Yale University
Department of Computer Science

The Perceptron Strikes Back

Richard Beigel
Yale University

Nick Reingold
Yale University

Daniel Spielman
Yale College

YALEU/DCS/TR-813
September 11, 1990

The Perceptron Strikes Back*

preliminary report

Richard Beigel[†] Yale University

Nick Reingold[†] Yale University

Daniel Spielman[‡] Yale College

Abstract

We show that circuits composed of a symmetric gate at the root with AND-OR subcircuits of constant depth can be simulated by probabilistic depth-2 circuits with essentially the same symmetric gate at the root and AND gates of small fanin at the bottom. In particular, every language recognized by a depth-d AC^0 circuit is decidable by a probabilistic perceptron of size $2^{O\left(\log^{5d} n\right)}$ and order $O\left(\log^{5d} n\right)$ that uses $O\left(\log^3 n\right)$ probabilistic bits. As a corollary, we present a new proof that depth-d AND-OR circuits computing the parity of n binary inputs require size $2^{n^{\Omega(1/d)}}$.

1. Introduction

An impetus for the study of circuit complexity has been its relationship with computational complexity [9]. The inspiration for this paper is the discovery by Toda and Ogiwara [14] that every language in PH is probabilistically polynomial time m-reducible to a language in PP.

Studied since the 1950s, perceptrons are depth-2 threshold circuits consisting of a single threshold gate at the root with AND-gates at the next level. Though perceptrons were proposed as a powerful component of vision systems, Minksy and Papert's monograph [12] showed that they were computationally very weak. Thus debunked, perceptrons all but disappeared from the scientific literature. Recently, however, circumstantial evidence for the computational power of perceptrons has been generated by the success of neural nets, which are networks of perceptrons. In this paper, we show that even a single perceptron is powerful enough to recognize every language in AC^0. To be precise, we show that probabilistic perceptrons of size $2^{\text{polylog}\, n}$ and order

*The authors may be reached by writing to Department of Computer Science, P.O. Box 2158, Yale Station, New Haven, CT 06520-2158, or by sending electronic mail to lastname-firstname@cs.yale.edu.

[†]Supported in part by NSF grants CCR-8808949 and CCR-8958528.

[‡]Supported in part by NSF grant CCR-8958528 under an REU supplement.

polylog n can simulate AC^0 circuits (constant-depth, unbounded-fanin AND-OR circuits of polynomial size) with small error probability. This result and others in this paper were independently discovered by Jun Tarui (personal communication, 1990). Thus we vindicate the reputation of the much maligned perceptron.

Contrasting results are known for deterministic perceptrons. For example, Fu Bin [7] has shown that there are depth-3 AC^0 predicates which cannot be computed by deterministic circuits consisting of a small weighted threshold of AND-gates of polylog fanin. Beigel [5] has some related results.

Our approach is inspired by Allender [1], who showed that every AC^0 circuit can be simulated by a probabilistic depth-2 circuit with a parity gate at the root and $2^{\text{polylog}\, n}$ AND gates of fanin polylog n on the bottom. Allender and Hertrampf [2] extended this result by reducing the number of random bits used. Hence they concluded that every AC^0 circuit can be simulated by a deterministic depth-3 threshold circuit.

2. Probabilistic Polynomials and Circuits

Definition 1. A *probabilistic circuit* is a circuit with two types of inputs: actual inputs $x_1, \ldots, x_n$ from $\{0,1\}$, and probabilistic inputs $\rho_1, \ldots, \rho_l$ where each ρ_i is chosen uniformly and independently from $\{0,1\}$. If C is a probabilistic circuit, then for any given $x_1, \ldots, x_n$ we think of the output $C(x_1, \ldots, x_n)$ as a random variable. Let C be a probabilistic circuit, and let f be a function of n inputs. *C computes f with error ϵ* if for every $x_1, \ldots, x_n$, the probability that $C(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ is at least $1 - \epsilon$.

Definition 2. A *probabilistic polynomial* is a polynomial in two types of variables: actual variables $x_1, \ldots, x_n$ from $\{0,1\}$, and probabilistic variables $\rho_1, \ldots, \rho_l$ where each ρ_i is chosen uniformly and independently from $\{0,1\}$. If P is a probabilistic polynomial, then for any given $x_1, \ldots, x_n$ we think of $P(x_1, \ldots, x_n)$ as a random variable. Let P be a probabilistic polynomial, and let f be a function of n variables. *P computes f with error ϵ* if for every $x_1, \ldots, x_n$, the probability that $P(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ is at least $1 - \epsilon$.

Definition 3. A *perceptron of size s and order m* is a depth-2 circuit with a threshold gate at the top and s AND-gates of fanin m on the bottom.

We will make use of the following result due to Valiant and Vazirani [15]:

Theorem 4. *Let $S \subseteq \{0,1\}^k$ be nonempty. Suppose $w_0, w_1, \ldots, w_k$ are randomly chosen from $\{0,1\}^k$. Let $S_0 = S$ and let*

$$S_i = \{v \in S : v \cdot w_0 \equiv v \cdot w_1 \equiv \cdots \equiv v \cdot w_i \equiv 0 \pmod 2\}$$

for each $i \in \{0, \ldots, k\}$, where $\cdot$ denotes inner product. Let $Pr_k(S)$ be the probability that $|S_i| = 1$ for some $i \in \{0, \ldots, k\}$. Then $Pr_k(S) \geq 1/4$.

Lemma 5. *For any $\epsilon > 0$, there exists a probabilistic polynomial of degree $O\left(\log(1/\epsilon)\log^2(n)\right)$, with n actual variables and $O\left(\log(1/\epsilon)\log^2(n)\right)$ probabilistic variables which computes the OR of the n actual variables with error probability at most ϵ.*

Proof: Let $b_0, \ldots, b_{n-1}$ be the actual variables and let $k = \lceil \log n \rceil$. Consider elements of $\{0,1\}^k$ as numbers from 0 to $n-1$ represented in binary. Let $S = \left\{v \in \{0,1\}^k : b_v = 1\right\}$, and let the S_i's be as in Theorem 4. The bits of each w_i are taken from distinct sets of the probabilistic bits of size k. Notice that

$$|S_i| = \sum_{v \in \{0,1\}^k} \left(b_v \wedge \left(v \cdot w_0 \equiv v \cdot w_1 \equiv \cdots \equiv v \cdot w_i \equiv 0 \pmod 2\right)\right). \tag{1}$$

Each summand in (1) can be computed as a polynomial in at most $O\left(\log^2 n\right)$ variables, where b_v and each bit in the w_i's are considered separate variables. Thus, for each $i \in \{0, \ldots, k\}$ we can compute $P_i = |S_i|$ as a polynomial with the inputs and the probabilistic bits as its variables. Consider $P = \prod_{i=1}^{k}(1 - P_i)$. P is a probabilistic polynomial with the b_i's as its actual inputs and the bits of the w_i's as its probabilistic inputs. If every b_i is zero, then every $|S_i| = 0$, so $P(b_0, \ldots, b_{n-1}) = 1$. If at least one b_i is 1, then, by Theorem 4, $P(b_0, \ldots, b_{n-1}) = 0$ with probability at least $1/4$.

We can amplify the probability by repeating the test, each time using new probabilistic bits. For each test, we construct a new probabilistic polynomial P which differs only in the probabilistic bits used. Let P' be the probabilistic polynomial obtained by multiplying together $O\left(\log(1/\epsilon)\right)$ different such P's. Notice that if every b_i is zero, then $P'(b_0, \ldots, b_{n-1}) = 1$, and if at least one b_i is one, then $P'(b_0, \ldots, b_{n-1}) = 0$ with probability at least $1 - \epsilon$. Since we repeat the test $O\left(\log(1/\epsilon)\right)$ times, we use $O\left(\log(1/\epsilon)\log^2 n\right)$ probabilistic bits in all. ∎

Lemma 6. *For any $\epsilon > 0$ and any function f computed by a depth-d, size-s AND-OR circuit with n inputs, there exists a probabilistic polynomial of degree $O\left(\left(\log(1/\epsilon)\log^2(n)\log(s)\right)^d\right)$ of n actual variables and $O\left(\log(1/\epsilon)\log^2(n)\right)$ probabilistic variables that computes f with error probability at most ϵ.*

Proof: Change each AND-gate in the circuit to the negation of an OR-gate of the negations of its inputs. By Lemma 5, for each OR-gate in the circuit there is a probabilistic polynomial that computes the value of the gate as a function of its inputs and the probabilistic bits with error probability at most ϵ/s. Let $\hat{P}$ be the composition of these polynomials. $\hat{P}$ has degree $O\left(\left(\log(1/\epsilon)\log^2(n)\log(s)\right)^d\right)$ and computes f with error probability at most ϵ. ∎

Definition 7. A set of gates G is *discerning* if for any sequence of weights $w_1, \ldots, w_f$ there exists a gate $g \in G$ of fanin f such that for all $x_1, \ldots, x_f$, if $\sum_{i=1}^{f} x_i w_i = 0$ then $g(x_1, \ldots, x_f) = 0$ and if $\sum_{i=1}^{f} x_i w_i = 1$ then $g(x_1, \ldots, x_f) = 1$. For example, the collection of all (weighted) threshold gates is discerning.

Theorem 8. *For any $\epsilon > 0$, any discerning set of gates G, and any function f computed by a depth-d, size-s AND-OR circuit with n inputs, there exists a probabilistic depth-2 circuit of size $2^{O\left(\left(\log(1/\epsilon)\log^2(n)\log(s)\right)^d\right)}$ composed of a gate from G at the root and AND gates of fanin $O\left(\left(\log(1/\epsilon)\log^2(n)\log(s)\right)^d\right)$ that uses $O\left(\log(1/\epsilon)\log^2(n)\right)$ probabilistic bits which computes f with error probability at most ϵ.*

Proof: By Lemma 6, there exists a probabilistic polynomial that computes f with error probability ϵ. Let $\hat{P}$ be such a polynomial. Since the variables of the polynomial are idempotent, it can be expanded into $2^{O\left(\left(\log(1/\epsilon)\log^2(n)\log(s)\right)^d\right)}$ monomials of degree $O\left(\left(\log(1/\epsilon)\log^2(n)\log(s)\right)^d\right)$, each of which can be computed by an AND gate of fanin $O\left((\log(1/\epsilon)\log^2(n)\log(s))^d\right)$. Thus, f can be computed with error probability at most ϵ by a gate from G with these AND gates as inputs. ∎

Definition 9. A *symmetric gate* is a gate with inputs $x_1, \ldots, x_n$ which computes $h(\sum_{i=1}^{n} x_i)$ for some underlying function h. We say that two symmetric gates are of the same *type* if the two underlying functions are the same.

Theorem 10. *For any $\epsilon > 0$ and any function f computed by a depth-d circuit of n inputs with a symmetric gate g at the root and m size-s AND-OR subcircuits, there exists a probabilistic depth-2 circuit with a symmetric gate of the same type as g at the root and $2^{O\left(\left(\log(m/\epsilon)\log^2(n)\log(ms)\right)^d\right)}$ AND gates of fanin $O\left(\left(\log(m/\epsilon)\log^2(n)\log(ms)\right)^d\right)$ on the bottom that uses $O\left(\log(m/\epsilon)\log^2(n)\right)$ probabilistic bits that computes f with error probability ϵ.*

Proof: By Lemma 6, for each subcircuit which computes an input to G, there exists a polynomial of degree $O\left(\left(\log(m/\epsilon)\log^2(n)\log(s)\right)^d\right)$ that computes the value of the subcircuit with error probability at most ϵ/m. As in Theorem 8, each polynomial can be computed by a weighted sum of AND gates. So, there is a symmetric gate of the same type as g with these AND gates as inputs which computes f with error probability ϵ. ∎

Corollary 11. *For any function f computed by a depth-d, size $2^{\log^k n}$ AND-OR circuit of n inputs, there exists a probabilistic perceptron of size $2^{O\left(\log^{2d(k+1)} n\right)}$ and order $O\left(\log^{2d(k+1)} n\right)$ that uses $O\left(\log^{k+2} n\right)$ probabilistic bits and computes f with error probability $1/2^{O\left(\log^k n\right)}$.*

Corollary 12. *For any polynomial p, and any function f computed by a depth-d AC^0 circuit with n inputs, there exists a probabilistic perceptron of size $2^{O\left(\log^{5d} n\right)}$ and order $O\left(\log^{5d} n\right)$ that uses $O\left(\log^3 n\right)$ probabilistic bits and computes f with error probability $O\left(1/p(n)\right)$.*

3. Parity and AC^0: a structural approach

There have been many proofs that the parity function is not computed by any AND-OR circuit having small depth and size [16, 4, 8, 11, 13]. We present a new proof based on our result and a result due to Aspnes, Beigel, Furst, and Rudich [3].

Suppose that the parity of n binary inputs is computed by an AND-OR circuit having size s and depth d. Then there exists a probabilistic perceptron having order $(\log(s))^{O(d)}$ that computes parity with probability at least 3/4. Therefore there is a deterministic perceptron of order $(\log(s))^{O(d)}$ that computes parity correctly on at least 3/4 of all inputs. As shown in [3], such a perceptron must have order $\Omega(\sqrt{n})$. Therefore

$$s = 2^{n^{\Omega(1/d)}}.$$

4. A Suggestion on Terminology

Many papers [9, 10, 1, 5, 17, 6] have considered circuits whose bottom level consists of AND-gates or OR-gates having fanin polylog n, while gates at other levels have unbounded fanin. Perceptrons have a single threshold gate at the root and polylog-fanin AND-gates at the bottom level. Since they have only one threshold gate, it is unnatural for us to think of perceptrons as depth-2 threshold circuits. We prefer to call them depth-1^+ threshold circuits. In general we suggest the notation "depth-d^+" for circuits having d levels of unbounded fanin gates, followed by one level consisting of polylog-fanin AND-gates or polylog-fanin OR-gates. For example, Yao's result [17] says that every bounded depth ACC circuit can be simulated by a deterministic depth-2^+ threshold circuit of size $2^{\text{polylog}\, n}$.

5. Concluding Remarks

It is a routine matter to verify that all the polynomials constructed in this paper have coefficients of size $2^{\text{polylog}(n/\epsilon)}$, so the threshold gates used in the circuits built above have weights of size $2^{\text{polylog}(n/\epsilon)}$ as well. Beigel [5] and independently Bin [7] have found depth-2^+ AC^0 predicates which cannot be computed by any deterministic perceptron with weights $2^{\text{polylog}\, n}$ and order polylog n.[1] Thus, we find that randomization strictly increases the power of perceptrons. Similar findings regarding randomness are known in many other areas of complexity theory. We wonder whether the use of randomness can assist in the design of more general neural nets.

Acknowledgments. We would like to thank Mike Fischer for helpful discussions and advice.

[1] Bin's predicate cannot be computed by such a perceptron regardless of weights.

References

[1] Eric Allender. A note on the power of threshold circuits. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 580–584, 1989.

[2] Eric Allender and Ulrich Hertrampf. On the power of uniform families of constant depth threshold circuits. Manuscript, 1990.

[3] Jim Aspnes, Richard Beigel, Merrick Furst, and Steven Rudich, 1990. Manuscript in progress.

[4] L. Babai. A random oracle separates PSPACE from the polynomial hierarchy. *Information Processing Letters*, 26:51–53, 1987.

[5] Richard Beigel. Polynomial interpolation, threshold circuits, and the polynomial hierarchy. Manuscript, January 1990.

[6] Richard Beigel, Nick Reingold, and Daniel Spielman. PP is closed under intersection. Technical Report YALEU/DCS/TR-803, Yale University, Dept. of Computer Science, 1990.

[7] Fu Bin. Separating PH from PP by relativization. Preprint, 1990.

[8] Jin-yi Cai. With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 21–29, 1986.

[9] Merrick Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, April 1984.

[10] A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, and G. Turán. Threshold circuits of boudned depth. In *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science*, pages 99–110, 1987.

[11] Johan Hastad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.

[12] Marvin L. Minksy and Seymour A. Papert. *Perceptrons*. MIT Press, Cambridge, MA, 1988. Expanded Edition. The first edition appeared in 1968.

[13] Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 77–82, 1987.

[14] Seinosuke Toda and Mitsunori Ogiwara. Counting classes are as hard as the polynomial-time hierarchy. Manuscript, 1990.

[15] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, 1985.

[16] Andrew C. Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1985.

[17] Andrew C. Yao. On ACC and threshold circuits. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, 1990. To appear.