IAS/PCMI Summer Session 2000
Clay Mathematics Undergraduate Program
Basic Course on Computational Complexity

Lecture 10: AC^0 Circuits Cannot Compute PARITY

David Mix Barrington and Alexis Maciel
July 28, 2000

Recall the sequence of class containments we have established:

$$\mathrm{AC}^0 \subseteq \mathrm{NC}^1 \subseteq \mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{AC}^1 \subseteq \cdots \subseteq \mathrm{NC} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}.$$

The hierarchy theorems of the preceding lecture imply that one of the containments between NL and PSPACE and one of the containments between P and EXP must be strict. But we do not know which one (or which ones). In this lecture, we show that $\mathrm{AC}^0 \subset \mathrm{NC}^1$ thus establishing that a specific inclusion in the above sequence is strict. And this is the only one which is currently known to be strict.

The precise result we will prove is that the PARITY language cannot be decided by AC^0 circuits, even if they are allowed to be of size $2^{n^{o(1)}}$, even if they are allowed to include gates that test whether the sum of their inputs is non-zero modulo 3, and even if they are allowed to be completely non-uniform. Since the NC^1 parity circuits are very uniform, this is more than enough to separate AC^0 from NC^1 under any kind of uniformity condition.

1. Lower Bounds via Polynomial Representations

Consider multivariate, multilinear polynomials over $\mathbf{Z}_3$. Multilinear means that no variable can appear with exponent larger than 1. For example, $1 + 2x_1 + x_1x_2x_3$ is one such polynomial. When multiplying these polynomials, we reduce the result using the identities $x_i^2 = x_i$. For example, multiplying the above polynomial by x_2 gives $x_2 + 2x_1x_2 + x_1x_2x_3$. From now on, by polynomial over $\mathbf{Z}_3$, we will always mean multivariate, multilinear polynomials over $\mathbf{Z}_3$.

Each n-variable polynomial over $\mathbf{Z}_3$ clearly defines a function from $\{0,1\}^n$ to $\mathbf{Z}_3$. For example, $1 - x_1$ defines $\mathrm{NOT}(x_1)$, $x_1 \cdots x_n$ defines $\mathrm{AND}(x_1, \ldots, x_n)$ and

$1 - ((1 - x_1) \cdots (1 - x_n))$ defines $\mathrm{OR}(x_1, \ldots, x_n)$. It is also true that every function can be defined by some polynomial (see the exercises). Since the number of functions and polynomials is the same, 3^{2^n}, we cannot have that two polynomials define the same function. Therefore, each function is defined by a unique polynomial and we say that this polynomial *represents* the function.

We can also talk of the *degree* of a function: it is the degree of its polynomial representation. For example, the above polynomials imply that NOT has degree 1 while AND and OR both have degree n. Another example is given by the function MOD_3. In general, $\mathrm{MOD}_m(x_1, \ldots, x_n)$ is defined to be 1 if and only if the sum of its inputs is non-zero modulo m. So MOD_2, for example, is simply the characteristic function of the PARITY language. The degree of MOD_3 is 2 because this function is represented by the polynomial $(x_1 + \cdots + x_n)^2$ over $\mathbf{Z}_3$, using the fact that a^2 is congruent to 1 modulo 3 if and only if a is not congruent to 0 modulo 3.

Now what is the degree of MOD_2? The easiest way to figure this out is to set $z_i = 1 - 2x_i$. This converts 0 and 1 to 1 and -1, respectively. Then $z_1 \cdots z_n$ is 1 if MOD_2 is 0, and -1 is MOD_2 is 1. Since $x = \frac{1}{2}(1 - z)$ does the opposite conversion of $1/-1$ values to $0/1$ values, we have that MOD_2 can be written as $\frac{1}{2}(1 - z_1 \cdots z_n)$, a polynomial of degree n in both the z_i's or the x_i's.

Note that what happened in the preceding paragraph is that we found the polynomial representation of MOD_2 when viewed as a function with 1/-1 values and output and translated that representation into one for MOD_2 when viewed as a function in the $0/1$ setting. In fact, when we talk about the MOD_2 function, we really mean the Boolean function defined by $\mathrm{MOD}_2(x_1, \ldots, x_n) = \mathrm{TRUE}$ if and only if the number of TRUE x_i's is odd. Whether FALSE and TRUE are represented as 0 and 1 or as 1 and -1 is really secondary. For our purposes, the key point to keep in mind is that the degree of a function is the same in either the $0/1$ or $1/-1$ setting.

We have shown that NOT and MOD_3 have constant degrees while AND, OR and MOD_2 have degree n. What happens when a constant-depth circuit is made up entirely of gates that compute constant-degree functions? Since degrees at most multiply when polynomials are composed, it is not hard to see that the circuit computes a constant-degree function. In particular, it cannot compute AND, OR or MOD_2. In fact,

Proposition 1 *If $d = o(\log n)$, then depth-d circuits with* NOT *and* MOD_3 *gates cannot compute* AND, OR *or* MOD_2.

We say that constant-degree functions are *easy* while functions of degree n are *hard*.

The above proposition is our first example of a lower bound established by using the degree of polynomial representations. Can we use the same idea to show that MOD_2 cannot be computed by AC^0 circuits? The obvious problem is that AC^0 circuits contain both easy and hard gates, so they can compute hard functions.

What we are going to do is show that the functions AND and OR are, in fact, *not that hard*, in the following sense:

Lemma 2 *Let $f_1, \ldots, f_m$ be functions of a common input x. For every k, there is a polynomial p of degree $2k$ such that when x is chosen at random (with each value equally likely), the probability that $p(f_1(x), \ldots f_m(x)) \neq \mathrm{OR}(f_1(x), \ldots f_m(x))$ is no greater than $1/2^k$.*

The same obviously holds for the AND function. We say that these functions can be *approximated* by polynomials of degree k with probability of error no greater than $1/2^k$.

We will then use this to show that if a small-depth circuit with NOT, AND, OR and MOD_3 gates is not too large, then the function it computes can be well-approximated by a polynomial of degree $o(\sqrt{n})$. More generally,

Lemma 3 *If $d = o(\log n)$ and C is depth-d circuit with an arbitrary number of constant-degree gates and $2^{o(n^{1/2d})}$ gates that can be well-approximated in the sense of Lemma 2, then there is a polynomial p of degree $o(\sqrt{n})$ such that when x is chosen at random, the probability that $p(x) \neq C(x)$ is $o(1)$.*

Finally, to get our circuit lower bound, we will show that MOD_2 is harder than that: it cannot be approximated by such polynomials.

Lemma 4 *If p is a polynomial of degree $o(\sqrt{n})$, then, when x is chosen at random, the probability that $p(x) \neq \mathrm{MOD}_2(x)$ is $1/2 - o(1)$.*

Putting the last two lemmas together, we get our main result:

Theorem 5 *If $d = o(\log n)$ and C is depth-d circuit with an arbitrary number of constant-degree gates and $2^{o(n^{1/2d})}$ gates that can be well-approximated in the sense of Lemma 2, then, when x is chosen at random, the probability that $C(x) \neq \mathrm{MOD}_2(x)$ is $1/2 - o(1)$.*

Corollary 6 *If C is a depth-d circuit with* NOT, AND, OR *and* MOD_3 *gates and C computes* MOD_2*, then either $d = \Omega(\log n)$ or C contains $2^{\Omega(n^{1/2d})}$* AND *and* OR *gates.*

Corollary 7 *If C is a constant-depth circuit with* NOT, AND, OR *and* MOD_3 *gates computing* MOD_2*, then C contains $2^{n^{\Omega(1)}}$* AND *and* OR *gates.*

2. The OR Function Is Not That Hard

In this section, we prove Lemma 2.

Proof (Of Lemma 2.) Fix an input x and denote by $f(x)$ the sequence $f_1(x), \ldots, f_m(x)$. Let $h = (c_1 f_1 + \cdots c_m f_m)^2$ be a polynomial with variables $f_1, \ldots, f_m$ and coefficients c_i randomly chosen from $\{0, 1\}$ (with each value equally likely). If $\mathrm{OR}(f(x)) = 0$, then $h(f(x))$ will always be 0, so the probability that $h(x) \neq \mathrm{OR}(f(x))$ is 0. If $\mathrm{OR}(f(x)) = 1$, then $f_i(x) = 1$ for some i. For every setting of the other coefficients, there is at most one value of c_i that can cause $h(f(x))$ to evaluate to 0. Therefore, the probability that $h(f(x)) \neq \mathrm{OR}(f(x))$ is at most $1/2$.

Now let $g = \mathrm{OR}(h_1, \ldots, h_k)$ be the OR of k independent copies of h. If $\mathrm{OR}(f(x)) = 0$, then all the h_i's evaluate to 0, so the probability that $g(f(x)) \neq \mathrm{OR}(f(x))$ is 0. If $\mathrm{OR}(f(x)) = 1$, then the probability that $g(f(x)) \neq \mathrm{OR}(f(x))$ is at most $1/2^k$. Note that g has degree $2k$ (in terms of the f_i's).

We have shown that for every x, when the degree-$2k$ polynomial g is chosen at random, then the probability that $g(f(x)) \neq \mathrm{OR}(f(x))$ is at most $1/2^k$. This implies that there is a degree-$2k$ polynomial g such that when x is chosen at random, then the probability that $g(f(x)) \neq \mathrm{OR}(f(x))$ is at most $1/2^k$. This can be established by a simple averaging argument (see the exercises). The polynomial g is the polynomial we were looking for. □

Using this, we now prove Lemma 3 that circuits with not too many easy and "not too hard" gates can be well-approximated by polynomials of degree $o(\sqrt{n})$.

Proof (Of Lemma 3.) Suppose the circuit contains 2^r gates that are not easy but can be well-approximated in the sense of Lemma 2. Let $k = r + n^{1/4d}$. Note that $k = o(n^{1/2d})$. Compose all the polynomials associated with each of the gates. The degree of the resulting polynomial p is at most $(2k)^d = o(\sqrt{n})$. In the worst case, all these polynomials disagree with their gates for different values of x. Therefore, the probability that $p(x) \neq C(x)$ is at most $2^r(\frac{1}{2})^k = (\frac{1}{2})^{n^{1/4d}} = o(1)$. □

3. The MOD_2 Function Is Much Harder

In this section, we prove Lemma 4, that the MOD_2 function cannot be well-approximated by polynomials of degree $\sqrt{n}$. First, we show that in the $1/-1$ setting, every function can be represented in terms of the MOD_2 function and two polynomials of degree at most $n/2$.

Lemma 8 *In the $1/-1$ setting, every function f can be written as $g\mathrm{MOD}_2 + h$ where g and h both have degree at most $n/2$.*

Proof Take a polynomial p representing f. The function h will consist of all the monomials in p that are of degree at most $n/2$. Now consider a monomial in p that has degree greater than $n/2$. Then that monomial can be written as $\prod_{i \in I} z_i = (z_1 \cdots z_n) \prod_{i \notin I} z_i$. The various monomials $\prod_{i \notin I} z_i$ will be the monomials of g. Note that the degree of each of these monomials is at most $n/2$. □

Proof (Of Lemma 4.) Convert p to a polynomial representing MOD_2 in the $1/-1$ setting. p still has degree $o(\sqrt{n})$ and its probability of error in representing the MOD_2 function is the same. Let D be the values of x where $p(x) = \mathrm{MOD}_2(x)$. We want to show that $|D|/2^n = 1/2 + o(1)$.

By the previous lemma, every function can be written as $g\mathrm{MOD}_2 + h$ where g and h both have degree at most $n/2$. This implies that every function with inputs in D, can be written as $q = gp + h$ where g and h both have degree at most $n/2$. Note that q has degree at most $n/2 + o(\sqrt{n})$. Therefore, all this implies that the number of functions on D cannot be larger than the number of polynomials of degree at most $n/2 + o(\sqrt{n})$, that is,

$$3^{|D|} \leq 3^{\sum_{d=0}^{n/2+o(\sqrt{n})} \binom{n}{d}}.$$

It is well-known that $\sum_{d=0}^{n/2+o(\sqrt{n})} \binom{n}{d} = (1/2+o(1))2^n$. Therefore, $|D|/2^n = 1/2+o(1)$. □

4. Lower Bounds for Larger Classes

In this and the previous lecture, we managed to prove some lower bounds. We established the Space Hierarchy Theorem by the diagonalization technique and we proved a lower bound for AC^0 by using low-degree polynomial representations. But still, in our chain of class containments,

$$\mathrm{AC}^0 \subseteq \mathrm{NC}^1 \subseteq \mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{AC}^1 \subseteq \cdots \subseteq \mathrm{NC} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP},$$

only one inclusion is known to be strict: $\mathrm{AC}^0 \subset \mathrm{NC}^1$. Can we use these two lower bound techniques to separate more classes?

For example, the Time Hierarchy Theorem implies that $\mathrm{P} \subset \mathrm{EXP}$. Can we push the technique further and show that $\mathrm{P} \subset \mathrm{NP}$? The answer is no. Without getting into the details (see Section 9.2 of Sipser's textbook for that), diagonalization, as well as other techniques based on machines simulating other machines, is a technique that *relativizes*, meaning that whatever is established also holds relative to any *oracle*. And it has been shown that $\mathrm{P} = \mathrm{NP}$ relative to some oracles, and $\mathrm{P} \subset \mathrm{NP}$ relative to others. Therefore, the P versus NP question will not be solved by such techniques.

In fact, these limitations of the diagonalization method were some of the original motivation for studying circuits classes such as AC^0. Since $\mathrm{P} \subseteq$ non-uniform PSIZE, to show that $\mathrm{P} \neq \mathrm{NP}$, it is enough to show that $\mathrm{NP} \not\subseteq$ non-uniform PSIZE. And it seemed like circuits would provide a setting that would lead more easily to the discovery of radically new lower bound techniques that do not relativize.

The plan then was to prove lower bounds for small classes and work our way up. AC^0 lower bounds were the starting point. However, proving lower bounds even for seemingly limited circuit classes has proved to be very difficult. In this lecture, we actually showed that constant-depth, polynomial-size circuits with unbounded fan-in NOT, AND, OR and MOD_3 gates cannot compute the MOD_2 function. Such circuits define the class $\mathrm{ACC}^0[3]$. The argument can be generalized to show that $\mathrm{ACC}^0[p]$ circuits cannot compute MOD_q if p and q are distinct primes (see the exercises). But no one knows how to push this further to a lower bound for $\mathrm{ACC}^0[m]$ circuits when m is composite. One obstacle is that $\mathbf{Z}_m$ is no longer a field and we can no longer rely on basic facts such as a^{m-1} is congruent to 1 modulo m if and only if a is not congruent to 0 modulo m.

Leaving aside circuits with modular gates, can we prove lower bounds for constant-depth, polynomial-size circuits with unbounded fan-in NOT, AND, OR and MAJORITY gates? These circuits define the class TC^0. (MAJORITY is the set of x's that have a majority of 1's.) The class TC^0 has proved to be surprisingly powerful. For example, let $\mathrm{ACC}^0 = \bigcup_{m \geq 2} \mathrm{ACC}^0[m]$. While it is easy to see that $\mathrm{ACC}^0 \subseteq \mathrm{TC}^0$, it is also true that non-uniform, depth-3 TC^0 circuits of size $n^{\log^{O(1)} n}$ can compute all of ACC^0. And non-uniform depth-3 TC^0 circuits, as well as DLOGTIME-uniform TC^0 circuits (as shown in the Advanced Course), can compute significant arithmetic functions such as DIVISION and ITERATED MULTIPLICATION.

When considering proving lower bounds for TC^0, we run into another difficulty. All the lower bound arguments that have been used so far on circuits have been characterized as *natural proofs*. Again without getting into the details, natural proofs cannot prove lower bounds for classes such as P or even TC^0, unless some widely

believed conjectures turn out to be false. Therefore, radically new ideas seem to be once again required.

5. Exercises

1. Show that every function from $\{0,1\}$ to $\mathbf{Z}_3$ can be defined by some polynomial over $\mathbf{Z}_3$.

2. Show that the binary AND function can be computed by a MOD_3 gate of constant fan-in. Use this to show that all of NC^1, including the AND, OR and MOD_2 functions, can be computed by constant fan-in, depth-$O(\log n)$ circuits with NOT and MOD_3 gates.

3. Show that constant-depth circuits with NOT gates, unbounded fan-in MOD_3 gates and one level of AND, OR and MOD_{10} gates of fan-in $o(n)$ can not compute the AND, OR or MOD_2 functions.

4. Say that a polynomial p over $\mathbf{Z}_3$ *weakly defines* a Boolean function f if $p(x) = 0$ when $f(x) =$ FALSE and $p(x) = 1$ or 2 when $f(x) =$ TRUE. Show that no polynomial of degree less than $n/2$ can weakly define the MOD_2 function.

5. Suppose that for every x, when g is chosen at random, the probability that $g(x) \neq \mathrm{OR}(f(x))$ is at most $1/2^k$. Show that there is g such that when x is chosen at random, then the probability that $g(x) \neq \mathrm{OR}(f(x))$ is at most $1/2^k$.

6. Explain why $\sum_{d=0}^{n/2+o(\sqrt{n})} \binom{n}{d} = (1/2 + o(1))2^n$. (Hint: Consider the normal approximation to the binomial distribution.)

7. Generalize the lower bound argument presented in this lecture to show that $\mathrm{ACC}^0[p]$ circuits cannot compute MOD_q if p and q are distinct primes. (Hint: Work with polynomials over a finite field of characteristic p with a qth root of unity.)