ACC Circuit Lower Bounds

Rishabh Vaid
IIT Kanpur

NEXP $\not\subset$ ACC

- What does it mean?
- Why do we care?
- NEXP is simply the ‘exponential’ version of NP
- The class ACC consists of circuit families with constant depth over unbounded fan-in AND, OR, NOT and MOD_m gates, where m is an arbitrary constant.
- What is a circuit family?

Non-Uniform Computation

- Allow a different logical circuit A_n to run on inputs of length n
- P/Poly := Problems solvable with a circuit family $\{A_n\}$, where $|A_n| < n^k$
- Conjectured: NP $\not\subset$ P/Poly
- Open: NEXP $\not\subset$ P/Poly
- Non-Uniformity can be very powerful – The halting problem can be solved with O(1) bits of advice!

Circuit Lower Bounds So Far

- Furst-Saxe-Sipser (‘81) : PARITY $\not\subset$ AC0
- Razborov-Smolensky (‘87): $MOD_p \not\subset$ AC0 with MOD_q, if p and q are unequal primes
- Barrington (‘89): Conjectured MAJORITY $\not\subset$ ACC
- Till recently it was open whether even NEXP $\not\subset$ ACC – Williams (‘10) finally resolved this positively

ACC Lower Bounds

Theorem: There is a language L in NEXP such that L does not have polynomial size ACC circuits

Proof Outline:

- Prove that if NEXP were in ACC, we would have faster algorithms for NEXP
- Show a contradiction with the Non-Deterministic Time Hierarchy Theorem
- What should L be?

NEXP Completeness - SUCCINCT 3SAT

- Given: A Boolean circuit C with n inputs and poly(n) size
- Task: Determine whether T(C) is a Yes instance of 3SAT
- SUCCINCT 3SAT is *very* NEXP complete – There are extremely efficient reductions from any language L in NEXP to SUCCINCT 3SAT

Efficient Cook Levin for NEXP

Theorem: For any language L in NTIME[2^n], given a string x, there is a poly-time reduction R, such that :

- x ∊ L iff R(x) ∊ SUCCINCT 3SAT
- R(x) is poly(n) size
- R(x) has at most $|x| + 4\log|x|$ inputs

Corollary : SUCCINCT 3SAT cannot be solved in $O(2^{n-\omega(\log n)})$ non-deterministic time on circuits with n inputs and poly(n) gates

Spinning Circuits into Algorithms

Theorem: If SUCCINCT 3SAT is solvable with poly-size ACC circuits, there is an $\epsilon > 0$ such that SUCCINT 3SAT is solvable in $O(2^{n - n^{\epsilon}})$ non-deterministic time, on all circuits with n inputs and poly(n) size

This is essentially because:

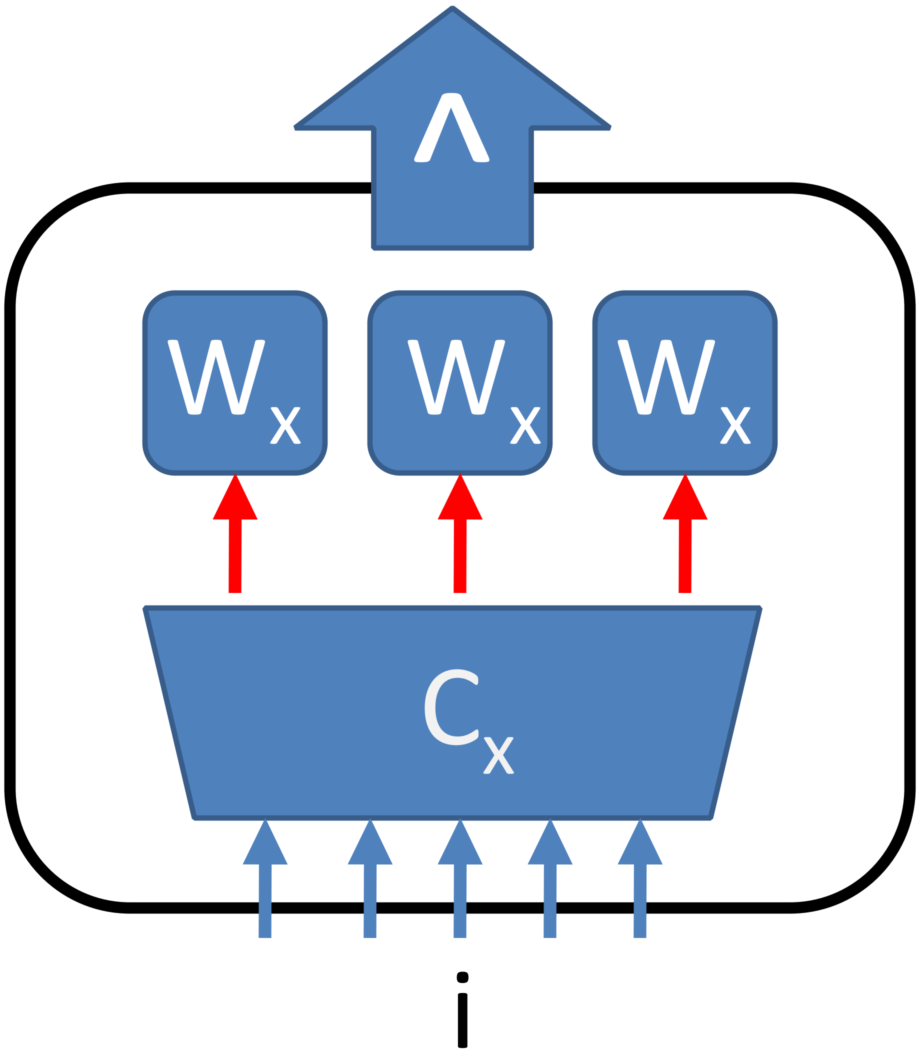
- If NEXP C ACC, all sorts of expensive computations can be simulated with ACC circuits (which we guess)
- ACC circuits are 'weak' i.e. ACC SAT can be solved in faster than brute force time

Succinct Satisfying Assignments

- Impagliazzo, Kabanets, Wigderson ('02): Suppose NEXP C P/Poly. Then for every C_x ϵ SUCCINCT 3SAT, there is a circuit W_x of poly($|C_x|$) size and O($|C_x|$) inputs such that T(W_x) is a satisfying assignment for F_x
- We may also assume W_x is an ACC circuit
- This is because there is an ACC family that solves Circuit Value Problem (Given C, x, does C(x) = 1)
- In place of C, feed the circuit an encoding of W_x. For any input x, the ACC circuit now outputs the same value as W_x

Checking if $T(W_x)$ Satisfies F_x

- The given circuit D_x outputs 1 on input i, if the ith clause is satisfied
- Hence, if $\neg D_x$ is unsatisfiable, $T(W_x)$ satisfies F_x
- However, C_x is unrestricted
- We must somehow replace C_x with an ACC circuit
- Guess an ACC circuit A_x equivalent to C_x

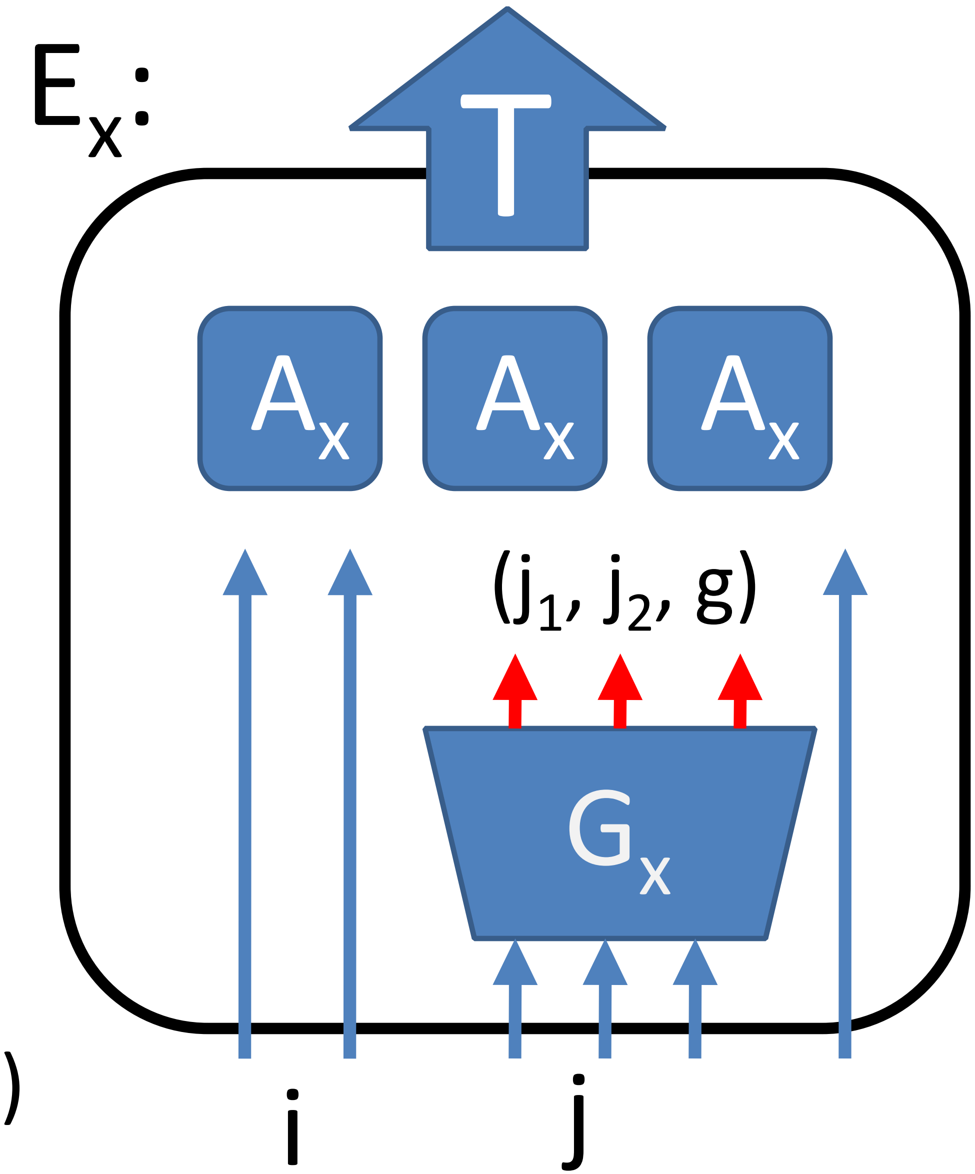


Equivalence of Circuits via ACC SAT

- Using the obvious way to check if C_x is equivalent to the guessed circuit A_x, we are lead into an infinite regress
- Key insight is that the structure of the (unrestricted) circuit C_x can be encoded using a small ACC circuit G_x, and then each step of the computation of A_x can be verified to be consistent with the corresponding step in C_x

Equivalence of Circuits via ACC SAT

- Extend the definition of C_x to output the value on the j^{th} wire
- G_x is a circuit which encodes the structure of C_x, by giving the controlling wires j_1 & j_2 and the gate g for any wire j (G_x is hard coded)
- T checks consistency



A Faster ACC SAT Algorithm

Ingredients:

- A representation of ACC (Beigel-Tarui '94): Every ACC function f can be expressed in the form $f(x) = g(h(x))$, where h is a sparse multilinear polynomial, and g is a lookup table
- Let K be the number of monomials in $h(x)$. K is quasipolynomial in circuit size
- Dynamic Programming Algorithm for computing $h(x)$ for all $x \in \{0,1\}^n$ in $2^n \mathrm{poly}(n)$ time

A Faster ACC SAT Algorithm

Theorem: For all d, m there is an $\epsilon > 0$ such that ACC[m] SAT with depth d, n inputs, $O(2^{n^\epsilon})$ size can be solved in $2^{n - n^\epsilon}$ time

- Fix a subset of k input variables, and evaluate the circuit C on all 2^k assignments of these variables. Let the OR of the induced circuits be denoted by C'
- C' has n-k inputs
- Size(C') = 2^k Size(C)
- C' is satisfiable iff C is satisfiable
- Decompose C' = $g' \circ h'$ where h' has n-k variables and the number of monomials is quasipolynomial in 2^kSize(C)
- Evaluate C' in $O(2^{n-k}\text{poly}(n) + K)$ time