C$-circuits of size $s$ can be done in $2^n \poly(n) + \poly(s)$ time, then $C$-$SAT$ is in $o(2^n)$ time.

**Proof:**

- Take an OR of all assignments to the first $n^\varepsilon$ inputs of $C$.
- For small $\varepsilon > 0$, can evaluate circuit on all $2^{n - n^\varepsilon}$ assignments in $2^{n - n^\varepsilon} \poly(n) + \poly(2^{2n^\varepsilon})$ time.
Future Work

• Replace NEXP with simpler complexity classes
  Very recently: replaced with \((\text{NEXP} \cap \text{coNEXP})\)
  For EXP: may need to improve on exhaustive
  search for more complex problems

• Replace ACC with stronger circuits
  Design SAT algorithms for other circuit classes!
  Using strong versions of PCP Theorem:
  very weak derandomization suffices

• Find more connections between
  algorithms and lower bounds!
Thank you!