Algorithms for Circuits and Circuits for Algorithms

Ryan Williams  Stanford
A view of algorithms and complexity – from 50,000 ft

- Algorithm designers
- Complexity theorists

- What makes some problems easy to solve? When can we find an efficient algorithm?
- What makes other problems difficult? When can we prove that a problem is not easy?

(When can we prove a lower bound on the resources needed to solve a problem?)
The tasks of the algorithm designer and the complexity theorist appear to be inherently opposite ones.

- **Algorithm designers**
- **Complexity theorists**

Furthermore, it is generally believed that **lower bounds** are *harder* than **algorithm design**

- In algorithm design, we “merely” have to find a single clever algorithm that solves a problem well
- In lower bounds, we must reason about *all possible* algorithms, and argue that none of them work well

This belief is strongly reflected in the literature
“Duality” Between Algorithms and Lower Bounds

**Thesis:** Algorithm design is *at least as hard as* proving lower bounds.

Not only are there deep links between the two... but in some cases they are the “same”!

A typical theorem from Algorithm Design: “Here is an algorithm $A$ that solves my pet problem, on all possible instances of the problem" 

A typical theorem from Lower Bounds: “Here is a proof $P$ that my problem cannot be solved, on all possible algorithms from some class"
“Duality” Between Circuit Analysis Algorithms and Circuit Lower Bounds

**Thesis:** Algorithm design is *at least as hard as* proving lower bounds.
Outline

• Circuit Analysis Algorithms (Algorithms for Circuits)

• Circuit Complexity (Circuits for Algorithms/Lower Bounds)

• Connections

• Outline of a Connection
Circuit Analysis problems are often computational problems on circuits given as input:

**Input:** A logical circuit $C = \ldots$

**Output:** Some property of the function computed by $C$

**Canonical Example:** Circuit Satisfiability Problem (Circuit SAT)

**Input:** Logical circuit $C$

**Decide:** Is the function computed by $C$ the “all-zeros” function?

Of course, Circuit SAT is NP-complete

But we can still ask if there are any algorithms solving Circuit SAT that are faster than the obvious “brute-force” algorithm which tries all $2^n$ input settings to the $n$ inputs of the circuit.
Let $\mathcal{C}$ be a class of Boolean circuits

$\mathcal{C} = \{\text{formulas}\}$, $\mathcal{C} = \{\text{arbitrary circuits}\}$, $\mathcal{C} = \{\text{CNF formulas}\}$

**The $\mathcal{C}$-SAT Problem:**
Given a circuit $K(x_1, \ldots, x_n) \in \mathcal{C}$, is there an assignment $(a_1, \ldots, a_n) \in \{0,1\}^n$ such that $K(a_1, \ldots, a_n) = 1$?

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

$\mathcal{C}$-SAT is solvable in $O(2^n |K|)$ time
where $|K|$ is the size of the circuit $K$
Circuit SAT Algorithms

For simple enough circuits, we know faster than $2^n$ algorithms

- **3-SAT**: $1.308^n$
- **4-SAT**: $1.469^n$
- **k-SAT**: $2^n - n/k$ time algorithms

[many authors ..., Hertli ‘11]

All known $c^n$ time algorithms for k-SAT have the property that, as $k \to \infty$, the constant $c \to 2$ (Local search, Backtracking, etc)

**Strong ETH**: $\forall \delta < 1, \exists k \geq 3$ s.t. $k$-SAT requires $2^{\delta n}$ time

**ETH**: $\exists \delta > 0$ s.t. 3-SAT requires $2^{\delta n}$ time
Circuit SAT Algorithms

For simple enough circuits, we know of faster algorithms

- **AC0-SAT** Constant-depth AND/OR/NOT

  [IMP ‘12] AC0-SAT in $2^n - \frac{n}{(c \log s)^d}$ time where $d = \text{depth}$, $s = \text{size}$
Circuit SAT Algorithms

For simple enough circuits, we know of faster algorithms

- **ACC-SAT**
  
  Constant-depth AND/OR/NOT/MODm
  
  \[
  \text{MOD}_6(x_1, \ldots, x_t) = 1 \quad \text{iff} \quad \sum_i x_i \text{ is divisible by 6}
  \]

  [W ‘11] ACC-SAT is in \(2^n - n^e\) time for circuits of size \(2^{n^{o(1)}}\)
Algorithm for ACC-SAT \[W’11\]

The ingredients:

1. A known representation of ACC via polynomials

   [Yao ’90, Beigel-Tarui’94] Every ACC function $f : \{0,1\}^n \rightarrow \{0,1\}$ can be put 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 over all 0-1 assignments, $h(x_1,\ldots,x_n) \in \{0,\ldots,K\}$
   - $K$ is not “too large” (quasipolynomial in circuit size)
   - $g : \{0,\ldots,K\} \rightarrow \{0,1\}$ can be arbitrary.

2. “Fast Fourier Transform” for multilinear polynomials to quickly evaluate $h$ on all its possible assignments
**Theorem** For all \( d \), there’s an \( \varepsilon > 0 \) such that ACC-SAT with depth \( d \), \( n \) inputs, \( 2^{n^\varepsilon} \) size can be solved in \( 2^n - \Omega(n^\varepsilon) \) time.

**Proof:**

1. Take an OR of all assignments to the first \( n^\varepsilon \) inputs of \( C \).
2. Beigel-Tarui:
   - For small \( \varepsilon > 0 \), evaluate on all \( 2^n - n^\varepsilon \) assignments in \( 2^n - n^\varepsilon \) poly\((n)\) time.

XXX
For simple enough circuits, we know of faster algorithms

- **ACC-THR-SAT** Constant-depth AND/OR/NOT/MOD_m with a layer of linear threshold fns at the bottom

[W ‘14] **ACC-THR-SAT** is in $2^n - n^e$ time for circuits of size $2^{n^{o(1)}}$
Circuit SAT Algorithms

• **DeMorgan-Formula-SAT**
  Formulas over AND/OR/NOT, each gate has fan-in at most 2
  [Santhanam ’10, CKKSZ ’14]
  DM-Formula-SAT is in $2^{n-n^e}$ time for formulas of size $< n^{2.99}$

• Formulas over AND/OR/NOT/XOR with fan-in two
  [Seto-Tamaki ’12, CKKSZ ’14]
  Formula-SAT is in $2^{n-n^e}$ time for formulas of size $< n^{1.99}$

• **Circuit-SAT**
  Generic circuits over AND/OR/NOT, fan-in 2

*Can we improve on $O(2^n s)$ time??*
Circuit Approximation Probability Problem

Let $C$ be a class of Boolean circuits

\textbf{\textit{$C$-CAPP:}}

Given a circuit $K(x_1,\ldots,x_n) \in C$, output $v$ such that

$$|v - \Pr_{x}[K(x) = 1]| < 1/10$$

Related to Pseudorandom Generators and Derandomization

[AW’85, Nisan’91, TX’13] AC0-CAPP is in $n^{\tilde{O}(\log^{d+4}s)}$ time

(n = inputs, s = size, d = depth)

[GMR’12] CNF-CAPP is in $n^{O(\log \log n)}$ time for poly(n) clauses

[IMZ’12] DM-Formula-CAPP: $2^{n^{e}}$ time for formulas of size $< n^{2.99}$

Formula-CAPP: $2^{n^{e}}$ time for formulas of size $< n^{1.99}$

Uses old techniques from lower bounds!
Circuit Analysis problems can also analyze functions directly:

Canonical Example:

Circuit Min [Yablonski ’59, KC’00]

Input: $2^n$-bit truth table of $f : \{0,1\}^n \to \{0,1\}$, $s \in \{1,\ldots,2^n\}$,

Decide: Is the minimum size of a circuit computing $f$ at most $s$?

It is widely conjectured that Circuit Min is not in P
If in P: Would contradict conventional wisdom in cryptography
Known: [Masek’79, AHMPS’08] DNF Min is NP-complete (uses lower bounds on DNF!)

Open: Is the Circuit Min problem NP-complete?
Open: Find any improvement over exhaustive search
Outline

• Circuit Analysis Algorithms (Algorithms for Circuits)
• Circuit Complexity (Circuits for Algorithms/Lower Bounds)
• Connections
• Outline of a Connection
Circuit Complexity of Infinite Languages

Allow a distinct logical circuit $A_n$ to run on inputs of length $n$

$P/poly = \text{Class of problems solvable with a circuit family } \{A_n\}$

such that $(\exists k \geq 1)(\forall n)$, the size of $A_n$ is at most $n^k$

This is an infinite computational model

$\{1^n \mid \text{the } n\text{th Turing machine halts on blank tape}\} \in P/poly$

The usual techniques of computability theory are essentially powerless for understanding $P/poly$
Circuits for Algorithms

\[ \text{P/poly} = \text{Problems solvable with a circuit family } \{A_n\} \]
where the \text{number of gates of } A_n \leq n^k

Most Boolean functions require huge circuits!

\textbf{Theorem [Shannon ‘49]} W.h.p., a randomly chosen function \( f : \{0,1\}^n \rightarrow \{0,1\} \) requires a circuit of size at least \( 2^n/n \)

\textbf{What “normal” algorithms can be simulated in P/poly?}

Non-uniformity can be \textbf{very} powerful!

\textbf{MAJOR OPEN PROBLEM:} Is \( \text{NEXP} \subset \text{P/poly} \)?

Can all problems with \textit{exponentially long} solutions be solved with \textit{polynomial size} circuit families?

Given “infinite” preprocessing time, can one construct small size circuits solving NEXP problems?
Circuits for Algorithms

These questions have interesting consequences, regardless of how they’re resolved.

[Karp-Lipton-Meyer ‘80]  \( \text{EXP} \subset \text{P/poly} \Rightarrow \text{P} \neq \text{NP} \)

[Folklore] Theorem

If every problem in \( 2^{O(n)} \) time has circuits smaller than \( 1.99^n \) size for infinitely many input lengths, then \( \text{P} \neq \text{NP} \)

[BFNW ’90]  \( \text{EXP} \not\subset \text{P/poly} \Rightarrow \text{Pseudorandom generators} \)

Theorem [Impagliazzo-Wigderson ‘97]

If some problem in \( 2^{O(n)} \) time needs circuits larger than \( 1.99^n \) for almost all input lengths, then \( \text{P} = \text{BPP} \)

Theorem [IKW ‘01]  \( \text{NEXP} \not\subset \text{P/poly} \Rightarrow \text{Can simulate MA in NP} \)
Outline

• Circuit Analysis Algorithms (Algorithms for Circuits)

• Circuit Complexity (Circuits for Algorithms/Lower Bounds)

• Connections

• Outline of a Connection
Connections

Algorithms for Circuits (Circuit Analysis):
*Designing faster circuit-analysis algorithms*

Circuits for Algorithms (Circuit Complexity):
*Designing small circuits to simulate complex algorithms*

Can we use one of these tasks to inform the other task?

Can interesting circuit-analysis algorithms tell us something about the *limitations* of circuits?
Can interesting circuit-analysis algorithms tell us something about the *limitations of circuits*?

[Karp-Lipton-Meyer ‘80]
Suppose we had extremely efficient circuit-analysis algorithms
Then we could prove that there are problems
solvable by an algorithm in $2^n$ time that are not in $P/poly$

$P = NP$
(Circuit SAT in P)
(Circuit Min in P)

$\Rightarrow$ There are problems in $\text{EXP}$
which are not in $P/poly$

This is an interesting conditional statement, but it has limited utility, since we do not believe the hypothesis is true!
Can interesting circuit-analysis algorithms tell us something about the *limitations of circuits*?

[Kabanets-Cai ’00]

Studied consequences of Circuit Minimization in P

**Given:** Truth table of a Boolean function $f$, parameter $s$

**Question:** Does $f$ have a circuit of size at most $s$?

**If Circuit Minimization is in P, then**

1. $\text{EXP}^{\text{NP}}$ requires maximum circuit complexity  
   (*new circuit lower bounds*)
2. $\text{BPP} = \text{ZPP}$
3. Discrete Log, Factoring, Graph Iso are in BPP [Allender-Das]
4. No strong pseudorandom functions (or PRGs)
SAT and Lower Bounds

A slightly faster algorithm for $\mathcal{C}$-SAT
\[ \Rightarrow \] Lower bounds against $\mathcal{C}$ circuits

\[ O\left(\frac{2^n}{n^{10}}\right) \]
Faster Circuit-SAT algorithms uncover a “weakness” in computing with circuits
Faster Circuit-SAT algorithms uncover a “strength” in less-than-exponential algorithms!
Faster “Algorithms for Circuits”  
[W’10, W’11]

An algorithm for:
- Circuit SAT in $O(2^n/n^{10})$ time  
  (n inputs and $n^k$ gates)
- Formula SAT in $O(2^n/n^{10})$
- ACC SAT in $O(2^n/n^{10})$
- Given a circuit C that’s either 
  $UNSAT$, or has $\geq 2^{n-1}$ satisfying assignments, 
  determine which, in $O(2^n/n^{10})$ time  
  (A Promise-BPP problem)

No “Circuits for Algorithms”

Would imply:
- $NEXP \not\subseteq P/poly$
- $NEXP \not\subseteq (non-uniform) NC^1$
- $NEXP \not\subseteq ACC$
  
  $NEXP \not\subseteq P/poly$
Can interesting circuit lower bounds tell us something about circuit-analysis algorithms?

Many well-known connections between circuit lower bounds and derandomization (CAPP)

For restricted classes of circuits, one can sometimes adapt the techniques used to prove circuit lower bounds to derive faster SAT algorithms for those circuits.

Example: Boolean formulas over AND, OR, NOT, fan-in 2

[Subbotovskaya ‘61] MOD2 on n bits cannot be computed with $n^{1.4999}$ size Boolean formulas with AND, OR, NOT gates.

[Santhanam’11] Satisfiability of $O(n)$-size Boolean formulas with AND and OR gates can be solved in $o(2^n)$ time.
Can interesting circuit lower bounds tell us something about circuit-analysis algorithms?

[Impagliazzo-Kabanets-Wigderson’02] There are “non-trivial” CAPP algorithms if and only if NEXP is not in P/poly.

What does non-trivial mean? We call a nondeterministic algorithm A “non-trivial for CAPP” if:
- A(C) runs in $2^{n^\varepsilon}$ time for all $\varepsilon$ on a Boolean circuit C of size $n$
- For infinitely many $n$, there’s $\geq 1$ accepting computation path on all C of size $n$, and every such computation path outputs a value $\nu$ within 1/10 of the acceptance probability of C.
Can interesting circuit lower bounds tell us something about circuit-analysis algorithms?

[W ’13] There are “non-trivial” algorithms for Circuit Min
IF AND ONLY IF
NEXP is not in P/poly

What does non-trivial mean?
We call an algorithm A “non-trivial for Circuit Min” if for all k,
- A(f) runs in poly(2^n) time on a Boolean function f of 2^n bits
- For infinitely many n,
  - There is a Boolean function f of 2^n bits, A(f) outputs 1
  - For all f computable with an n^k size circuit, A(f) outputs 0

Corollary: An “equivalence” between Circuit Min and CAPP
Outline

• **Circuit Analysis Algorithms**
  (Algorithms for Circuits)

• **Circuit Complexity**
  (Circuits for Algorithms/Lower Bounds)

• **Connections**

• **Outline of a Connection**
Outline of a Connection

For every “robust” circuit class $\mathcal{C}$, Faster $\mathcal{C}$-SAT algorithms imply Lower Bounds against $\mathcal{C}$

**Theorem**  If $\mathcal{C}$-SAT of circuits with $n$ inputs and $n^{O(1)}$ size is in $O(2^n/n^{10})$ time, then $\text{NEXP} \not\subseteq \mathcal{C}$
Assume

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

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

Assume $P = NP$ and $EXP \subset P/poly$

$EXP \subset P/poly \Rightarrow \exists$ polysize circuits $C$ encoding accepting computation tableaus:

For every exptime machine $M$ and every string $x$,

$C(M,x,i,j)$ prints the content of the jth cell of $M(x)$ in step $i$

The behavior of $M(x)$ can be simulated as follows:

$(\exists C)(\forall i, j) [C$ makes consistent claims of cells $j, j+1, j+2$ in steps $i, i+1]$

This part is computable in coNP

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

This is an NP problem

$P = NP \Rightarrow M(x)$ is in $P$. But then we contradict the time hierarchy!
Assume

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

Impagliazzo-Kabanets-Wigderson ’01:
NEXP ⊂ P/poly ⇒ ∃ circuits C encoding accepting tableaus:
For every nondeterministic $2^n$ time machine M and every string x, C(M, x, i, j) prints the jth cell of M(x) in step i, for some accepting path

The behavior of M(x) can be simulated as follows:
$(\exists C)(\forall i, j) [C \text{ makes consistent claims of cells } j, j+1, j+2 \text{ in steps } i, i+1]$

Express this part as a Circuit-SAT instance with n variables??

⇒ (The major difficulty:
⇒ M The number of inputs to the Circuit-SAT instance would be $\approx 2n$

But then NTIME[$2^n$] ⊆ NTIME[$2^n/n^{10}$], contradicting the nondeterministic time hierarchy!
Assume

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

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

\[ \text{NEXP} \subset \text{P/poly} \Rightarrow \exists \text{ circuits C efficient encoding tableaus} \]

For every nondeterministic \( 2^n \) time machine \( M \) and every string \( x \), \( C(x,i) \) prints \( O(n) \) bits of content representing \( M(x) \) in step \( i \)

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

\[ (\exists \ C)(\forall i) [\text{C makes consistent claims about the content of step } i] \]

Can express this as a Circuit SAT instance with \( n + 5 \log n \) inputs

\[ \Rightarrow (\exists \ C)R(x,C), \text{ where } R(x,C) \text{ is an } O(2^n/n^5) \text{ time predicate} \]

\[ \Rightarrow M(x) \text{ is in nondeterministic } O(2^n/n^5) \text{ time.} \]

But then NTIME[\( 2^n \)] \( \subset \) NTIME[\( 2^n/n^5 \)], contradicting the nondeterministic time hierarchy!
Future Progress

• Precisely *which* circuit lower bounds are implied by *which* faster circuit analysis algorithms?

Open (Ended): *Is there a circuit-analysis problem such that mild improvements over brute force imply \( \text{EXP} \not\subset \text{P/poly} \)?

• What prevents us from solving CAPP in \( O(2^n/n^{10}) \) time on circuits with \( n \) inputs?
  We believe \( P = \text{BPP} \), so this should definitely be possible...

Open: *Can Boolean formulas of size \( s \) be evaluated on all \( n \)-variable assignments in \( \text{poly}(s) + 2^n\text{poly}(n) \) time?*

• Find more connections! (They are there.)
Thank you!