An ACC satisfiability algorithm (lecture notes)

Course: Circuit Complexity, Autumn 2024, University of Chicago
Instructor: William Hoza (williamhoza@uchicago.edu)

---

1 The algorithmic method of proving circuit lower bounds

Recall that "$\mathsf{AC}^0[m]$ circuits" can use AND gates, OR gates, and MOD_m gates, all with unbounded fan-in. There are constants and literals at the bottom. When m is not a power of a prime, the class $\mathsf{AC}^0[m]$ is poorly understood. As mentioned previously in this course, it is an open problem to show $\mathsf{NP} \not\subseteq \mathsf{AC}^0[6]$.

On the bright side, there is a line of work showing that there are "somewhat explicit" functions that cannot be computed by small $\mathsf{AC}^0[6]$ circuits and similar models. In particular, Murray and Williams showed $\mathsf{NQP} \not\subseteq \mathsf{ACC}$ [MW18], where NQP is nondeterministic quasipolynomial time and $\mathsf{ACC} = \bigcup_m \mathsf{AC}^0[m]$. The full proof that $\mathsf{NQP} \not\subseteq \mathsf{ACC}$ is beyond the scope of this course, but we will present some elements of the proof. At a high level, the proof that $\mathsf{NQP} \not\subseteq \mathsf{ACC}$ has two steps. The first step is a nontrivial *satisfiability algorithm* for $\mathsf{AC}^0[m]$ circuits:

Theorem 1 (Nontrivial satisfiability algorithm for ACC)**.** *For all constants $m, d \in \mathbb{N}$, there exists a constant $\varepsilon > 0$ such that the following holds. Given the description of a depth-d $\mathsf{AC}^0_d[m]$ circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size at most 2^{n^ε}, it is possible to determine whether C is satisfiable in time 2^{n-n^ε}.*[1]

The second step (ignoring some technicalities) is to show that for any circuit class $\mathcal{C}$, if there is a nontrivial satisfiability algorithm with the parameters described above, then $\mathsf{NQP} \not\subseteq \mathcal{C}$. The second step is very interesting, but we will focus on the first step (Theorem 1) in these lecture notes.

2 Depth reduction for ACC circuits

Recall that $\mathbb{Z}[x_1, \dots, x_n]$ is the set of n-variate polynomials with integer coefficients.

Definition 1 (L_1 norm of a polynomial)**.** If $h \in \mathbb{Z}[x_1, \dots, x_n]$, then we define $L_1(h)$ to be the sum of the absolute values of the coefficients.

Definition 2 (SYM^+)**.** We define $\mathsf{SYM}^+[k]$ to be the class of functions $C\colon \{0,1\}^n \to \{0,1\}$ of the form $C(x) = g(h(x))$, where $h \in \mathbb{Z}[x_1, \dots, x_n]$ satisfies $\deg(h) \leq k$ and $L_1(h) \leq 2^k$. Note that h is multilinear without loss of generality. The function $g\colon \mathbb{Z} \to \{0,1\}$ can be arbitrary, but we emphasize that it is a function of just one integer variable.

You can double check that each function in $\mathsf{SYM}^+[k]$ can be computed by a "SYM of AND of literals," where the AND gates have fan-in at most k and the SYM gate has fan-in at most $2^{O(k)}$. Consequently, each function in $\mathsf{SYM}^+[k]$ can be computed by a TC^0_3 circuit of size $2^{O(k)}$. The following theorem is a key step in the proof of Theorem 1, as well as being interesting in its own right.

Theorem 2 (Simulating $\mathsf{AC}^0[m]$ circuits using SYM^+ circuits)**.** *Let $m, d \in \mathbb{N}$ be constants. If $C\colon \{0,1\}^n \to \{0,1\}$ is an $\mathsf{AC}^0_d[m]$ circuit of size $S \geq n$, then $C \in \mathsf{SYM}^+[\operatorname{polylog} S]$.*

When d and m are growing parameters, the best bound known is $C \in \mathsf{SYM}^+[(\log S)^{O(d \cdot s)}]$, where s is the number of distinct prime factors of m [CP19]. In these lecture notes, we assume d and m are constant for simplicity.

---

[1] We assume a *random access* model of computation throughout these lecture notes.

2.1 Simulating MOD_m gates using MOD_p gates

Lemma 1. *Let $p, e \in \mathbb{N}$ be constants, where p is prime and $e \geq 1$. Then $\mathsf{MOD}_{p^e} \in \mathsf{AC}^0[p]$.*

Proof. We prove it by induction on e. The base case $e = 1$ is trivial. For the inductive step, let $e \geq 2$, let $x \in \{0,1\}^n$, and let N be the Hamming weight of x. We claim that[2]

$$\mathsf{MOD}_{p^e}(x) = \mathsf{MOD}_p(x) \vee \mathsf{MOD}_{p^{e-1}}\left(\bigwedge_{i \in S_1} x_i, \ldots, \bigwedge_{i \in S_{\binom{n}{p}}} x_i\right), \tag{1}$$

where $S_1, S_2, \ldots, S_{\binom{n}{p}}$ is an enumeration of all size-p subsets of $[n]$. If N is not a multiple of p, this is trivial: $\mathsf{MOD}_{p^e}(x) = \mathsf{MOD}_p(x) = 1$. Now assume N is a multiple of p. In this case, observe that

$$\binom{N}{p} = \frac{N \cdot (N-1) \cdots (N-p+1)}{p \cdot (p-1) \cdots 1}.$$

In both the numerator and the denominator, only the first term is a multiple of p. Therefore, the exponent of p in the prime factorization of $\binom{N}{p}$ is one less than the exponent of p in the prime factorization of N. That is, $p^e \mid N$ if and only if $p^{e-1} \mid \binom{N}{p}$. Eq. (1) follows. By induction, Eq. (1) shows $\mathsf{MOD}_{p^e} \in \mathsf{AC}^0[p]$; note that $\mathrm{poly}(\binom{n}{p}) = \mathrm{poly}(n)$ since p is a constant. $\square$

More generally, let m be an arbitrary positive integer, with prime factorization $m = p_1^{e_1} \cdot p_2^{e_2} \cdots p_s^{e_s}$. Then

$$\mathsf{MOD}_m(x) = \mathsf{MOD}_{p_1^{e_1}}(x) \vee \cdots \vee \mathsf{MOD}_{p_s^{e_s}}(x).$$

Thus, we can simulate an $\mathsf{AC}^0[m]$ circuit using AND gates, OR gates, MOD_{p_1} gates, MOD_{p_2} gates, ..., and MOD_{p_s} gates. The depth blows up by a constant factor and the size blows up polynomially, assuming m is a constant.

2.2 Eliminating one layer of MOD_p gates

Lemma 2 (Modulus-amplifying polynomials)**.** *For every $k \in \mathbb{N}$, there exists a polynomial $M_k \in \mathbb{Z}[x]$ such that $\deg(M_k) = O(k)$, $L_1(M_k) = 2^{O(k)}$, and for every $x \in \mathbb{Z}$ and every $p \in \mathbb{N}$,*

$$\begin{aligned} x \equiv 0 \pmod{p} &\implies M_k(x) \equiv 0 \pmod{p^k} \\ x \equiv 1 \pmod{p} &\implies M_k(x) \equiv 1 \pmod{p^k}. \end{aligned} \tag{2}$$

Proof. Define

$$M_k(x) = \sum_{i=0}^{k-1} \binom{2k-1}{i} \cdot x^{2k-1-i} \cdot (1-x)^i.$$

The degree and L_1 bounds are straightforward. Observe that $M_k(x)$ is a multiple of x^k, which proves Eq. (2). Now suppose $x \equiv 1 \pmod{p}$. Then $1 - x$ is a multiple of p, so $(1-x)^i \equiv 0 \pmod{p^k}$ whenever $i \geq k$. Consequently,

$$\begin{aligned} M_k(x) &\equiv \sum_{i=0}^{2k-1} \binom{2k-1}{i} \cdot x^{2k-1-i} \cdot (1-x)^i \pmod{p^k} \\ &= (x + 1 - x)^{2k-1} && \text{by the binomial theorem} \\ &= 1. \end{aligned}$$

$\square$

[2]Recall that we defined $\mathsf{MOD}_m(x) = 1 \iff x_1 + \cdots + x_n \not\equiv 0 \pmod{m}$, which is opposite to the way many sources define it.

Lemma 3 (SYM^+ can simulate $\mathsf{SYM}^+ \circ \mathsf{MOD}_p$)**.** *Let $n, k \in \mathbb{N}$, let p be prime, and let $C\colon \{0,1\}^n \to \{0,1\}$ be a formula consisting of variables feeding into MOD_p gates feeding into a $\mathsf{SYM}^+[k]$ gate. Then $C \in \mathsf{SYM}^+[O(k^3 \cdot p \cdot \log n)]$.*

Proof. By introducing dummy variables if necessary, we can write

$$C(x) = g\left(\left(\sum_{i=1}^{L} c_i \prod_{j=1}^{k} \mathsf{MOD}_p(x_{ij1}, \ldots, x_{ij\ell})\right) \bmod p^{k+2}\right),$$

where $g\colon \mathbb{Z} \to \{0,1\}$, each $c_i \in \mathbb{Z}$, we have $\sum_{i=1}^{L} |c_i| \le 2^k$, and $\ell \le n$. (Reducing mod p^{k+2} doesn't destroy any information, because the sum lies between -2^k and 2^k.) Therefore,

$$\begin{aligned}
C(x) &= g\left(\left(\sum_{i=1}^{L} c_i \bigwedge_{j=1}^{k} \left(\sum_{t=1}^{\ell} x_{ijt} \not\equiv 0 \mod p\right)\right) \bmod p^{k+2}\right) && \text{by definition of } \mathsf{MOD}_p \\
&= g\left(\left(\sum_{i=1}^{L} c_i \cdot \mathbb{1}\left[\prod_{j=1}^{k} \sum_{t=1}^{\ell} x_{ijt} \not\equiv 0 \mod p\right]\right) \bmod p^{k+2}\right) && \text{because a product of nonzero elements is nonzero in any field, including } \mathbb{F}_p \\
&= g\left(\left(\sum_{i=1}^{L} c_i \cdot M_{k+2}\left(\left(\prod_{j=1}^{k} \sum_{t=1}^{\ell} x_{ijt}\right)^{p-1}\right)\right) \bmod p^{k+2}\right) && \text{by Fermat's little theorem and modulus amplification.}
\end{aligned}$$

The expression above has the format of SYM^+: first we apply a multivariate polynomial, and then we apply a univarate function ("reduce mod p^{k+2}, then apply g"). The degree of the polynomial is at most $k \cdot (p-1) \cdot \deg(M_{k+2}) = O(p \cdot k^2)$. The L_1 norm of this polynomial is at most $2^k \cdot L_1(M_{k+2}) \cdot (\ell^{k \cdot (p-1)})^{\deg(M_{k+2})} = n^{O(p \cdot k^3)}$. ∎

2.3 Simulating the entire circuit

Proof sketch of Theorem 2. There are several steps, but none is too difficult, given the tools that we have developed.

1. Replace each MOD_m gate with AND gates, OR gates, MOD_{p_1} gates, ..., and MOD_{p_s} gates, as described in Section 2.1.

2. Replace each AND/OR gate with a probabilistic polynomial over the field $\mathbb{F}_2$ with error $0.1/S$ and degree $\ell = O(\log S)$. Note that a degree-ℓ polynomial over $\mathbb{F}_2$ is a $\mathsf{MOD}_2 \circ \mathsf{AND}_\ell$ circuit, where the MOD_2 gate has fan-in at most $S^{O(\log S)}$. Let $\mathcal{D}$ be the resulting distribution over circuits.

3. Independently sample $t = O(n)$ circuits $C_1, \ldots, C_t \sim \mathcal{D}$ and set $C(x) = \mathsf{MAJ}_t(C_1(x), \ldots, C_t(x))$. By Hoeffding's inequality and the union bound over all $x \in \{0,1\}^n$, there is some fixing of $C_1, \ldots, C_t$ such that C computes f. Note that each C_i consists of MOD_2 gates, MOD_{p_1} gates, MOD_{p_2} gates, ..., MOD_{p_s} gates, and AND_ℓ gates (with literals and constants at the bottom).

4. By introducing dummy gates if necessary, we can ensure that *all gates at the same level are of the same type*. In other words, we can compute f using a circuit of the following form:
$$\mathsf{MAJ}_t \circ (\mathsf{MOD}_2 \circ \mathsf{MOD}_{p_1} \circ \cdots \circ \mathsf{MOD}_{p_s} \circ \mathsf{AND}_\ell)^{O(1)}.$$

5. Note that $\mathsf{MAJ}_t \in \mathsf{SYM}^+[\log t]$. We eliminate the layers underneath the SYM^+ gate one by one to get a $\mathsf{SYM}^+[k]$ circuit. To handle MOD_p layers, we use Lemma 3. To handle AND_ℓ layers, we use the trivial fact $\mathsf{SYM}^+[k] \circ \mathsf{AND}_\ell \subseteq \mathsf{SYM}^+[k \cdot \ell]$. Since the number of layers is $O(1)$, we get $f \in \mathsf{SYM}^+[\mathrm{polylog}(S)]$.

∎

3 The satisfiability algorithm

Lemma 4 (Fast multipoint evaluation of multilinear polynomials)**.** *Let $h\colon \{0,1\}^n \to \mathbb{Z}$ be a multilinear polynomial with integer coefficients. Given h, represented as a list of 2^n coefficients, it is possible to compute $h(x)$ for all $x \in \{0,1\}^n$ in time $2^n \cdot \operatorname{poly}(n) \cdot \operatorname{polylog}(L_1(h))$.*

Proof. Let $s = L_1(f)$. If $n = 0$, then the problem is trivial. Otherwise, there are polynomials h_0, h_1 such that

$$h(x_1, \dots, x_n) = h_0(x_2, \dots, x_n) + x_1 \cdot h_1(x_2, \dots, x_n). \tag{3}$$

Considered as lists of coefficients, h_0 and h_1 are simply the first half and the second half of h, respectively. In particular, $L_1(h_0) \leq s$ and $L_1(h_1) \leq s$. We recursively compute $h_0(x)$ and $h_1(x)$ for all $x \in \{0,1\}^{n-1}$, and then we use Eq. (3) to compute $h(x)$ for all $x \in \{0,1\}^n$. The time complexity of this algorithm, $T(n, s)$, satisfies

$$T(n, s) \leq 2T(n-1, s) + 2^n \cdot \operatorname{poly}(n, \log s),$$

because $|h_0(x)| \leq s$ and $|h_1(x)| \leq s$ for all $x \in \{0,1\}^{n-1}$. This implies $T(n, s) \leq 2^n \cdot \operatorname{poly}(n, \log s)$. □

Proof sketch of Theorem 1. First, let us design a satisfiability algorithm that runs in time $2^{\operatorname{polylog}(S)} + 2^n \cdot \operatorname{poly}(n)$, given a circuit of size S where $n \leq S \leq 2^n$.

1. By Theorem 2, it is possible to write C in the form $C(x) = g(h(x))$ where $h\colon \{0,1\}^n \to \mathbb{Z}$ is a multilinear polynomial satisfying $L_1(h) \leq 2^{\operatorname{polylog} S}$. We only showed that such a representation *exists*, but it turns out that it can be *constructed* in quasipolynomial ($2^{\operatorname{polylog} S}$) time, if we represent h as a list of monomials with nonzero coefficients and we represent g as a list of $2^{\operatorname{polylog}(S)}$ output values. We omit the proof that g and h can be efficiently constructed.

2. Rewrite h as a list of 2^n coefficients (many of which may be zero). This can be done in time $2^{\operatorname{polylog} S} + 2^n \cdot \operatorname{poly}(n)$.

3. Compute $h(x)$ for all $x \in \{0,1\}^n$. By Lemma 4, this can be done in time $2^n \cdot \operatorname{poly}(n)$.

4. Check whether there is some $x \in \{0,1\}^n$ such that $g(h(x)) = 1$. This can be done in time $2^n \cdot \operatorname{poly}(n)$.

Now we are ready to design a satisfiability algorithm with the parameters described in the theorem statement. Given an $\mathsf{AC}^0_d[m]$ circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size 2^{n^ε}, we define $C'\colon \{0,1\}^{n-n^\varepsilon} \to \{0,1\}$ by the rule

$$C'(x) = \bigvee_{y \in \{0,1\}^{n^\varepsilon}} C(xy).$$

Then C is satisfiable if and only if C' is satisfiable. The circuit C' is an $\mathsf{AC}^0_{d+1}[m]$ circuit of size 2^{2n^ε}, so using the algorithm described above, we can decide whether it is satisfiable in time $2^{\operatorname{poly}(n^\varepsilon)} + 2^{n-n^\varepsilon} \cdot \operatorname{poly}(n)$. The theorem follows by picking a small enough ε. □

References

[CP19] Shiteng Chen and Periklis A. Papakonstantinou. "Depth reduction for composites". In: *SIAM J. Comput.* 48.2 (2019), pp. 668–686. ISSN: 0097-5397. DOI: 10.1137/17M1129672.

[MW18] Cody Murray and Ryan Williams. "Circuit lower bounds for nondeterministic quasi-polytime: an easy witness lemma for NP and NQP". In: *Proceedings of the 50th Annual Symposium on Theory of Computing (STOC)*. 2018, 890–901. DOI: 10.1145/3188745.3188910.