CSE200: Complexity theory
Circuit lower bounds

Shachar Lovett

February 26, 2020

1 Small depth classes

The class NC (Nick's class) corresponds to boolean circuits with poly-logarithmic depth and fan-in two. The class AC (Alternating class) corresponds to boolean circuits with poly-logarithmic depth and large fan-in. They both capture computation which is highly parallel - can be computed in a poly-logarithmic number of "rounds" using many parallel processors.

Definition 1.1 (NC)**.** *The class NC^i is the class of circuits with AND/OR/NOT gates of fan-in 2, depth $O(\log(n)^i)$ and polynomial size. Define $NC = \cup_{i \geq 1} NC^i$.*

Definition 1.2 (AC)**.** *The class AC^i is the class of circuits with AND/OR/NOT gates of unbounded fan-in, depth $O(\log(n)^i)$ and polynomial size. Define $AC = \cup_{i \geq 1} AC^i$.*

Claim 1.3. *For any $i \geq 0$ it holds $NC^i \subset AC^i \subset NC^{i+1}$. In particular $NC = AC$.*

Proof. The containment $\mathrm{NC}^i \subset \mathrm{AC}^i$ is obvious. To show that $\mathrm{AC}^i \subset \mathrm{NC}^{i+1}$ note that any AND gate of unbounded fan-in m can be converted to a binary tree of fan-in 2 ANDs of depth $\log m$. As the circuits have polynomial size we have $m = \mathrm{poly}(n)$, and so the depth increases by a factor of $O(\log n)$. □

Many problems have algorithms in (uniform) NC: integer addition and multiplication, matrix multiplication and determinant, and finding a maximal matching are some examples. We suspect that NC $\neq$ P but cannot even separate NC from PH.

The class NC^0 corresponds to functions which depend just on a constant number of input bits. The class AC^0 is more interesting and can compute nontrivial functions, e.g approximate majority. We don't know how to show $\mathrm{NC}^1 \neq P$. However, we can prove lower bounds for NC^0 and AC^0, which correspond to constant depth circuits. The canonical hard function we will choose is PARITY, which computes if the sum of the bits is even or odd. It is simple to see that PARITY can be computed by a circuit of depth $O(\log n)$ and fan-in two, namely that PARITY $\in \mathrm{NC}^1$. We will prove first that PARITY is not in NC^0, and then develop the techniques to prove that also it is not in AC^0.

Theorem 1.4. *PARITY is not it NC^0.*

Proof. An NC^0 circuit of depth d can read at most 2^d bits, and so for $d = O(1)$ cannot compute PARITY. □

The main theorem we will prove here is that PARITY cannot be computed in AC^0.

Theorem 1.5. *PARITY is not it AC^0.*

More specifically, we will show that constant depth circuits that compute parity, of any fan-in, need to have an exponential size. The first super-polynomial bound of this type was proved by Furst, Saxe and Sipser [1], and this was improved to an exponential lower bound by Yao [6] and then improved to an optimal bound by Håstad [2]. The proof we will follow here is by Razborov [3] and Smolensky [5], which allows to prove that small circuits cannot even approximate PARITY. However, to stay focused, we will restrict our attention to proving lower bounds on circuits which compute PARITY exactly.

As a warm up, we first prove a lower bound for depth-2 circuits with unbounded fan-in.

Theorem 1.6. *A depth-2 circuit computing PARITY must have size $\Omega(2^n)$.*

Proof. A depth-2 circuit is essentially either a DNF or CNF. Below, we consider the case of a DNF, where the other cases are analogous.

Let $\varphi = D_1 \vee D_2 \ldots \vee D_m$ be a DNF computing the PARITY function, where each D_i is an AND of literals (variables or their negation). Let us first argue that all the terms must have n variables. Assume some term D_i has less than n variables, and let x_j be a variable not participating in D_i. We can find an assignment to the variables $\{x_i : i \neq j\}$ which make D_i true. Let us now set the variables outside D_i so that the parity will be 1 (false). Then we get that $\varphi(x) \neq PARITY(x)$ on this input.

So, we got that all terms have exactly n variables, hence they evaluate to 1 on a single input and 0 on the remaining inputs. As PARITY has 2^{n-1} inputs where it evaluates to 1, φ must have at least 2^{n-1} terms. □

The main tool to prove that PARITY $\notin \mathrm{AC}^0$ is polynomials, which we describe next.

2 Polynomials

A real-valued multilinear polynomial (which we simply call a polynomial from now on) is an expression of the form

$$p(x_1, \ldots, x_n) = \sum_{S \subseteq [n]} p_S \prod_{i \in S} x_i,$$

with coefficients $p_S \in \mathbb{R}$. Any boolean function has a unique expression as a polynomial. In fact, this holds even for functions on boolean inputs with a real-valued output.

Claim 2.1. *Any function $f : \{0,1\}^n \to \mathbb{R}$ has a unique polynomial computing it.*

Proof. Let $\mathcal{H} = \{h : \{0,1\}^n \to \mathbb{R}\}$ denote the vector space over $\mathbb{R}$ of real-valued functions on boolean inputs. Its dimension is $\dim(\mathcal{H}) = 2^n$. Any polynomial $p(x)$ defines a real-valued function on $\{0,1\}^n$. Let $\mathcal{P}$ denote the vector space of functions defined by polynomials, where clearly $\mathcal{P} \subset \mathcal{H}$. The claim would follow by showing that in fact $\mathcal{P} = \mathcal{H}$. In order to show that, we will show that $\dim(\mathcal{P}) = 2^n$.

The space $\mathcal{P}$ is spanned by the 2^n monomial functions $M_S(x) = \prod_{i \in S} x_i$ for $S \subseteq [n]$. We will show that they form a basis, which will establish the dimension. Assume towards a contradiction that they are linearly dependent. This linear dependency forms a nonzero polynomial $p(x) = \sum p_S M_S(x)$ which evaluates to 0 on all boolean inputs. We will prove that this is impossible, unless all the coefficients $p_S = 0$.

Assume not, and let $T \subseteq [n]$ be minimal such that $p_T \neq 0$. Consider the input $x = 1_T$, namely $x_i = 1$ if $i \in T$ and $x_i = 0$ if $i \notin T$. Then

$$p(1_T) = \sum_S p_S \prod_{i \in S} (1_T)_i = \sum_S p_S 1_{S \subseteq T} = p_T \neq 0.$$

Thus we get that p cannot map all boolean inputs to zero, unless it is the zero polynomial. $\square$

The *degree* of a polynomial p is the maximal $|S|$ such that $p_S \neq 0$. Namely, the largest number of variables in a monomial in p. We denote by $\mathcal{P}_k$ the family of all functions computed by polynomials of degree at most k,

$$\mathcal{P}_k = \{p(x) : \deg(p) \leq k\}.$$

Note that $\mathcal{P}_k$ is a subspace of the linear space of all functions $f : \{0,1\}^n \to \mathbb{R}$, spanned by the monomials of degree $\leq k$. Sometimes it will be convenient to represent bits as $\{-1,1\}$ instead of $\{0,1\}$. Note that we can always do this change by replacing $x_i \in \{0,1\}$ with $1 - 2x_i \in \{-1,1\}$, and that this transformation does not change the degree of the polynomial.

3 AC^0 circuits can be approximated by low-degree polynomials

As a stepping stone towards proving that PARITY is not in AC^0, we show that any function computed by an AC^0 circuit can be well-approximated by a low-degree polynomial. In general, we say that a polynomial $p(x) = p(x_1, \ldots, x_n)$ approximates a boolean function $f : \{0,1\}^n \to \{0,1\}$ with error ε if

$$\Pr_{x \in \{0,1\}^n} [p(x) \neq f(x)] \leq \varepsilon.$$

The size of the circuit is the number of gates in it, including the inputs.

Theorem 3.1. *Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function computed by an AC^0 circuit with depth d and size s. For any $\varepsilon > 0$ there exists a polynomial p of degree $(c \log(s/\varepsilon))^{2d}$ that approximates f with error ε, where $c > 0$ is some absolute constant.*

We first prove the following lemma handling a single AND gate. To recall, AND : $\{0,1\}^n \to \{0,1\}$ maps the input 1^n to 1, and all other inputs to 0.

Lemma 3.2. *Let $n \in \mathbb{N}, \varepsilon > 0$. There exists a distribution $\mathcal{D}$ over polynomials p of degree $k = O(\log(n)\log(1/\varepsilon))$ such that the following holds. For any $x \in \{0,1\}^n$,*

$$\Pr_{p\sim\mathcal{D}}[p(x) = AND(x)] \geq 1 - \varepsilon.$$

Proof. Let $1 \leq t \leq \log n$ be chosen uniformly. Let $A \subseteq [n]$ be a random set of size 2^t. Define the polynomial

$$p_A(x) = \left(\sum_{i\in A} x_i\right) - |A| + 1.$$

Observe that $p_A(1^n) = 1$ holds for any A. We will show that if $x \in \{0,1\}^n \setminus 1^n$ then

$$\Pr_A[p_A(x) = 0] \geq \alpha/\log n$$

for some absolute constant $\alpha > 0$.

To see that, assume the number of zeros in x is between 2^a and 2^{a+1} for some $0 \leq a \leq \log(n)$. Let us condition on the event that $t = \log(n) - a$, which happens with probability $1/\log n$. In such a case, $|A| = n/2^a$ and the average number of zeros in A is between 1 and 2. Moreover, the probability that there there is exactly one zero in A is at least some absolute constant $\alpha > 0$, in which case $p_A(x) = 0$.

To conclude, we construct the distribution $\mathcal{D}$. Sample $A_1, \ldots, A_k$ as above independently for $k = O(\log n \cdot \log(1/\varepsilon))$, and set

$$p(x) = p_{A_1}(x) \ldots p_{A_k}(x).$$

Then we have that:

1. $p(1^n) = 1$ holds with probability one.
2. If $x \in \{0,1\}^n \setminus 1^n$ then $\Pr[p_A(x) = 0] \geq 1 - \varepsilon$ since

$$\Pr[p_A(x) \neq 0] = \prod_{i=1}^{k} \Pr[p_{A_i}(x) \neq 0] = (1 - \alpha/\log n)^k \leq \exp(-\alpha k/\log n) \leq \varepsilon.$$

□

A similar lemma holds for OR gates. NOT gates clearly can be computed by the polynomial $p(x) = 1 - x$. We can now prove Theorem 3.1.

Proof of Theorem 3.1. Let C be a circuit of depth d and size s that computes the function f. Let $v_1, \ldots, v_s$ denote the nodes of C, such that $v_i = x_i$ for $i = 1, \ldots, n$, and such that for $i > n$ the inputs to v_i are in $\{v_j : j < i\}$. Given an input x let $v_i(x)$ denote the value compute at the gate v_i on input x, where $f(x) = C(x) = v_s(x)$. We denote by I_i the set of inputs to v_i, and by $g_i \in \{\text{AND}, \text{OR}, \text{NOT}\}$ the function computed at the node, so that

$$v_i(x) = g_i\left(v_j(x) : j \in I_i\right).$$

Set $\delta = \varepsilon/s$. As a first step, apply Lemma 3.2 or the analogous lemmas for OR or NOT gates to each gate in the circuit. For a gate i with $i > n$, we get a distribution $\mathcal{D}_i$ of polynomials p_i of degree $k = O(\log s \cdot \log(1/\delta)) = O(\log^2(s/\varepsilon))$, such that for every input x it holds that

$$\Pr_{p_i \sim \mathcal{D}_i}\left[v_i(x) \neq p_i\left(v_j(x) : j \in I_i\right)\right] \leq \delta.$$

By the union bound, with probability $1 - \delta s = 1 - \varepsilon$ all the polynomials compute the correct value. Let $p(x)$ be the composition of all these polynomials which compute $f(x) = v_s(x)$, where we also use the base case $v_i(x) = x_i$ for $i = 1, \ldots, n$. It gives a distribution $\mathcal{D}$ of polynomial p of degree k^d, which satisfy

$$\Pr_{p \sim \mathcal{D}}\left[f(x) \neq p(x))\right] \leq \varepsilon.$$

As this holds for any input x, we obtain by an averaging argument that

$$\Pr_{p \sim \mathcal{D}, x \in \{0,1\}^n}\left[f(x) \neq p(x)\right] \leq \varepsilon.$$

Thus there must be a polynomial p^* in the support of $\mathcal{D}$ such that

$$\Pr_{x \in \{0,1\}^n}\left[f(x) \neq p^*(x)\right] \leq \varepsilon.$$

In particular, p^* is a polynomial of degree at most $k^d = O(\log(s/\varepsilon))^{2d}$. □

4 PARITY cannot be approximated by low-degree polynomials

We already saw that any function computed by a small AC^0 circuit can be approximated by a low-degree polynomial. To prove that PARITY cannot be computed by a small AC^0 circuit, we will show that PARITY cannot be approximated by a low-degree polynomial. This allows to prove that PARITY cannot even be approximated by small AC^0 circuits.

Theorem 4.1. *Let $p(x)$ be an n-variate polynomial of degree k. Then*

$$\Pr_{x \in \{0,1\}^n}[p(x) = PARITY(x)] \leq \frac{1}{2} + O\left(\frac{k}{\sqrt{n}}\right).$$

Before proving Theorem 4.1, we show how it implies Theorem 1.5, namely that PARITY is not in AC^0. Specifically, we obtain the following corollary.

Corollary 4.2. *Assume that the n-bit PARITY function is computed by an AC^0 circuit with depth d and size s. Then $s \geq 2^{\Omega(n^{1/4d})}$.*

Proof. Apply Theorem 3.1 with $\varepsilon = 0.1$. There exists a polynomial $p(x)$ of degree $k = (c\log(s))^{2d}$ that approximates PARITY with error 0.1. Theorem 4.1 shows that this requires $k = \Omega(\sqrt{n})$. Hence $\log s = \Omega(n^{1/4d})$. □

Proof of Theorem 4.1. It will be convenient to view p as a polynomial over $\{-1,1\}^n$. Note that in this basis,

$$\mathrm{PARITY}(x_1,\ldots,x_n) = \prod_{i=1}^{n} x_i.$$

Let A denote the set of inputs on which p agrees with PARITY, namely

$$A = \{x \in \{-1,1\}^n : p(x) = \mathrm{PARITY}(x)\}.$$

Let $V = \{f : A \to \mathbb{R}\}$ denote the vector space of functions from A to $\mathbb{R}$. Its dimension is $\dim(V) = A$. We will show that A is small by finding a small basis for V. We already know that any function $f : \{-1,1\}^n \to \mathbb{R}$ can be written as a polynomial

$$f(x) = \sum_{S\subseteq[n]} f_S \prod_{i\in S} x_i.$$

The crucial observation is that if $x \in A$ we can replace $\prod_{i=1}^{n} x_i$ with the low degree polynomial $p(x)$. Moreover, if $|S| \geq (n+k)/2$ then for any $x \in A$:

$$\prod_{i\in S} x_i = \prod_{i\in S} x_i \cdot \prod_{i=1}^{n} x_i \cdot p(x) = \prod_{i\in[n]\setminus S} x_i \cdot p(x),$$

which is a polynomial of degree $n - |S| + k \leq (n+k)/2$. Hence, all functions $f : A \to \mathbb{R}$ can be written as polynomials of degree at most $(n+k)/2$. The dimension of this vector space is the number of monomials of degree at most $(n+k)/2$, which is

$$\sum_{i=0}^{(n+k)/2} \binom{n}{i} = \left(\frac{1}{2} + O\left(\frac{k}{\sqrt{n}}\right)\right) \cdot 2^n.$$

Thus we obtain that

$$\Pr[p(x) = \mathrm{PARITY}(x)] = 2^{-n}|A| = \frac{1}{2} + O\left(\frac{k}{\sqrt{n}}\right).$$

□

5 Natural proofs

Following the sequence of lower bounds against AC^0, there was a belief in the early 1990s that similar combinatorial or algebraic techniques can possibly prove lower bounds against P/poly, and in particular would prove super-polynomial lower bounds against some natural problems. However, in a landmark work in 1994, Razborov and Rudich [4] showed that all the known lower bound techniques (including the ones we discussed here) fall into a framework that they called "natural proofs". They showed that if we believe that cryptography is secure, then such proofs can never prove lower bounds against P/poly.

At a high level, a lower bound proof is "natural" if it identifies a property that simple functions have, but random functions do not. Here, by simple functions we mean ones computed by small circuits, say. Then to prove that a hard function is not computed by a small circuit (it is not simple), one only needs to verify that it lacks the above property. We define natural proofs for general circuit classes $\mathcal{C}$, for example $\mathrm{NC}^0, \mathrm{AC}^0$ or P/poly.

Definition 5.1. *Let $\mathcal{C}$ be a circuit class. A natural property against $\mathcal{C}$ is a subset $\mathcal{P}$ of the functions $f : \{0,1\}^* \to \{0,1\}$ that satisfy the following requirements:*

- ***Usefulness:*** *If $f \in \mathcal{C}$ then $f \notin \mathcal{P}$.*
- ***Largeness:*** *For large n, a random function $f : \{0,1\}^n \to \{0,1\}$ is in $\mathcal{P}$ with high probability.*
- ***Constructivity:*** *Given the truth table of a function $f : \{0,1\}^n \to \{0,1\}$ we can test if $f \in \mathcal{P}$ in time $2^{O(n)}$.*

As an example, let us construct a natural property against NC^0.

Claim 5.2. *There is a natural property against NC^0.*

Proof. Let $\mathcal{C}$ be the property of boolean functions f that depend on all their inputs. It is a natural property against NC^0 since:

- Usefulness: If $f \in \mathrm{NC}^0$ then it depends on a constant number of inputs, and hence not in $\mathcal{C}$.
- Largeness: a random function depends on all its inputs with high probability. To see that, note that there are 2^{2^n} boolean functions on n inputs. If a boolean function does not depend on some input x_i then it is effectively a boolean function on $n-1$ inputs, hence there are at most $n \cdot 2^{2^{n-1}}$ such functions. The ratio $n \cdot 2^{2^{n-1}}/2^{2^n} = n/2^{2^{n-1}}$ tends to zero as n grows.
- Constructivity: given a truth table of a function $f : \{0,1\}^n$ to $\{0,1\}$, we can check for all $i \in [n]$, if it holds that $f(x) = f(x^i)$ for all $x \in \{0,1\}^n$, where x^i is x with the i-th bit flipped. This takes time $n \cdot 2^n = 2^{O(n)}$.

□

Let us first prove that if we believe cryptography, then there is no natural property against P/poly. Concretely, we will rely on the assumption that there are hard one-way functions. Formally, we assume that there are functions $f : \{0,1\}^n \to \{0,1\}$ which can be computed in polynomial time, but that given only query access to f, they cannot be distinguished from completely random functions in time less than 2^{n^c} for some $c > 0$.

Theorem 5.3. *Assume that there exists a natural property $\mathcal{P}$ against P/poly. Then for any $\varepsilon > 0$ and large enough n, any function $f : \{0,1\}^n \to \{0,1\}$ computed in polynomial time can be distinguished from a random function in time 2^{n^ε}.*

Proof. Let $f : \{0,1\}^n \to \{0,1\}$ be a function computed in P/poly. Pick $\varepsilon > 0$ and set $k = n^\varepsilon$. Let $g : \{0,1\}^k \to \{0,1\}$ be the restriction of f to the first k bits of input, by fixing the remaining $n-k$ bits to zero. Note that g is also computed in P/poly. We can check if $g \in \mathcal{P}$ in time $2^{O(k)}$. By our assumption, it does not have the property. On the other hand, if f was a random function on n bits, then g would have been a random function on k bits, and hence in this case $g \in \mathcal{P}$ with high probability. Thus we can differentiate between a poly-time computable f and a random f in time $2^{O(n^\varepsilon)}$ for any $\varepsilon > 0$. □

It turns out that the proof that PARITY is not in AC^0 can be transformed into a natural property against AC^0.

Theorem 5.4. *There is a natural property against AC^0.*

Proof sketch. Let us next consider the proof we saw that PARITY is not in AC^0, and check if it is a natural proof. The first approach is to use Theorem 3.1 directly, namely that any AC^0 circuit can be approximated by a polynomial of poly-logarithmic degree. Lets consider functions $f : \{0,1\}^n \to \{0,1\}$. Define

$$\mathcal{P}_1 = \{f : f \text{ cannot be approximated by a polynomial} \\ \text{of degree } (\log n)^{O(1)} \text{ up to error } 0.1\}$$

Note that $\mathcal{P}_1$ is useful (AC^0 functions do not have the property) and large (a random function is likely to have the property). However, it is not clear why it is constructive - the natural way to test if a function f can be approximated by a polynomial of degree $k = (\log n)^c$ is to test all such polynomials, which will take too much time - exponential in the number of possible monomials, which is about n^k. To recall, for constructivity we want to allow runtime single exponential in n.

The way to make it constructive is to inspect the proof more carefully. First, it is not hard to see that any function $F : \{0,1\}^n \to \mathbb{R}$ can be decomposed as

$$F(x) = p_1(x) + p_2(x) \cdot \mathrm{PARITY}(x)$$

where p_1, p_2 are polynomials of degree $\leq n/2$. In that sense, PARITY is "universal". We would define the property that a function is universal as our natural property:

$$\mathcal{P}_2 = \{f : \text{ any function } F : \{0,1\}^n \to \mathbb{R} \text{ can be expressed as } F = p_1 + p_2 \cdot f \\ \text{where } p_1, p_2 \text{ are polynomials of degree } \leq n/2\}$$

We claim that $\mathcal{P}_2$ is useful against AC^0: if $f \in \mathrm{AC}^0$ then it can be approximated by a polynomial of degree $k = (\log n)^{O(1)}$. Hence if $F = p_1 + p_2 f$ then F can be approximated by a polynomial of degree $n/2 + (\log n)^{O(1)}$, which is not true for all functions. This implies that if $f \in \mathrm{AC}^0$ then $f \notin \mathcal{P}_2$. In addition, $\mathcal{P}_2$ is constructive, as given the truth table of f one can check if $f \in \mathcal{P}_2$ by solving a linear program of size $2^{O(n)}$, which can be done in time $2^{O(n)}$. What is now unclear is whether $\mathcal{P}_2$ is large. It seems to be true, but it is unclear how to prove it. To overcome this, we will instead consider another property:

$$\mathcal{P}_3 = \{f : \text{ the linear space of } F : \{0,1\}^n \to \mathbb{R} \text{ that can be expressed as } F = p_1 + p_2 \cdot f \\ \text{where } p_1, p_2 \text{ are polynomials of degree } \le n/2 \text{ has dimension } \ge 0.9n\}$$

This refined definition can be shown to still be useful against AC^0 and constructive, and in addition it is also large. □

References

[1] M. Furst, J. B. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical systems theory*, 17(1):13–27, 1984.

[2] J. Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 6–20. Citeseer, 1986.

[3] A. A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes*, 41(4):333–338, 1987.

[4] A. A. Razborov and S. Rudich. Natural proofs. In *Journal of Computer and System Sciences*. Citeseer, 1994.

[5] R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 77–82. ACM, 1987.

[6] A. C.-C. Yao. Separating the polynomial-time hierarchy by oracles. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 1–10. IEEE, 1985.