Notes for Lecture 22

Today we continue our discussion of computing parity with bounded-depth circuits. First, we complete the polynomial-based proof from the previous lecture that parity is not in $\mathbf{AC_0}$. Next, we show that parity is not in $\mathbf{AC_0}$ through a proof with random restrictions on the circuit's inputs.

1 Approximating OR

In the last lecture, we we needed a way of approximating OR with a low-degree polynomial. The following lemma says that we can approximately represent OR with a polynomial of degree exponentially small in the the fan-in of the OR gate. We'll use the notation that x is a vector of k bits, x_i is the ith bit of x, and $\mathbf{0}$ is the vector of zeros (of the appropriate size based on context).

Lemma 1 *For all k and ϵ, there exists a distribution G over functions $g : \{0,1\}^k \to \mathbb{R}$ such that*

1. *g is a degree $O((\log \frac{1}{\epsilon})(\log k))$ polynomial, and*
2. *for all $x \in \{0,1\}^k$,*

$$\Pr_{g \sim G}[g(x) = x_1 \vee \ldots \vee x_k] \geq 1 - \epsilon. \tag{1}$$

Proof Idea: We want a random polynomial $p : \{0,1\}^k \to \mathbb{R}$ that computes OR. An obvious choice is

$$p_{\text{bad}}(x_1, \ldots, x_k) = 1 - \prod_{i \in \{1,\ldots,k\}} (1 - x_i)\,, \tag{2}$$

which computes OR with no error. But it has degree k, whereas we'd like it to have logarithmic degree. To accomplish this amazing feat, we'll replace the tests of all k variables with just a few tests of random batches of variables. This gives us a random polynomial which computes OR with one-sided error: when $x = \mathbf{0}$, we'll have $p(x) = 0$; and when some $x_i = 1$, we'll almost always (over our choice of p) have $p(x) = 1$.

Proof: We pick a random collection C of subsets of the bits of x. (That is, for each $S \in C$ we have $S \subseteq \{1, \ldots, k\}$). We'll soon see how to pick C, but once the choice has been made, we define our polynomial as

$$p(x_1, \ldots, x_k) = 1 - \prod_{S \in C} \left(1 - \sum_{i \in S} x_i\right). \tag{3}$$

Why does p successfully approximate OR? First, suppose $x_1 \vee \ldots \vee x_k = 1$. Then we have $x = \mathbf{0}$, and:

$$p(0, \ldots, 0) = 1 - \prod_{S \in C} \left(1 - \sum_{i \in S} 0\right) = 0. \tag{4}$$

So, regardless of the distribution from which we pick C, we have

$$\Pr_C [p(\mathbf{0}) = 0] = 1. \tag{5}$$

Next, suppose $x_1 \vee \ldots \vee x_k = 1$. We have $p(x) = 1$ iff the product term is zero. The product term is zero iff the sum in some factor is 1. And that, in turn, happens iff there's some $S \in C$ which includes *exactly one* x_i which is 1. Formally, for any $x \in \{0,1\}^k$, we want the following to be true with high probability.

$$\exists S \in C. \, (|\{i \in S : x_i = 1\}| = 1) \tag{6}$$

Given that we don't want C to be very large (so that the degree of the polynomial is small), we'll have to pick C very carefully. In order to accomplish this, we turn to the Valiant-Vazirani reduction, which you may recall from Lecture 7:

Lemma 2 (Valiant-Vazirani) *Let $A \subseteq \{1, \ldots, k\}$, let a be such that $2^a \leq |A| \leq 2^{a+1}$, and let H be a family of pairwise independent hash functions of the form $h : \{1, \ldots, k\} \to \{0,1\}^{a+2}$. Then if we pick h at random from H, there is a constant probability that there is a unique element $i \in A$ such that $h(i) = \mathbf{0}$. Precisely,*

$$\Pr_{h \sim H} [|\{i \in A : h(i) = \mathbf{0}\}| = 1] \geq \frac{1}{8} \tag{7}$$

With this as a guide, we'll define our collection C in terms of pairwise independent hash functions. Let $t > 0$ be a value that we will set later in terms of the approximation parameter ϵ. Then we let $C = \{S_{a,j}\}_{a \in \{0,\ldots,\log k\}, j \in \{1,\ldots,t\}}$ where the sets $S_{a,j}$ are defined as follows.

- For $a \in \{0, \ldots, \log k\}$:
  - For $j \in \{1, \ldots, t\}$:
    - Pick random pairwise independent hash function $h_{a,j} : \{1, \ldots, k\} \to \{0,1\}^{a+2}$
    - Define $S_{a,j} = h^{-1}(\mathbf{0})$. That is, $S_{a,j} = \{i : h(i) = \mathbf{0}\}$.

Now consider any $x \neq \mathbf{0}$ which we're feeding to our OR-polynomial p. Let A be the set of bits of x which are 1, i.e., $A = \{i : x_i = 1\}$, and let a be such that $2^a \leq |A| \leq 2^{a+1}$. Then we have $a \in \{0, \ldots, \log k\}$, so C includes t sets $S_{a,1}, \ldots, S_{a,t}$. Consider any one such $S_{a,j}$. By Valiant-Vazirani, we have

$$\Pr_{h_{a,j} \sim H} [|\{i \in A : h_{a,j}(i) = \mathbf{0}\}| = 1] \geq \frac{1}{8} \tag{8}$$

which implies that

$$\Pr_{h_{a,j} \sim H} [|\{i \in A : i \in S_{a,j}\}| = 1] \geq \frac{1}{8} \tag{9}$$

so the probability that there is some j for which $|S_{a,j} \cap A| = 1$ is at least $1 - \left(\frac{7}{8}\right)^t$, which by the reasoning above tells us that

$$\Pr_p [p(x) = x_1 \vee \ldots \vee x_k] \geq 1 - \left(\frac{7}{8}\right)^t. \tag{10}$$

Now, to get a success probability of $1-\epsilon$ as required by the lemma, we just pick $t = O(\log \frac{1}{\epsilon})$. The degree of p will then be $|C| = t(\log k) = O((\log \frac{1}{\epsilon})(\log k))$, which satisfies the degree requirement of the lemma. $\Box$

Note that given this lemma, we can also approximate AND with an exponentially low degree polynomial. Suppose we have some G which approximates OR within ϵ as above. Then we can construct G' which approximates AND by drawing g from G and returning g' such that

$$g'(x_1, \dots, x_k) = 1 - g(1 - x_1, \dots, 1 - x_k). \tag{11}$$

Any such g' has the same degree as g. Also, for a particular $x \in \{0,1\}^k$, g' clearly computes AND if g computes OR, which happens with probability at least $1 - \epsilon$ over our choice of g.

2 Random Restrictions

In this section we give an alternate proof that parity $\notin \mathbf{AC_0}$ using the technique of random restrictions. This is the original method that was used to prove parity $\notin \mathbf{AC_0}$ in [FSS84] and improved in [Y85, H86]. The polynomial-based technique used in the last lecture was introduced in several years later in [S87].

A *restriction* fixes some inputs of a circuit to constant values, and leaves other inputs free. More formally, a restriction is a function $\rho : \{1, \dots, n\} \to \{0, 1, *\}$. We apply a restriction ρ to a circuit C with n inputs to obtain a restricted circuit C_ρ as follows:

- For each $i \in \{1, \dots, n\}$:
  - If $\rho(i) = 0$, set the ith input of C to 0.
  - If $\rho(i) = 1$, set the ith input of C to 1.
  - If $\rho(i) = *$, leave the ith input of C as a free variable.

Now let's see how restrictions are used to prove our main theorem:

Theorem 3 *There does not exist a constant-depth, unbounded-fanin, polynomial-size circuit which computes the parity function of its inputs.*

PROOF IDEA: We'll consider only circuits which at the first level consist only of ORs, at the second level consist only of ANDs, and continue alternating in this way. This is depicted in Figure 1. (Any circuit can be coverted to this form with at most a factor of 2 increase in the depth.)

The proof will be by contradiction. First, we show that parity circuits of depth 2 must have exponential width (Lemma 4). Next, we suppose by way of contradiction that we have an $\mathbf{AC_0}$ circuit C of some constant depth d which computes parity. We give a way of squashing C down to depth $d-1$ while still computing parity on many variables. We can repeat the method $d-2$ times to obtain a parity circuit of depth 2 and subexponential size, which contradicts Lemma 4.

Restrictions enter in our method of circuit squashing. We apply a random restriction ρ_1 to C's inputs, obtaining C_{ρ_1}. We prove that the top level OR gates of C_{ρ_1} are likely to only

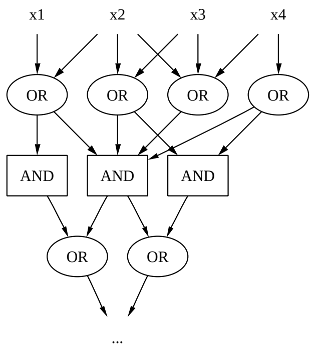

Figure 1: A circuit with alternating levels of ANDs and ORs.

depend on a constant number of inputs, but a good number of variables still remain free (Corollary 6). C_{ρ_1} will compute parity (or the inverse of parity, which is just as good) on those free variables. Then we show that after applying a second random restriction ρ_2 and obtaining $C_{\rho_1 \rho_2}$, the second-level AND gates are also likely to depend on only a constant number of inputs (Lemma 7). Now we flip the first two layers, which we can do with at most an exponential size increase of those layers (Lemma 8), which is fine because the size of both layers was made constant by the two rounds of restrictions. After the flip, the second and third layers are both ORs, so we can collapse them both into a single layer, obtaining a circuit of depth $d-1$ that computes parity on many variables.

Statements and proofs of the lemmas mentioned above follow.

Lemma 4 *If a DNF or a CNF computes parity of n variables, then:*

1. *Each term includes all n variables, and*
2. *There are at least 2^{n-1} terms.*

PROOF: We will prove the lemma for CNFs, which have OR gates at their top level and a single AND of all the ORs at the second level. The proof for DNFs is quite similar.

For any CNF circuit C:

1. *Each term includes all n variables:* Suppose by way of contradiction that C has some term t which does not depend on some variable x_i. Then when all inputs to t are 0, t outputs 0 and the single AND gate on the next level outputs 0, which is the output of the whole circuit. Now flip the value of x_i. The output of t is still 0, and thus the output of C has not changed. But since we've only changed one variable, the parity has flipped. Alas, we have a contradiction! So every term must depend on all variables.

2. *There are at least 2^{n-1} terms:* To compute parity, C must output 0 on 2^{n-1} different settings of the input variables. C outputs 0 only when one of the terms (OR gates) outputs 0. But each OR gate outputs 0 on exactly one setting of the input variables. Thus, C must have at least 2^{n-1} terms.

□

Lemma 5 *Let ρ be a random restriction defined as*

$$\rho(i) = \begin{cases} 0 & \textit{with probability } \frac{1}{2} - \frac{1}{\sqrt{n}} \\ 1 & \textit{with probability } \frac{1}{2} - \frac{1}{\sqrt{n}} \\ * & \textit{with probability } \frac{2}{\sqrt{n}} \end{cases} \tag{12}$$

Then an OR or AND gate with n inputs restricted with ρ depends on at most c inputs with probability at least $1 - \frac{1}{n^{c/3}}$.

Proof Idea: We'll break the proof into two cases. (1) Large OR gates will have some input fixed at 1 w.h.p., and thus the entire gate can be replaced with a constant 1. Similarly, large AND gates can be replaced with 0 w.h.p. (2) Small gates will w.h.p. be reduced enough to depend on only c inputs.

Proof:

Case 1: Suppose the gate has at least $c \log n$ inputs before applying the restriction. Then:

$$\begin{aligned} \Pr_\rho[\text{gate depends on} \leq c \text{ inputs}] &\geq \Pr_\rho[\text{gate becomes constant}] \\ &\geq \Pr_\rho[\text{some input set to 1 (OR gate) or 0 (AND gate)}] \\ &\geq 1 - \left(\frac{1}{2} + \frac{1}{\sqrt{n}}\right)^{c \log n} \\ &\geq 1 - \left(\sqrt{\frac{1}{2}}\right)^{c \log n} \quad \text{(for large enough } n\text{)} \\ &= 1 - \frac{1}{n^{c/2}} \\ &\geq 1 - \frac{1}{n^{c/3}}. \end{aligned}$$

Case 2: Suppose the gate has at most $c \log n$ inputs. Then:

$$\begin{aligned} \Pr_\rho[\text{gate depends on} \leq c \text{ inputs}] &\geq 1 - \sum_{t=c+1}^{c \log n} \binom{c \log n}{t} \left(\frac{2}{\sqrt{n}}\right)^t \\ &\geq 1 - (c \log n)^c \left(\frac{1}{\sqrt{n}}\right)^c \\ &\geq 1 - \frac{1}{n^{c/3}} \text{ (for large enough } n\text{)}. \end{aligned}$$

□

Corollary 6 *If a circuit has n^k gates, then there is a restriction with $\frac{\sqrt{n}}{2}$ free variables such that all top-level gates have fan-in at most $4k$.*

PROOF: We apply Lemma 5 with $c = 4k$, obtaining a circuit C_ρ such that

$$\Pr\left[\text{some gate of } C_\rho \text{ depends on } > 4k \text{ variables}\right] < n^k \frac{1}{n^{4k/3}} = n^{-1/3} \ll 1 \qquad (13)$$

and

$$\Pr\left[C_\rho \text{ depends on } < \frac{\sqrt{n}}{2} \text{ variables}\right] \approx 2^{O(-\sqrt{n})} \ll 1 \qquad (14)$$

which follows from a Chernoff bound. This guarantees the existance of a circuit which satisfies the corollary. □

Lemma 7 *Given any c-CNF C (in which each OR depends on at most c variables) and any $k > 0$, there is a constant $b(c,k)$ such that after applying the random restriction ρ of Lemma 5 to C, with probability more than $1 - \frac{1}{n^k}$, C_ρ depends on at most $b(c,k)$ variables.*

PROOF: The proof is by induction on c.

Base case: $c = 1$. The OR gates each only have 1 input, so we can replace the entire CNF with a single AND gate. Then by Lemma 5, Lemma 7 is satisfied when $b(1,k) = 3k$.

Inductive case: Suppose the lemma holds for $c - 1$; we'll prove that it's true for c. Let M be a maximal set of disjoint clauses. That is, the clauses in M share no variables (disjointedness), and every clause not in M shares a variable with some clause in M (maximality). (You can think of this as picking a maximal matching in the hypergraph whose nodes are the circuit's variables and whose hyperedges are the sets of variables used by each clause.) One of two cases is true:

Case 1: $|M| \geq 2^c k \log n$. Then:

$$\begin{aligned}
\Pr_\rho\left[C \text{ depends on} \leq b(c,k) \text{ inputs}\right] &\geq \Pr_\rho\left[C \text{ becomes constant}\right] \\
&\geq \Pr_\rho\left[\text{some OR gate unsatisfied}\right] \\
&\geq 1 - \left(1 - \left(\frac{1}{2} - \frac{1}{\sqrt{n}}\right)^c\right)^{|M|} \\
&\approx 1 - e^{-\frac{1}{2^c}|M|} \\
&> 1 - \frac{1}{n^k}.
\end{aligned}$$

Note that in the third inequality we use the fact that the clauses of M are independent: the probability that a particular OR gate in M is unsatisfied is independent of the probability that any other OR gate in M is unsatisfied.

Case 2: $|M| \leq 2^c k \log n$. Let V be the set of variables in the clauses of M, so $|V| \leq c|M| = c2^c k \log n$. We'll first show that the restriction reduces V to constant size:

$$\begin{aligned}
\Pr_\rho\left[\leq i \text{ variables of } V \text{ stay free}\right] &\geq 1 - \binom{2^c k \log n}{i}\left(\frac{1}{\sqrt{n}}\right)^i \\
&\geq 1 - \frac{1}{n^{i/3}} \text{ (for large enough } n).
\end{aligned}$$

So to get the probability greater than $1 - \frac{1}{n^k}$ as required by the lemma, we choose $i = 4k$. But we're not yet done: we've shown that the gates of M shrink to constant size, but we have said nothing about the gates not in M. To do that, recall that every clause not in M shares a variable with a clause in M, so if we restrict the remaining $4k$ free inputs of M, obtaining C'_ρ, then every clause in C'_ρ will have some input fixed. C'_ρ is thus a $(c-1)$-CNF and we can apply the inductive hypothesis: C'_ρ depends on at most $b(c-1, k+1)$ variables. Now, there are 2^{4k} ways to set the $4k$ variables, so C_ρ depends on at most

$$b(c,k) = 4k + 2^{4k} b(c-1, k+1) \tag{15}$$

variables.

□

Lemma 8 (The Switching Lemma) *A CNF or DNF of c variables can be represented as a DNF or a CNF, respectively, of size at most 2^c.*

We omit the proof of this lemma, but note that we can actually prove a stronger fact: *any* function of c variables can be written as a DNF or as a CNF of size at most 2^c.

References

[FSS84] M. Furst, J. Saxe, and M. Sipser. *Parity, circuits, and the polynomial hierarchy*, Mathematical Systems Theory 17, pp. 13-27, 1984.

[H86] J. Hastad. *Almost Optimal Lower Bounds for Small Depth Circuits*, Proceedings of the 18th ACM Symposium on Theory of Computing, pp. 6-20, 1986.

[S87] R. Smolensky. *Algebraic methods in the Theory of Lower Bounds for Boolean Circuit Complexity*, Proceedings of the 19th ACM Symposium on Theory of Computing, pp. 77-82, 1987.

[Y85] A. Yao. *Separating the Polynomial Hierarchy by Oracles*, Proceedings of the 26th IEEE Symposium on Foundations of Computer Science, pp. 1-10, 1985.