Slides From the Previous Lectures

- https://people.csail.mit.edu/rrw/winter-school-lec1.pdf
- https://people.csail.mit.edu/rrw/winter-school-lec2.pdf

Recap: E^{NP} needs exp. size ACC circuits

Design a faster ACC-SAT algorithm

The Algorithm: For every d, m, there is an $\varepsilon > 0$ such that ACC-SAT on circuits with n inputs, 2^{n^ε} size, depth d, and MODm gates is solvable in 2^{n-n^ε} time

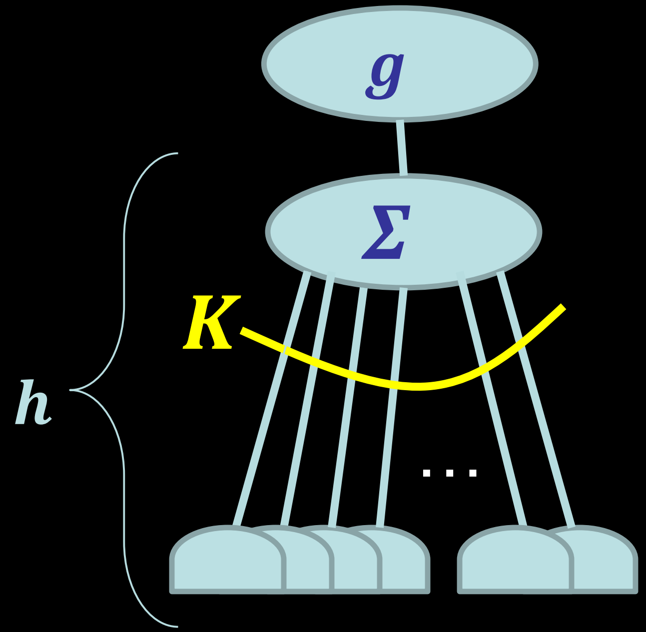
Show that faster ACC-SAT algorithms imply lower bounds against ACC

The LB Connection: If ***C***-SAT on circuits with n inputs and 2^{n^ε} size is in $O(2^n/n^{10})$ time, then **E^{NP} doesn't have 2^{n^ε} size *C*-circuits.**

Algorithm for SAT on ACC Circuits

Ingredients:

1. **Old representation** [Yao'90, Beigel-Tarui'94,Green et al'95]
   For every ACC function $f : \{0,1\}^\star \rightarrow \{0,1\}$ and every $\boldsymbol{n}$, we can write $f_n : \{0,1\}^n \rightarrow \{0,1\}$ as:
   $$\boldsymbol{f_n(x_1, \ldots, x_n) = g(h(x_1, \ldots, x_n))}, \text{ where}$$
   - $\boldsymbol{h}$ is a multilinear polynomial of at most K monomials, $h(a) \in \{0, \ldots, K\}$ for all $a \in \{0,1\}^n$
   - K is not "too large" *(quasi-polynomial in circuit size)*
   - $\boldsymbol{g} : \{0, \ldots, K\} \rightarrow \{0,1\}$ is a fixed "simple" function
2. **"Fast Fourier Transform" for multilinear polynomials:**
   Given a multilinear polynomial h in its coefficient representation, the value $h(a)$ can be computed over all points $a \in \{0,1\}^n$ in $\boldsymbol{2^n\, poly(n)}$ time.



[Chen-Papakonstantinou'19]

$$K \leq 2^{(\log s)^{O(dr)}}$$

where s = size, d = depth, r = # prime divisors of m

Fast Multipoint Evaluation

Theorem: Given the $\mathbf{2^n}$ coefficients of a multilinear polynomial $\boldsymbol{h}$ in $\boldsymbol{n}$ variables, $\boldsymbol{h(a)}$ can be computed on all points $\boldsymbol{a} \in \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}}$ in $\mathbf{2^n}\, \boldsymbol{poly(n)}$ time.

Can write: $\boldsymbol{h(x_1, \ldots, x_n) = x_1\, h_1(x_2, \ldots, x_n) + h_2(x_2, \ldots, x_n)}$

Want a 2^n table T that contains the value of h on all 2^n points.

Algorithm: **If $n = 1$ then return $T = [h(0), h(1)]$**

Recursively compute the 2^{n-1}-length table T_1 for the values of h_1,
and the 2^{n-1}-length table T_2 for the values of h_2

Return the table $T = (T_2)(T_1 + T_2)$ of 2^n entries

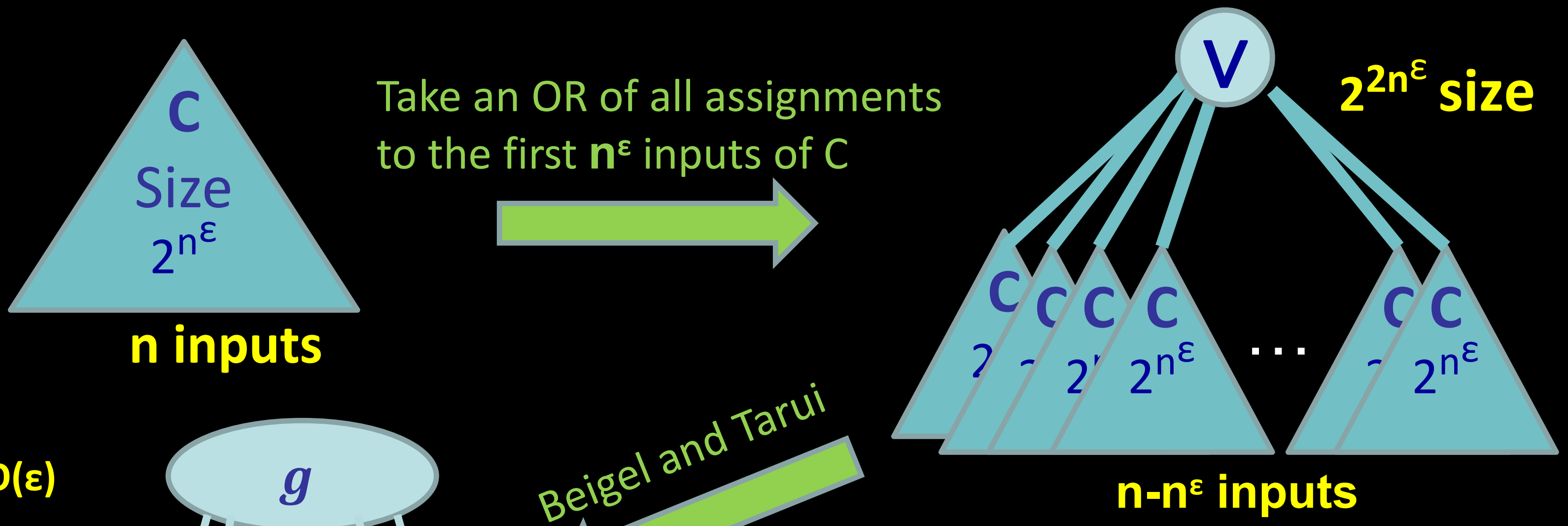
Running time has the recurrence $\boldsymbol{R(2^n) \le 2 \cdot R(2^{n-1}) + 2^n\, poly(n)}$

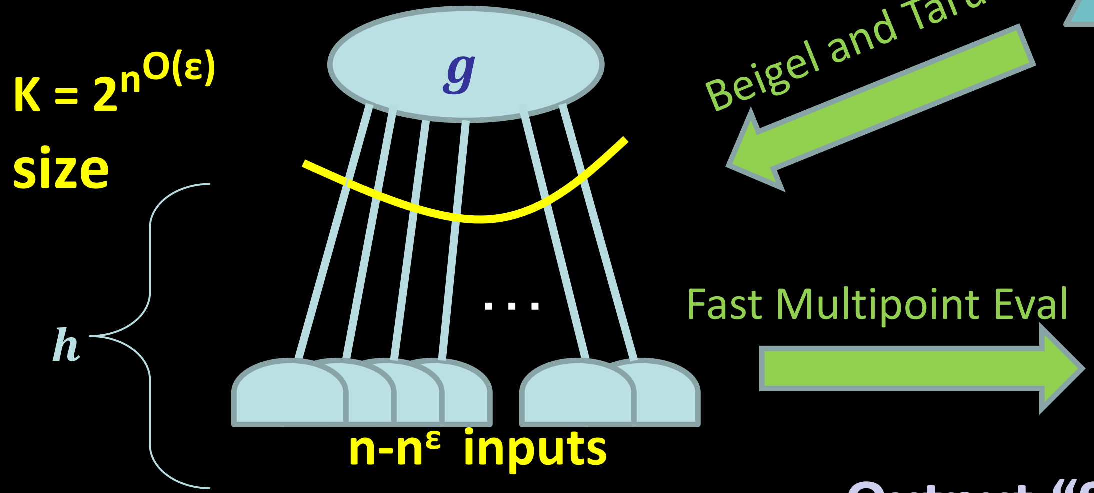
Corollary: We can evaluate g of h on all $a \in \{0, 1\}^n$, in only $2^n\, poly(n)$ time

ACC Satisfiability Algorithm

Theorem: For every d, m, there is an $\varepsilon > 0$ such that ACC-SAT on circuits with n inputs, 2^{n^ε} size, depth d, and MODm gates is solvable in 2^{n-n^ε} time

Proof:





For small ε > 0, evaluate h on all 2^{n-n^ε} assignments in 2^{n-n^ε} poly(n) time

Output "SAT" ⇔ $\exists a \in \{0,1\}^{n-n^\varepsilon}$ s.t. $g(h(a)) = 1$

The LB Connection: If *ACC*-SAT on circuits with n inputs and 2^{n^ε} size is in $O(2^n/n^{10})$ time, then **E^{NP} doesn't have 2^{n^ε} size *ACC*-circuits.**

Given circuit $\boldsymbol{C} : \{\mathbf{0},\mathbf{1}\}^{\boldsymbol{n}} \rightarrow \{\mathbf{0},\mathbf{1}\}$, let $\boldsymbol{tt(C)}$ be its truth table: the output of $\boldsymbol{C}$ on all 2^n assignments, in lexicographical order

Succinct 3SAT: ***Given a circuit C, does $tt(C)$ encode a satisfiable 3CNF?***

Key Idea: Succinct 3SAT is NEXP-complete, in a very strong way…

Lemma 1 Succinct 3SAT for AC0 circuits of n inputs and n^{10} size is solvable in nondeterministic $2^n\, poly(n)$ time but ***not*** in nondeterministic $\frac{2^n}{n^5}$ time.

Upper bound: Evaluate the AC0 circuit on all 2^n inputs, get a 2^n-length 3CNF instance, guess and check a SAT assignment, in $2^n\, poly(n)$ time

Lower bound: **[JMV'13]** Every $L \in NTIME[2^n]$ can be reduced in poly-time to a Succinct 3SAT instance which is AC0, $m = n + 4\log(n)$ inputs, n^{10} size

So, if Succinct3SAT is in $2^m/m^5$ time, then L can be decided in time $o(2^n)$

Contradicts the nondeterministic time hierarchy theorem!

The LB Connection: If **E^{NP} has 2^{n^ε} size *ACC*-circuits** and *ACC*-SAT on circuits with n inputs and 2^{n^ε} size is in $O(2^n/n^{10})$ time, then *contradiction*

Succinct 3SAT: *Given a circuit C, does $tt(C)$ encode a satisfiable 3CNF?*

Lemma 1 Succinct 3SAT for ACC circuits of n inputs and n^{10} size is solvable in nondeterministic $2^n\, poly(n)$ time but ***not*** in nondeterministic $\frac{2^n}{n^5}$ time.

Goal: Use ACC circuits for E^{NP} & the ACC-SAT algorithm, to solve Succinct 3SAT faster.

Say that **Succinct 3SAT has "succinct" SAT assignments** if

for **every $\boldsymbol{C}$** (of n inputs and n^{10} size) such that $\boldsymbol{tt(C)}$ encodes a satisfiable 3CNF $\boldsymbol{F}$,

there is an ACC circuit $\boldsymbol{D}$ of $\boldsymbol{2^{n^{10\varepsilon}}}$ **size** such that

$\boldsymbol{tt(D)}$ encodes a variable assignment $\boldsymbol{A}$ that satisfies $\boldsymbol{F}$.

(Imagine F has variables $x_1, \ldots, x_{2^n}$. Then $D(i)$ outputs a 0-1 assignment to variable x_i in *F)*

If a succinct SAT assignment exists, we only have to guess a witness of length $\boldsymbol{2^{n^{10\varepsilon}}}$

Lemma 2 If **E^{NP}** has 2^{n^ε} size ACC circuits then **Succinct 3SAT has "succinct" SAT assignments**

The LB Connection: If **E^{NP} has $2^{n^{\varepsilon}}$ size *ACC*-circuits** and
ACC-SAT on circuits with n inputs and $2^{n^{\varepsilon}}$ size is in $O(2^n/n^{10})$ time, then *contradiction*

Succinct 3SAT: ***Given a circuit C, does $tt(C)$ encode a satisfiable 3CNF?***

Lemma 2 If **E^{NP}** has $2^{n^{\varepsilon}}$ size ACC circuits then
Succinct 3SAT has "succinct" SAT assignments

Proof The following is an **E^{NP}** procedure:

On input (C, i), where $i \in \{1, \ldots, 2^n\}$, C has n inputs & n^{10} size
Compute $F = tt(C)$, think of F as a 3CNF formula.
Use a SAT oracle and search-to-decision for SAT, to find the lexicographically first SAT assignment to F.
Output the i-th bit of this assignment.

E^{NP} has $2^{n^{\varepsilon}}$ size ACC circuits $\Rightarrow$ there is a $2^{|C|^{\varepsilon}} \leq 2^{n^{10\varepsilon}}$ size ACC circuit $\boldsymbol{D(C, i)}$
which outputs the $\boldsymbol{i}$-th bit of a satisfying assignment to $\boldsymbol{F = tt(C)}$.

Now for any circuit **C' of n^{10}** size, define the circuit **E(i) := D(C', i)**
Then **E** has $2^{n^{10\varepsilon}}$ size, and the assignment **tt(E)** satisfies **tt(C')**

The LB Connection: If E^{NP} has 2^{n^ε} size *ACC*-circuits and *ACC*-SAT on circuits with n inputs and 2^{n^ε} size is in $O(2^n/n^{10})$ time, then *contradiction*

Succinct 3SAT: ***Given a circuit C, does $tt(C)$ encode a satisfiable 3CNF?***

Goal: **Use *ACC circuits for E^{NP} & the ACC-SAT algorithm* to put Succinct3SAT in $NTIME[2^n/n^5]$ (contradiction!)**

Outline of Succinct3SAT algorithm:

Given a Succinct3SAT instance C (an AC0 circuit of n^{10} size, n inputs)

1. **Guess** a $2^{n^{10\varepsilon}}$ size ACC circuit Y encoding a SAT assignment for the exponentially-long 3CNF $F := tt(C)$

Lemma 2 If E^{NP} has 2^{n^ε} size ACC circuits then **Succinct 3SAT has "succinct" SAT assignments**

2. **Check** that $tt(Y)$ satisfies F in $O(2^n/n^5)$ time, using the $O(2^n/n^{10})$-time ACC-SAT algorithm.

The entire process will take $O(2^n/n^5)$ nondeterministic time.

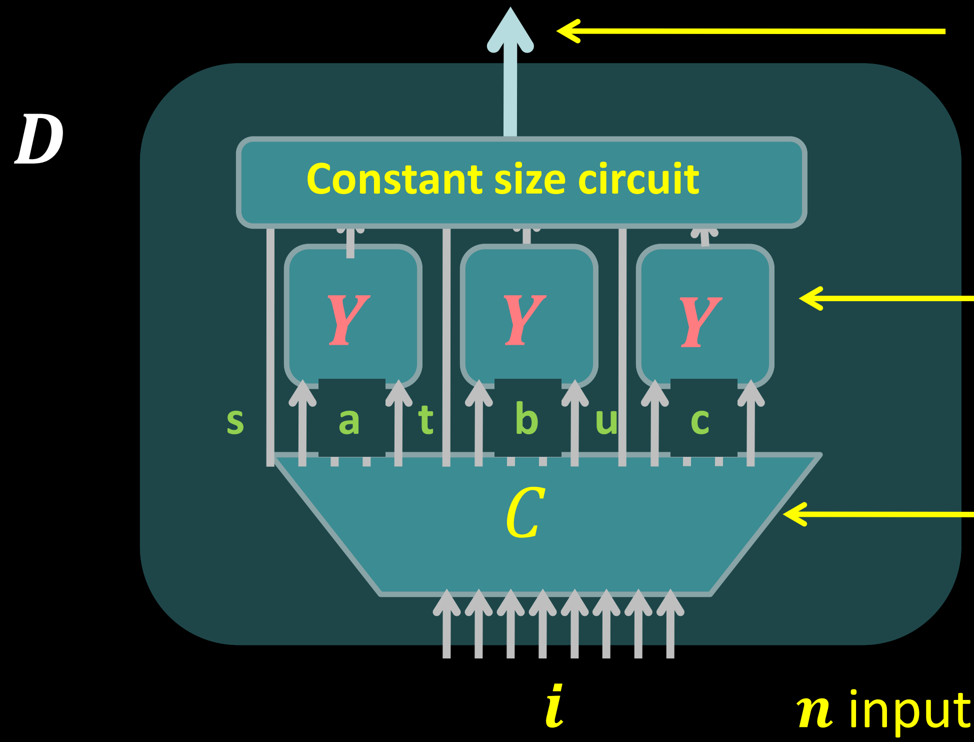
Faster Algorithm for Succinct3SAT

Given Succinct3SAT instance C **(an n-input, n^{10}-size AC0 circuit)**

Guess: ACC circuit Y of $2^{n^{10\varepsilon}}$ size

[Want: $Y(i)$ outputs the i-th bit of a satisfying assignment for $F \coloneqq tt(C)$]

To check Y, construct the following circuit D of $O(2^{n^{10\varepsilon}})$ size:



Output 1 $\Leftrightarrow$ variable assignment encoded by Y ***does not*** satisfy the i-th clause of F

$$\neg\big((s \oplus Y(a)) \vee (t \oplus Y(b)) \vee (u \oplus Y(c))\big)$$

Outputs claimed assignments to the variables x_a, x_b, x_c of F

Outputs the ith clause of 3CNF $F \coloneqq tt(C)$: $(\pm x_a \vee \pm x_b \vee \pm x_c)$

n input bits

D is UNSAT $\Leftrightarrow$ for all i, $D(i) = 0$ $\Leftrightarrow$ for all i, $tt(Y)$ satisfies i-th clause of F

$\Leftrightarrow$ $tt(Y)$ is a satisfying assignment to F. **Using ACC-SAT algorithm, takes $o(2^n)$ time to check!**

Theorem If ACC SAT with n inputs, 2^{n^ε} size is in $O(2^n/n^{10})$ time, then Quasi-NP doesn't have $n^{O(1)}$ size ACC circuits.

Proceed just as before, but use the following lemma:

Lemma [MW'18] "Easy Witness Lemma for Quasi-NP"

If **Quasi-NP ⊆ P/poly** then for all $\varepsilon \in (0,1)$,
Succinct 3SAT on n-input circuits of size 2^{n^ε} has $2^{O(n^\varepsilon)}$ -size circuits encoding SAT assignments.

Idea: The problem ***Succinct 3SAT on n-input circuits of size 2^{n^ε}*** is actually a Quasi-NP (complete) problem:

- input is of length $N \sim 2^{n^\varepsilon}$
- takes about $O(2^n) \leq O(2^{(\log N)^\wedge(1/\varepsilon)})$ time to guess-and-check a variable assignment, and verify it is satisfying

[MW'18] shows Quasi-NP in P/poly implies every Quasi-NP verifier has witnesses encoded by poly-size circuits!

Theorem If ACC SAT with n inputs, $2^{n^{\varepsilon}}$ size is in $O(2^n/n^{10})$ time, then Quasi-NP doesn't have $n^{O(1)}$ size ACC circuits.

Proceed just as before, but use the following lemma:

Lemma [MW'18] "Easy Witness Lemma for Quasi-NP"

If **Quasi-NP ⊆ P/poly** then for all $\varepsilon \in (0,1)$,
Succinct 3SAT on n-input circuits of size $2^{n^{\varepsilon}}$ has $2^{O(n^{\varepsilon})}$ -size circuits encoding SAT assignments.

Lemma If **P ⊆ ACC** then all poly-size *unrestricted* circuit families have equivalent poly-size ACC circuit families ***(exercise!)***

Therefore we can assume WLOG that the $2^{O(n^{\varepsilon})}$ *size circuits encoding SAT assignments are in fact ACC circuits, and apply the previous argument (with minor modifications)*

Weak Derandomization Suffices for Lower Bounds!

Gap-$\mathcal{C}$-SAT: Given $K(x_1,\ldots,x_n)$ from $\mathcal{C}$, and the **promise** that either (a) $K \equiv 0$, or (b) $Pr_x[K(x) = 1] \geq 1/2$, **decide** which is true.

Theorem If Gap-$\mathcal{C}$-SAT on circuits with n inputs and 2^{n^ε} size is in $O(2^n/n^{10})$ time, then **E^{NP} doesn't have 2^{n^ε} size $\mathcal{C}$-circuits.**

Proof Idea: Same as before, but **replace** the reduction to Succinct 3SAT with a ***succinct PCP reduction to "Succinct MAX CSP"!***

Lemma 3 [BGHSV'05,…,BV'14] For all $L \in NTIME(2^n)$, there is a reduction S_L from L to **MAX CSP** such that:

- $x \in L \Rightarrow$ **All constraints of $S_L(x)$ are satisfiable by some assignment**
- $x \notin L \Rightarrow$ **No assignment satisfies more than ½ of the constraints of $S_L(x)$**
- $|S_L(x)| = 2^n$ **poly(n), each constraint of $S_L(x)$ is on $O(1)$ variables**
- **There is a poly-size AC0 circuit that given i, outputs the i-th constraint of $S_L(x)$.**

(Exercise: Verify that this means we can replace ACC-SAT with Gap-ACC-SAT!)

Open Problems

- **Replace Quasi-NP with smaller complexity classes**
  Could we prove NP is not in polynomial-size ACC?

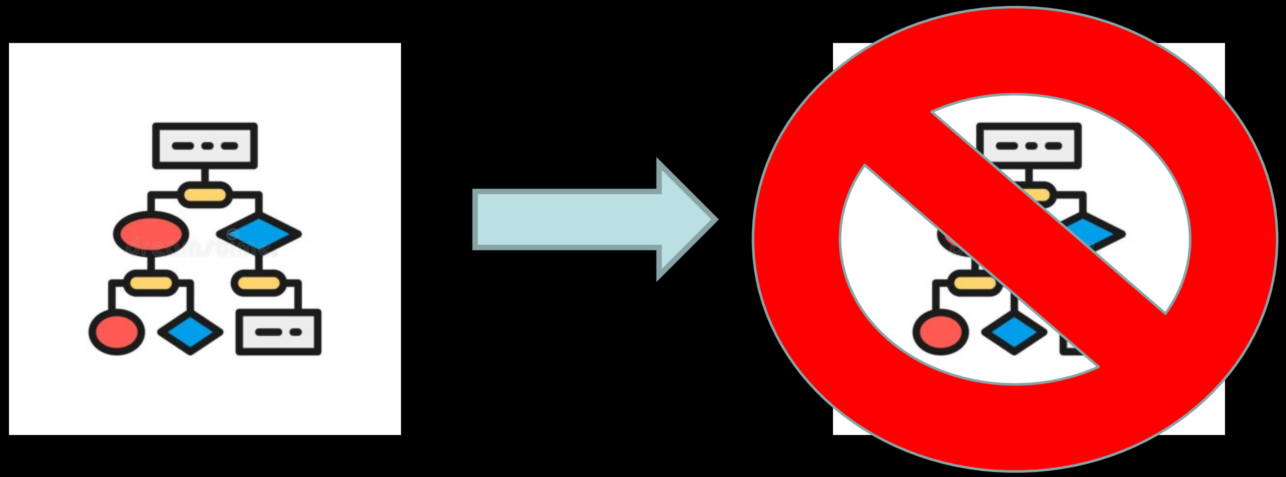
- **Replace ACC with stronger circuits**
  Current strongest: "MAJ of SUM of ACC of THR" [Chen-Lu-Lyu-Oliveira'21]
  Obtained from a #SAT algorithm for ACC of THR
  Design SAT/#SAT algorithms for stronger circuits!

Example Open Problem: ***Can Boolean formulas of size*** s ***be evaluated on all n-variable assignments in*** $2^{s^{o(1)}} + 2^n \cdot poly(n)$ ***time?***

- **Simplify the proofs!**

- **More connections between algorithms and lower bounds *(next lecture!)***

How to Prove Lower Bounds With Algorithms



Lecture 3: The Mysteries of the Missing String

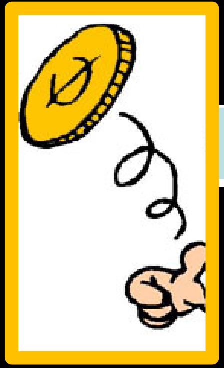
(based on work with Nikhil Vyas, ITCS 2023)

Which Functions Have High Circuit Complexity?

Nearly all of them… "Most" functions require *huge* circuits!

Theorem [Shannon '49, Lupanov '58]

With high probability,
a randomly chosen function $\boldsymbol{f : \{0,1\}^n \rightarrow \{0,1\}}$
does not have circuits of size less than $\sim \boldsymbol{2^n/n}$
(and: every $\boldsymbol{f}$ has a circuit of size about $\sim \boldsymbol{2^n/n}$)



Which "natural" functions exhibit this exponential behavior?

Which Functions Have High Circuit Complexity?

What is the "*smallest*" complexity class containing a function of ***maximum*** $(2^n/n)$ circuit complexity?

Smallest known class: $E^{\Sigma_2 P} = TIME\left[2^{O(n)}\right]^{\Sigma_2 P}$ [1970s]

Theorem: There is a function $f \in E^{\Sigma_2 P}$ that requires ***maximum*** circuit complexity.

Recall: $f \in \Sigma_2\, P \Leftrightarrow$ *there is a polynomial* $p(n)$ *and a* $p(n)$*-time verifier* $V(x, y, z)$ *such that for all* x, $f(x) = 1 \Leftrightarrow$ *(*$\exists$ *p-length* y*)(*$\forall$ *p-length* z*)*$[V(x, y, z)$ *accepts*$]$

A Canonical Problem in $\Sigma_2 P$: **Circuit Minimization**

Input: Boolean circuit C

Decide: Is there a circuit smaller than C that computes the same function?

"$\Sigma_2\, P$ algorithm" for **Circuit Minimization:**

Existentially guess circuit C' of size less than C, with an equal number of inputs

Universally check over all inputs x that $C(x) = C'(x)$

A Function With Max Circuit Complexity

Theorem: There is a function $f \in E^{\Sigma_2 P}$ that requires ***maximum*** circuit complexity.

Proof Sketch: On $x \in \{0,1\}^\star$ of length n,

// first, find the max circuit complexity $s^\star$ of a function on n inputs

$s \coloneqq 2^n$

while ($\forall f: \{0,1\}^n \rightarrow \{0,1\}$, f has a circuit of size at most s)

// can formulate this as an $2^{O(n)}$-length query to a $\Sigma_2 P$ oracle

$s \coloneqq s - 1$

// at this point, max circuit complexity $s^\star = s + 1$

Use a "search-to-decision" reduction on the $\Sigma_2 P$ oracle to find the

lexicographically first truth table $T \in \{0,1\}^{2^n}$ that has circuits of size s

Output x-th bit of T

A Function With Max Circuit Complexity

Theorem: There is a function $f \in E^{\Sigma_2 P}$ that requires ***maximum*** circuit complexity.

Proof Sketch: On $x \in \{0,1\}^\star$ of length n,

// first, find the max circuit complexity $s^\star$ of a function on n inputs

$s := 2^n$

while ($\forall\ f: \{0,1\}^n \to \{0,1\}$, f has a circuit of size at most s)

// can formulate this as an $2^{O(n)}$-length query to a $\Sigma_2 P$ oracle

$s := s - 1$

// at this point, max circuit complexity $s^\star = s + 1$

Problem: "Complexity"

Input: (f, s) where $f \in \{0,1\}^{2^n}$, $k \in \{1, \dots, s\}$

Decide: Is there a 2^n-bit $g < f$ that has circuit complexity at least s?

Complexity $\in \Sigma_2 P$ [guess $g < f$, check g is different from all circuits C of size $< s$]

We can construct the lex. first truth table that needs circuits of size at least s, using 2^n queries to Complexity of length $O(2^n)$

[start with $f := 1^{2^n}$, try to set bits of f to 0, and check if $(f, s) \in$ Complexity]

Which Functions Have High Circuit Complexity?

What is the *smallest* complexity class containing a function of ***maximum* $(2^n/n)$** circuit complexity?

Smallest known: $E^{\Sigma_2 P} = TIME\left[2^{O(n)}\right]^{\Sigma_2 P}$ [1970s]

Recall: $f \in \Sigma_2\, P \Leftrightarrow$ *there is a polynomial* $p(n)$ *and a* $p(n)$*-time verifier* $V(x,y,z)$ *such that for all* x, $f(x) = 1 \Leftrightarrow$ *(∃ p-length y)(∀ p-length* z*)*$[V(x,y,z)$ *accepts*$]$

Theorem: There is a function $f \in E^{\Sigma_2 P}$ that requires ***maximum*** circuit complexity.

This lower bound relativizes! Let $A: \{0,1\}^\star \to \{0,1\}$ **be an arbitrary "oracle"**

$f \in \Sigma_2\, P^A \Leftrightarrow$ *there is a polynomial* $p(n)$ *and a* $p(n)$*-time verifier* $V^A(x,y,z)$ *such that for all* x, $f(x) = 1 \Leftrightarrow$ *(∃ p-length y)(∀ p-length* z*)*$[V^A(x,y,z)$ *accepts*$]$

Theorem: There's $f \in E^{\Sigma_2 P^A}$ that requires ***maximum*** A-oracle circuit complexity.

[A-oracle circuit: can have unbounded fan-in gates that compute A]

Which Functions Have High Circuit Complexity?

What is the *smallest* complexity class containing a function of
maximum $(2^n/n)$ circuit complexity?

Smallest known: $E^{\Sigma_2 P} = TIME\left[2^{O(n)}\right]^{\Sigma_2 P}$ [1970s]

Recall: $f \in \Sigma_2 P \Leftrightarrow$ *there is a polynomial* $p(n)$ *and a* $p(n)$*-time verifier* $V(x, y, z)$ *such that for all* x, $f(x) = 1 \Leftrightarrow$ *(∃ p-length y)(∀ p-length z)*$[V(x, y, z)$ *accepts*$]$

Theorem: There is a function $f \in E^{\Sigma_2 P}$ that requires ***maximum*** circuit complexity.

Corollary: *If* $P = NP$ *then there is an* $f \in E$ *that requires **maximum** circuit complexity.*

Proof: $P = NP \Rightarrow E^{\Sigma_2 P} = E^{NP^{NP}} = E^{NP^P} = E^{NP} = E^P = E$

Thus, one could prove $P \neq NP$*, if one could show that every function in* E *has circuit complexity at most* $H(n) - 1$

[where $H(n)$*=maximum circuit complexity for* n*-bit functions]*

Which Functions Have High Circuit Complexity?

What is the *smallest* complexity class containing a function of ***maximum*** $(2^n/n)$ circuit complexity?

Smallest known: $E^{\Sigma_2 P} = TIME\left[2^{O(n)}\right]^{\Sigma_2 P}$ [1970s]

Theorem: There is a function $f \in E^{\Sigma_2 P}$ that requires ***maximum*** circuit complexity.

If we could show E^{NP} needs $2^{\Omega(n)}$-size circuits (for example),
then by derandomization [NW,IW], we would have $BPP \subseteq P^{NP}$ [major open problem!]
There doesn't seem to be ***any*** real barrier to improving the $E^{\Sigma_2 P}$ to $\Sigma_2 E$
[Recall: $\Sigma_2 E = NTIME\left[2^{O(n)}\right]^{\mathrm{NP}}$, the exponential-time analogue of $\Sigma_2 P$]
(one consequence would be $BPP \subseteq \Sigma_2 P$, but we already know this is true!)

Which Functions Have High Circuit Complexity?

What is the *smallest* complexity class containing a function of
maximum $\boldsymbol{(2^n/n)}$ circuit complexity?

Smallest known: $E^{\Sigma_2 P} = TIME\left[2^{O(n)}\right]^{\Sigma_2 P}$ [1970s]

Does $\Sigma_2 E$ contain a function requiring $2^{\Omega(n)}$-size circuits?
[Recall: $\Sigma_2 E$ = $NEXP^{NP}$, the exponential-time analogue of $\Sigma_2 P$]

Kannan [1981] $\Sigma_2 E$ requires $n^{polylog\, n}$ -size circuits (in fact "sub-half-exponential")

But we don't even know if there is an oracle A under which $\Sigma_2 E$ has $2^{o(n)}$-size circuits!

That is, the following is open:

Is there an oracle A such that $\Sigma_2 E^A$ has an A-oracle circuit of $2^{o(n)}$ size?

(note: there ***is*** an oracle A such that $\Sigma_2 E^A$ requires $2^{\Omega(n)}$-size A-oracle circuits)

So it remains possible there is a ***relativizing proof*** that $\Sigma_2 E$ requires $2^{\Omega(n)}$-size circuits, i.e., one that works by standard "black-box" techniques!

We will give a completely new way of thinking about this question!

The Missing String Problem

Missing String: Given a list of M strings of length N ($M < 2^N$), find a string not on the list

If the list is truth tables of "easy" functions, we are asking you to find a "hard" function!

Theorem [Folklore? K'81] There is an $\tilde{O}(M)$-time algorithm for Missing String.

Proof: *"Halving Algorithm".* Our missing string will be $x = x_1 x_2 \cdots x_N$

Probe the 1st bit of each string.

Set x_1 = "minority" bit of those M bits — M probes

Probe the 2nd bits of all $t_2 \leq M/2$ strings with first bit = x_1

Set x_2 = "minority" bit of those t_2 bits — $\leq M/2$ probes

Probe the 3rd bits of all $t_3 \leq M/4$ strings with first two bits = $x_1 x_2$

Set x_3 = "minority" bit of those t_3 bits — $\leq M/4$ probes

After $(\log_2 m) + 1$ rounds, no strings are left. Fill any remaining x_i's with zeroes.

Total number of probes is $O(M)$.

MISSING STRING and High-Complexity Functions

There is a strong ***equivalence*** between questions like:

Is there an oracle A such that $\Sigma_2 E^A$ has an A-oracle circuit of $2^{o(n)}$ size?

And the *circuit complexity* of **MISSING STRING!**

"Efficient" circuits for Missing String correspond to "efficient" constructions of hard functions!

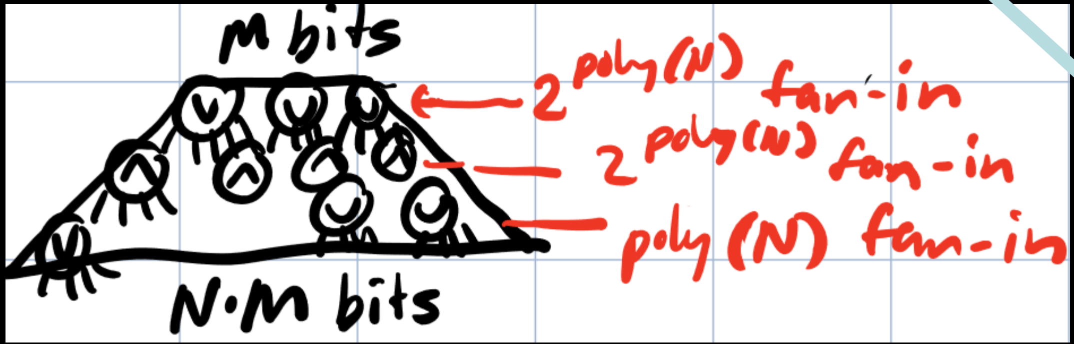
Example Theorem:

For every oracle A, $\Sigma_2 E^A$ requires $2^{\varepsilon n}$-size A-oracle circuit complexity (a.e.)

$$\Leftrightarrow$$

There are *uniform* ***depth-3*** AC0 circuits for **MISSING STRING** of $2^{poly(N)}$ size and $poly(N)$ bottom fan-in, on all lists of length $M \approx 2^{N^\varepsilon}$

Depth 3 is crucial here!



Note: since $M \approx 2^{N^\varepsilon}$ we're asking for a circuit that has size *quasi-polynomial* in its input length!

MISSING STRING and High-Complexity Functions

There is a strong ***equivalence*** between questions like:

Is there an oracle A such that $\Sigma_2 E^A$ has an A-oracle circuit of $2^{o(n)}$ size?

And the *circuit complexity* of **MISSING STRING!**

"Efficient" circuits for Missing String correspond to "efficient" constructions of hard functions!

Example Theorem:

For every oracle A, NE^A requires $2^{\varepsilon n}$-size A-oracle circuit complexity (a.e.)

Known to be FALSE [Wilson'84]

$\Leftrightarrow$

There are *uniform* ***depth-2*** AC0 circuits for **MISSING STRING** of $2^{poly(N)}$ size and $poly(N)$ bottom fan-in, on all lists of length $M \approx 2^{N^{\varepsilon}}$

Therefore, no depth-2 circuits!

Depth 3 is crucial here!

MISSING STRING and High-Complexity Functions

There is a strong ***equivalence*** between questions like:

Is there an oracle A such that $\Sigma_2 E^A$ has an A-oracle circuit of $2^{o(n)}$ size?

And the *circuit complexity* of **MISSING STRING!**

"Efficient" circuits for Missing String correspond to "efficient" constructions of hard functions!

Example Theorem:

For every oracle A, $\Sigma_3 E^A$ requires $2^{\varepsilon n}$-size A-oracle circuit complexity (a.e.)

Known to be TRUE [Kannan'81]

$\Longleftrightarrow$

There are *uniform* ***depth-4*** AC0 circuits for **MISSING STRING** of $2^{poly(N)}$ size and $poly(N)$ bottom fan-in, on all lists of length $M \approx 2^{N^\varepsilon}$

Therefore, depth-4 circuits exist!

Depth 3 is crucial here!

Missing String and High-Complexity Functions

Theorem [Easier Case]:

Also works for AND-OR-AND circuits

There are *uniform* ***OR-AND-OR*** circuits for **Missing String** of $2^{poly(N)}$ size and $poly(N)$ bottom fan-in, on all lists of length $M = 2^{O(N^{\varepsilon} \log N)}$

$\Rightarrow \Sigma_2 E$ requires $2^{\varepsilon n}$-size circuit complexity

This is the "algorithms imply lower bounds" direction!

Proof Sketch:

Our $\Sigma_2 E$ function will evaluate the depth-3 circuit for **Missing String** with its input fixed to be:
the list of all 2^n-bit truth-tables of functions $f: \{0,1\}^n \to \{0,1\}$ with circuits of size $2^{\varepsilon n}$

$N = 2^n$, $M = 2^{O(2^{\varepsilon n} n)}$ [there are $2^{O(2^{\varepsilon n} n)}$ circuits of size $2^{\varepsilon n}$]

On input $x \in \{0,1\}^n$, our $\Sigma_2 E$ function evaluates the x-th output of the Missing String circuit:

Existentially guess an input wire to the output OR gate which is true, coming from an AND gate g
Universally try all input wires to the AND gate g, coming from an OR gate h
Evaluate all $poly(N)$ literals with wires into h, **accept** iff at least one literal is true.

Both the existential and universal guesses take $poly(N)$ bits to write down (fan-in is $2^{poly(N)}$)

Assuming we can compute local information about the gates of the circuit in $poly(N)$ time, the above process can be implemented in $poly(N) = 2^{O(n)}$ time on a Σ_2 machine.