Announcements:

- Don’t forget: ps3 due 11/18
- Project presentations will start 12/2

We will post the schedule soon
How Fine-Grained Algorithms Can Imply Circuit Complexity Lower Bounds

Ryan Williams
Circuit Complexity: A Crash Course

(Ask questions!)
Circuits

For each $n$, have a circuit $C_n$ to be run on all inputs of length $n$.

Circuit model has “programs with infinite-length descriptions”.

$P/poly = \{ f : \{0, 1\}^* \to \{0, 1\} \text{ computable by a circuit family } \{C_n\} \}$

where for every $n$, the size of $C_n$ is at most $\text{poly}(n)$.

Each circuit is “small” relative to its number of inputs.
Concrete limitations on computing within the known universe

“Any computer solving most instances of this 10⁴-bit problem needs at least 10^{125} bits to be described”

Universe stores < 10^{125} bits  [Bekenstein ‘70s]  [Meyer-Stockmeyer ‘70s]
Functions with High Circuit Complexity

“Most” functions require huge circuits!

Theorem [Shannon ’49, Lupanov ‘58]
With high probability, a randomly chosen function $f : \{0,1\}^n \rightarrow \{0,1\}$ does not have circuits of size less than $2^n/n$ (and: every $f$ has a circuit of size about $2^n/n$)

Which “natural” functions exhibit this exponential behavior?
If there is a $f : \{0,1\}^* \rightarrow \{0,1\}$ computable in $2^{O(n)}$ time that does not have circuits of size at most $2^{n/100}$ then Randomized Time $\equiv$ Deterministic Time

Rough intuition: If $f$ “looks random” to all circuits, then $f$ can be used to replace true randomness in any computation!
Algorithms vs Circuit Families

T(n) \textit{time} on inputs of length n

BPP is in P/poly

C_n has \approx T(n) \textit{size}

There is a family where every C_n has \approx \text{size } n

No algorithm at all!

Some undecidable problems are in P/poly

EXPONENTIAL TIME (2^n \textit{steps})

EXP is in P/poly is open!

Conjecture: NP \not\subseteq P/poly

every C_n has \approx n^2 \textit{size} !!
Here endeth the Crash Course
Two Difficult Areas of Research

Fine-Grained Improvements for Solving NP Problems

Given: Verifier \( V(x, y) \) which reads \( w(|x|) \) bits of witness \( y \), runs in \( t(|x|) \) time.

Find: Deterministic or Randomized Algorithm which:

1. Runs in \emph{less than} \( 2^{w(|x|) \cdot t(|x|)} \) time
2. Given any input \( x \), finds a witness \( y \) such that \( V(x, y) \) accepts (or conclude none)

Circuit Complexity (Non-Uniform Algorithms)

Given: Any \textbf{NP} problem \( \Pi \) (or \textbf{NEXP} problem!)

Find: Sequence of algorithms \( \{A_n\} \) such that for some \( k \):

1. \( |A_n| \leq n^k + k \)
2. On all inputs \( x \) of length \( n \), \( A_n(x) \) correctly solves \( \Pi \) on \( x \) in \( O(n^k) \) time.

\textbf{Prove that no such sequences of algorithms exist for} \( \Pi \)
One Seems Easier Than The Other...

Fine-Grained Improvements for Solving NP Problems

Given: Verifier $V(x, y)$ which reads $w(|x|)$ bits of witness $y$, runs in $t(|x|)$ time.

Find: Deterministic or Randomized Algorithm which:

1. Runs in less than $2^{w(|x|)}t(|x|)$ time
2. Given any input $x$, finds a witness $y$ such that $V(x,y)$ accepts (or conclude none)

Circuit Complexity (Non-Uniform Algorithms)

Given: Any NP problem $\Pi$ (or NEXP problem!)

Find: Sequence of algorithms $\{A_n\}$ such that for some $k$:

1. $|A_n| \leq n^k + k$
2. On all inputs $x$ of length $n$, $A_n(x)$ correctly solves $\Pi$ on $x$ in $O(n^k)$ time.

Prove that no such sequences of algorithms exist for $\Pi$
One Seems Easier Than The Other...

<table>
<thead>
<tr>
<th>Fine-Grained Improvements</th>
<th>Circuit Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-SAT: O(1.308^n) time</td>
<td>Given: Any NP problem Π (or NEXP problem!)</td>
</tr>
<tr>
<td>k-SAT: O(2^n - n/k)</td>
<td>Find: Sequence of algorithms {A_n} such that for some k:</td>
</tr>
<tr>
<td>Hamiltonian Path: O(1.66^n)</td>
<td>1.</td>
</tr>
<tr>
<td>Vertex Cover: O(1.3^n)</td>
<td>2. On all inputs x of length n, A_n(x) correctly solves Π on x in O(n^k) time.</td>
</tr>
<tr>
<td>on degree-3 graphs: O(1.09^n)</td>
<td><strong>Prove that no such sequences of algorithms exist for Π</strong></td>
</tr>
<tr>
<td>Max-2-SAT: O(1.8^n)</td>
<td></td>
</tr>
<tr>
<td>3-Coloring: O(1.33^n)</td>
<td></td>
</tr>
<tr>
<td>k-Coloring: O(2^n)</td>
<td></td>
</tr>
</tbody>
</table>
One Seems Easier Than The Other...

**Fine-Grained Improvements**
- 3-SAT: $O(1.308^n)$ time
- k-SAT: $O(2^{n\cdot n/k})$
- Hamiltonian Path: $O(1.66^n)$
- Vertex Cover: $O(1.3^n)$
  - on degree-3 graphs: $O(1.09^n)$
- Max-2-SAT: $O(1.8^n)$
- 3-Coloring: $O(1.33^n)$
- k-Coloring: $O(2^n)$

**Circuit Complexity**
- For all these algorithms on the LHS, we don’t know how to get non-uniform algorithms (circuits) that are any better
- Best lower bound known: There is a function in NP that requires circuits of size $5n + o(n)$
- Cannot yet rule out that \( \text{NEXP} \) is in \( \text{P/poly} \)...
Faster “Algorithms for Circuits”
[R.W. ’10,’11]

Deterministic algorithms for:
- Circuit SAT in $O(2^n/n^{10})$ time with $n$ inputs and $n^k$ gates
- Formula SAT in $O(2^n/n^{10})$
- $C$-SAT in $O(2^n/n^{10})$
- Given a circuit of $n^k$ size that’s either $\textit{UNSAT}$, or has $\geq 2^{n-1}$ SAT assignments, determine which in $O(2^n/n^{10})$ time (Easily solved w/ randomness!)

No “Circuits for NEXP”

Would imply:
- $\text{NEXP} \not\subset \text{P/poly}$
- $\text{NEXP} \not\subset \text{Poly-size Formulas}$
- $\text{NEXP} \not\subset \text{poly-size } C$

$\text{NEXP} \not\subset \text{P/poly}$
Better “Algorithms for Circuits”
[Murray-W. ’18]
Det. algorithm for some $\epsilon > 0$:
- Circuit SAT in $O(2^{n-n^\epsilon})$ time with $n$ inputs and $2^{n^\epsilon}$ gates
- Formula SAT in $O(2^{n-n^\epsilon})$
- $C$-SAT in $O(2^{n-n^\epsilon})$
- Given a circuit of $2^{n^\epsilon}$ size that’s either UNSAT, or has $\geq 2^{n-1}$ SAT assignments, determine which in $O(2^{n-n^\epsilon})$ time (Easily solved w/ randomness!)

No “Circuits for Quasi-NP”
Would imply:
- $\text{NTIME}[n^{\text{polylog } n}] \not\subset \text{P/poly}$
- $\text{NTIME}[n^{\text{polylog } n}] \not\subset \text{NC1}$
- $\text{NTIME}[n^{\text{polylog } n}] \not\subset C$

$\text{NTIME}[n^{\text{polylog } n}] \not\subset \text{P/poly}$
Even Faster $\implies$ “Easier” Functions

Fine-Grained SAT Algorithms
[Murray-W. ’18]
Det. algorithm for some $\epsilon > 0$:
- Circuit SAT in $O(2^{(1-\epsilon)n})$ time on $n$ inputs and $2^{\epsilon n}$ gates
- Formula SAT in $O(2^{(1-\epsilon)n})$
- $C$-SAT in $O(2^{(1-\epsilon)n})$
- Given a circuit of $2^{\epsilon n}$ size that’s either UNSAT, or has $\geq 2^{n-1}$ SAT assignments, determine which in $O(2^{(1-\epsilon)n})$ time (Easily solved w/ randomness!)

No “Circuits for NP”
Would imply:
- $NP \not\subset SIZE(n^k)$ for all $k$
- $NP \not\subset Formula\text{-}SIZE(n^k)$
- $NP \not\subset C\text{-}SIZE(n^k)$ for all $k$
Why on Earth would it be true?

∃ ∀ "Non-Trivial" Circuit Analysis Algorithm

SAT? YES/NO

∃ "interesting" f

∀

Circuit Lower Bounds
Concrete Lower Bounds From Algs!

Thm [R.W.’11]: NEXP \not\subset ACC^0

Thm [Murray-W’18]: NTIME[\(n^{\text{poly} \log n}\)] \not\subset ACC^0

NEXP = NTIME[2^{n^{O(1)}}]

ACC^0: polynomial size, constant depth circuits with AND, OR, and MOD[m] gates for some constant m.

A simple but Annoying Circuit Class to prove lower bounds for (proposed in 1986 by Barrington)
How It Was Proved

Let $\mathbb{C}$ be a “typical” circuit class (like $\text{ACC}^0$)

**Thm A [W’11]:**

If for all $k$, $\mathbb{C}$-SAT on $n^k$-size is in $O(2^n/n^k)$ time, then $\text{NEXP}$ does not have poly-size $\mathbb{C}$-circuits.

**Thm B [W’11]:**

$\exists \varepsilon$, $\text{ACC}^0$-SAT on $2^{n^\varepsilon}$ size is in $O(2^{n-n^\varepsilon})$ time.

An inefficiency!

Theorem B gives a much stronger algorithm than is needed in Theorem A.

This is exactly the starting point of [Murray-W’18]...
More on Theorem A

Let $\mathbb{C}$ be some circuit class (like $\text{ACC}^0$)

Thm A [W’11]:

If for all $k$, $\mathbb{C}$-SAT on $n^k$-size is in $O(2^n/n^k)$ time,
then $\text{NEXP}$ does not have poly-size $\mathbb{C}$-circuits.

Idea. Show that if we assume both:

(1) $\text{NEXP}$ has poly-size $\mathbb{C}$-circuits, and

(2) a faster $\mathbb{C}$-SAT algorithm

Then we can show $\text{NTIME}[2^n] \subseteq \text{NTIME}[o(2^n)]$
Proof Sketch of Theorem A

Idea. Assume

(1) NEXP has poly-size $\mathbb{C}$-circuits, and
(2) a faster $\mathbb{C}$-SAT algorithm

Show that $\text{NTIME}[2^n] \subseteq \text{NTIME}[o(2^n)]$

Take any problem L in nondeterministic $2^n$ time. Given an input $x$, we “compute” L on $x$ by:
1. Guessing some witness $y$ of $O(2^n)$ length.
2. Checking $y$ is a witness for $x$ in $O(2^n)$ time.
Proof Sketch of Theorem A

Idea. Assume

(1) NEXP has poly-size \(\mathbb{C}\)-circuits, and
(2) a faster \(\mathbb{C}\)-SAT algorithm

Show that \(\text{NTIME}[2^n] \subseteq \text{NTIME}[o(2^n)]\)

Take any problem \(L\) in nondeterministic \(2^n\) time.
Given an input \(x\), we will “compute” \(L\) on \(x\) by:
1. Guessing some witness \(y\) of \(o(2^n)\) length.
2. Checking \(y\) is a witness for \(x\) in \(o(2^n)\) time.
Proof Sketch of Theorem A

Idea. Assume

1. NEXP has poly-size \( \mathbb{C} \)-circuits, and
2. a faster \( \mathbb{C} \)-SAT algorithm

Show that \( \text{NTIME}[2^n] \subseteq \text{NTIME}[o(2^n)] \)

Take any problem \( L \) in nondeterministic \( 2^n \) time. Given an input \( x \), we will “compute” \( L \) on \( x \) by:

1. Guessing some witness \( y \) of \( o(2^n) \) length.
2. Checking \( y \) is a witness for \( x \) in \( o(2^n) \) time.
Proof Sketch of Theorem A

Idea. Assume

(1) NEXP has poly-size \( \mathbb{C} \)-circuits, and

(2) a faster \( \mathbb{C} \)-SAT algorithm

Show that \( \text{NTIME}[2^n] \subseteq \text{NTIME}[o(2^n)] \)

Take any problem \( L \) in nondeterministic \( 2^n \) time. Given an input \( x \), we will “compute” \( L \) on \( x \) by:

1. Guessing some witness \( y \) of \( o(2^n) \) length.
2. Checking \( y \) is a witness for \( x \) in \( o(2^n) \) time.
Guessing Short Witnesses

1. Guess a witness \( y \) of \( o(2^n) \) length.

**Easy Witness Lemma [IKW’02]:**
If \( \text{NEXP} \) has polynomial-size circuits,
then all \( \text{NEXP} \) problems have “easy witnesses”

**Def.** An \( \text{NEXP} \) problem \( L \) has easy witnesses if
\( \forall \) Verifiers \( V \) for \( L \) and \( \forall x \in L \),
\( \exists \) poly(\( |x| \))-size circuit \( D_x \) such that \( V(x,y) \) accepts,
where \( y = \text{Truth Table of circuit } D_x \).

1’. Guess poly(\( |x| \))-size circuit \( D_x \).
Verifying Short Witnesses

2. Check $y$ is a witness for $x$ in $o(2^n)$ time.

Assuming NEXP has polynomial-size circuits, “easy witnesses” exist for every verifier $V$. We choose a $V$ for an NEXP-complete $L$ so that

\[
\text{Checking a witness for } x \equiv \\
\text{Solving a } \mathbb{C}\text{-UNSAT instance with } poly(|x|) \text{ size} \\
\text{and } n = |x| + O(\log|x|) \text{ inputs}
\]

Then, $2^n / n^k$ time for $\mathbb{C}$-UNSAT $\Rightarrow o(2^{|x|})$ time
Verifying Short Witnesses

Assuming NEXP has polynomial-size circuits, “easy witnesses” exist for every verifier V. We choose a V for an NEXP-complete L so that

2. Check $y$ is a witness for $x$ in $o(2^n)$ time.

Checking a witness for $x$

$\equiv$

Distinguishing *unsatisfiable* circuits from those with *many* satisfying assignments

(Uses the PCP Theorem!)
Proof Sketch of Theorem A

Idea. Assume

(1) NEXP has poly-size $\mathbb{C}$-circuits, and
(2) a faster $\mathbb{C}$-SAT algorithm

Show that $\text{NTIME}[2^n] \subseteq \text{NTIME}[o(2^n)]$

Take any problem L in nondeterministic $2^n$ time. Given an input $x$, we will “compute” L on $x$ by:

1. Guessing a circuit $D_x$ of poly($|x|$) size
2. Checking $D_x$ encodes a witness for $x$ in $o(2^n)$ time
End