Lower Bounds **(CS294-152, Fall'18)**

A Nice Polynomial Representation for ACC^0

Notes for 10/29/18

Scribe: Haaris Khan

Announcements: Project Proposals are due **TODAY**!

Due to timing constraints, if you have yet to scribe for the course, please fill out the "When2Meet" and specify what times you are available for an extra lecture. An announcement will be made regarding when those lecture dates/times have been set.

Our remaining schedule looks as follows:

1. Nov 5: Lecture
2. Nov 7-9 or 28 - 29: Lecture
3. Nov 12: No lecture (Veteran's Day)
4. Nov 19: Two Guest Lectures (Amir Y and Kasper Green L.)
5. Nov 26: Project Presentations

Email Ryan for any questions regarding the project.

1 Preliminaries

The best general lower bound that's known for ACC^0 is much weaker than what is known for $\mathsf{AC}^0[p]$:

Theorem 1.1. *There are functions in* $\mathsf{NTIME}(n^{polylog(n)})$ *which are not in poly-size* ACC^0.

The lower bound above is useful, however, in exploiting a polynomial representation for ACC^0 (albeit not as "nice" as the probabilistic polynomial representation seen previously for $\mathsf{AC}^0[p]$).

Here, we will explore the exact representation, and in the future we will see how to use this for deriving lower bounds.

2 Polynomial Representation of ACC^0: Construction

Definition 1. *Function $f : \{0,1\}^n \to \{0,1\}$ is symmetric if $\exists\ g : \{0,1,...,n\} \to \{0,1\}$ such that $\forall x, f(x) = g(\sum_i x_i)$*

We further define SYM as the class of such symmetric boolean functions

Theorem 2.1 (Beigel-Tarui 91 (building on Yao))**.** *Every symmetric function f has a corresponding symmetric function f' s.t. f of s $\mathsf{AC}^0[m]$ circuits of size s and depth d can be represented using f', with $s^{polylog(s)}$ AND gates, where the polylog factor is dependent on d and m*

Moreover, the transformation can be computed deterministically in $s^{polylog(s)}$ time, provided a starting "f of ACC" circuit as input.

We will give a rough idea of Biegel-Tarui for $\mathsf{AC}^0[p]$ circuits, where $p = 6$. Do note, however, that this sketch can be generalized to any p, where we write out the MOD terms as MODs over general prime factors of p.

2.1 Sketch of Biegel-Tarui

Consider the class of $\mathsf{AC}^0[6]$ size-s depth-d circuits. First, note that these circuits can be represented as a SYM of $s^{polylog(s)}$ AND gates. (Note that this circuit class already poses dificulties in proving lower bounds, even when the depth is limited to 3).

Now, extend this class by considering a probabilistic construction (that can be derandomized)

1. Write OR as MOD_2 and AND of MOD_2 (DeMorgan's Laws)
2. Think of the $\mathsf{AC}^0[6]$ circuit C as being *layered* (similar to how we viewed AC^0). Layers of (MOD_3 of MOD_2 of AND), repeated $\leq d$ times, by sticking in dummy gates that don't do anything. Essentially, we have composition up until depth d. Size is still $\leq s^c$ for some $c \leq O(d)$
3. Construct 10n circuits $C_1, ..., C_{10n}$ as follows:
   3.1. Replace each AND gate (over $\{0,1\}$) with a probabilistic F_3 polynomial of degree $2c * \log(s)$. The error here is bounded by $\frac{1}{s^{2c}}$, so our whole probabilistic circuit has error $\leq \frac{s^c}{s^{2c}} = \frac{1}{s^c}$
   We can write each such AND gate (with fan-in S) as a probabilistic polynomial as a MOD_3 of $S^{2c\log(S)}$ ANDs, where each AND has a fan-in $\leq 2c * \log(S)$

3.2. Write each MOD_3 of AND of fan-in $\leq 2c * \log(S)$ as a MOD_3 of S^{2c} MOD_2s of fanin $M = 2c * \log(S)$

More particularly, the AND gate on v variables can be written as a sum (mod 3) of $\leq 2^v \mathsf{MOD}_2$s.

(We can do this because *every* function on v variables can be written in the Fourier basis, as a real-valued sum of $\leq 2^v \mathsf{MOD}_2$s on v variables. Moreover, each fourier coefficient for these MOD_2s has the form $\frac{j}{2^v}$, $j \in [-2^v, ..., 2^v]$. In F_3, we're permitted to divide by 2, so our answer is preserved over F_3).

3.3. Now each C_i is $\leq 2d$ layers of (MOD_3 of MOD_2).

4. Sample $10n$ of these polynomials for the $MAJORITY(C_1, ..., C_{10n})$. Call the result D. Through a Chernoff bound, we have

$$\Pr(MAJ(C_1, ..., C_{10n})(x)) \neq C(x)) << \frac{1}{2^n}$$

Furthering this with a Union Bound argument,

$$\Pr_{C_1,...C_{10n}}(\exists x s.t. D(X) \neq C(X)) << 1$$

Which implies there exists some gates $C_1, ..., C_{10n}$ such that the majority function computed on those gates are equivalent to the original function, i.e. D is correct on all 2^n inputs with good probability.

5. Transform our $MAJORITY \circ (\mathsf{MOD}_3 \circ \mathsf{MOD}_2)^{\mathcal{O}(d)}$ into a SYMMETRIC function of ANDs.

More generally, we will show that we can take *any* SYM of MOD_3 of MOD_2 and convert it into a SYM of MOD_2 of "small" fan-in, and vice versa.

Concretely, suppose we have a SYM of t MOD_3 of subcircuits $C_{i,1}, ..., C_{i,u}$, which output $\{0, 1\}$, and where each $C_{i,j}$ is a MOD_2 of some inputs (note: there are $t * u$ subcircuits in total).

Let the symmetric function be f, which has t inputs. Then our total circuit is equivalent to:

$$f(\mathsf{MOD}_3(\sum_{j=1}^{u} C_{1,j}), ..., \mathsf{MOD}_3(\sum_{j=1}^{u} C_{t,j}))$$

We want to convert this function into something more of the form:

$$g(AND, ..., AND)$$

Where each AND is over some subset of the inputs.

Next, we take a slight aside and introduce the magical mathematical objects that are *Modulus-Amplifying Polynomials*.

Definition 2 (Modulus-Amplifying Polynomials)**.** *Polynomial $P_L(x)$ is L-mod-amplifying if* $\forall m, x \in \mathbb{Z}, m >= 2$*,*

$$x = 0 \textit{ mod } m \implies P_L(x) = 0 \textit{ mod } m^L$$
$$x = 1 \textit{ mod } m \implies P_L(x) = 1 \textit{ mod } m^L$$

P_L "amplifies the modulus" from m to m^L

Key Property: For prime p, $\forall a \in Z$, $a^{p-1} = 0$ if $a = 0,$ and 1 otherwise, so we have: $\mathsf{MOD}_p(a) = (1 - a^{p-1})$ mod p $= P_L(1 - a^{p-1})$ mod p^L (!).

(Recall $\mathsf{MOD}_p(x) = 1$ iff x is divisible by p).

Theorem 2.2 (B-T)**.** *For every L, there exists an L-mod-amplifying polynomial P_L of degree $\leq 2L$ that can efficiently be constructed.*

For our purposes, we set $L := 1 + \log_3(t)$.

Consider:

$$(*) = \sum_{i=1}^{t} P_L(1 - (\sum_{j=1}^{u} C_{i,j})^2)$$

By the mod-amplifying property, this is equivalent to

$$(*) = \sum_{i=1}^{t} P_L(\sum_{j=1}^{u} C_{i,j})^2) \text{ mod } 3^L(\text{ because } t < 3^L)$$
$$= \sum_{i=1}^{t} \mathsf{MOD}_3(\sum_{j=1}^{u} C_{i,j}) \text{ mod } 3^L(\text{ by (!)})$$

Key Observation: Since $t < 3^L$, the sum $(*) = t'$ mod 3^L, where t' is the number of MOD_3 gates going into f which are true.

This means we can determine the output of the original circuit

$$f(\mathsf{MOD}_3(\sum_{j=1}^{u} C_{1,j}), ..., \mathsf{MOD}_3(\sum_{j=1}^{u})C_{t,j})$$

from $(*)$.

In other words, $\exists g : \mathbb{Z} \to \{0, 1\}$ such that

$$f(\mathsf{MOD}_3(\sum_{j=1}^{u} C_{1,j}), ..., \mathsf{MOD}_3(\sum_{j=1}^{u})C_{t,j}) = g(\sum_{i=1}^{t} P_L(1 - (\sum_{j=1}^{u} C_{i,j})^2)$$

If we write out the monomials of the polynomial P as a sum of ANDs of fan-in $\leq 2L$, we yield a SYM of ANDs of fan-in $2L \leq O(\log(t))$ of the subcircuits $C_{i,1}, ..., C_{i,u}$.

3 Algorithms for Analyzing ACC^0

What can we do with this SYM and AND representation of circuits? It turns out we do know something that has a provable lower bound:

Proposition 1. *The SYM of AND can be used to algorithmically **analyze** a given ACC^0 circuit, surprisingly fast.*

Furthermore, that circuit-analysis algorithm can be used to prove a lower bound.

3.1 Circuit-SAT

Circuit SAT: Given circuit C of n inputs, size s, does there $\exists\ x$ such that $C(x) = 1$?

The naive algorithm, exhaustive search, solves this in $2^n poly(s)$ time. Can this be solved faster? For ACC^0 circuits, the answer is yes!

Let $AC^0_d[m]$ be AC circuits of depth d with AND, OR, NOT, MOD_m gates.

Theorem 3.1 (W'11)**.** *For all d and m, $\exists \delta > 0$ such that the SAT problem for AC^0 circuits can be solved in 2^{n-n^δ} time.*

We prove this theorem via a classic divide-and-conquer result.

Recall a multilinear polynomial has form:

$$p(x_1, ..., x_n) = \sum_{s \subseteq [n]} c_s \prod_{i \in s} x_i$$

Lemma 3.1. *Given a multilinear polynomial p in n variables with m monomials (specified as a list of the coefficients c_S), there is a $poly(n) * (2^n + m)$ time algorithm for evaluating p on all points in $\{0,1\}^n$.*

Imagine m is huge, close to 2^n. A naive algorithm for evaluating this 2^n-size representation on all 2^n points would take at least $2^n * 2^n = 4^n$ time! We are getting a quadratic speedup here.

Proof (Lemma). We describe the algorithm.

If $n = 1$, simply output $[p(0), p(1)]$
Otherwise, since p is multilinear, write:

$$p(x_1, ..., x_n) = p_0(x_1, ..., x_n) + x_n * p_1(x_1, ..., x_{n-1})$$

(Note that constructing p_0, p_1 just requires sorting the coefficients)

Recursively obtain the 2^{n-1}-length table T_0 for p_0, and the 2^{n-1}-length table T_1 for p_1.

Then the 2^n length table for p_0 is just $[T_0(T_0 + T_1)]$, which can be constructed in $\mathcal{O}(2^n)$ time from T_0 and T_1.

The running time is

$$R(n) \leq 2 * R(n-1) + poly(n) * min(2^n, m)$$

Which implies $R(n) \leq poly(n) * (2^n + m)$. □

Proof (W'11). Let $\epsilon > 0$ be sufficiently small, to be set later. Given an $AC^0_d[m]$ circuit C of size 2^{n^ϵ}

Let $k := n^\epsilon$. Define the $(n-k)$-input circuit $C'(y)$ as $C'(y) := \vee_{a \in \{0,1\}^k} C(a, x)$

Note:

- C is SAT $\iff$ C' is SAT
- C' has depth $\leq d + 1$

Convert C' into a SYM of ANDs with size 2^{n^ϵ} (via Beigel-Tarui). This SYM of ANDs has size quasi-polynomial in 2^{n^ϵ}, which is $2^{n^{c \cdot \epsilon}}$ for some $c >= 1$.

Now construct a multilinear polynomial $p(x_1, ..., x_{n-l})$ where each monomial corresponds to an AND gate of the SYM of ANDs (so p is taking the sum of the ANDs).

Since D is a SYM of ANDs, there exists some function $g : \{0, ..., n\} \to \{0, 1\}$ such that $D(x) = g(p(x)) \; \forall x$.

By the lemma, we can evaluate p on all $2^{n-k} = 2^{n-n^\epsilon}$ inputs in $poly(n) \cdot (2^{n^{\frac{1}{2}}} + 2^{n-n^\epsilon})$ time.

So we set $\epsilon = \frac{1}{2c}$ and the running time becomes $poly(n) \cdot (2^{n^{\frac{1}{2}}} + 2^{n-n^\epsilon})$.

Finally, compute g on all 2^{n-k} inputs, which tells us whether D is SAT or not. $\square$