Powering Requires Threshold Depth 3

Alexander A. Sherstov

The University of Texas at Austin, Department of Computer Sciences, Austin, TX 78712 USA

Abstract

We study the circuit complexity of the powering function, defined as $\mathrm{POW}_m(Z) = Z^m$ for an n-bit integer input Z and an integer exponent $m \leqslant \mathsf{poly}(n)$. Let $\widehat{\mathsf{LT}}_d$ denote the class of functions computable by a depth-d polynomial-size circuit of majority gates. We give a simple proof that $\mathrm{POW}_m \notin \widehat{\mathsf{LT}}_2$ for any $m \geqslant 2$. Specifically, we prove a $2^{\Omega(n/\log n)}$ lower bound on the size of any depth-2 majority circuit that computes POW_m. This work generalizes Wegener's earlier result that the squaring function (i.e., POW_m for the special case $m = 2$) is not in $\widehat{\mathsf{LT}}_2$. Our depth lower bound is optimal due to Siu and Roychowdhury's matching upper bound: $\mathrm{POW}_m \in \widehat{\mathsf{LT}}_3$.

The second part of this research note presents several counterintuitive findings about the membership of arithmetic functions in the circuit classes $\widehat{\mathsf{LT}}_1$ and $\widehat{\mathsf{LT}}_2$. For example, we construct a function $f(Z)$ such that $f \notin \widehat{\mathsf{LT}}_1$ but $5f \in \widehat{\mathsf{LT}}_1$. We obtain similar findings for $\widehat{\mathsf{LT}}_2$. This apparent brittleness of $\widehat{\mathsf{LT}}_1$ and $\widehat{\mathsf{LT}}_2$ highlights a difficulty in proving lower bounds for arithmetic functions.

Keywords: Computational complexity, threshold circuits, circuit lower bounds, complexity of arithmetic operations.

1 Introduction

An essential problem in complexity theory is understanding the complexity of fundamental arithmetic operations. Among the most studied computational models in this respect is that of a polynomial-size circuit of majority gates [2, 4, 7, 8, 9, 10]. This model is surprisingly powerful; in fact, it is conceivable that every function in P is computable by a depth-3 polynomial-size majority circuit. Thus, a natural complexity measure in this model is circuit depth. Let $\widehat{\mathsf{LT}}_d$ denote the class of functions computable by a depth-d polynomial-size circuit of majority gates. The exact

Email address: `sherstov@cs.utexas.edu` (Alexander A. Sherstov).

depth requirement ($\widehat{\mathsf{LT}}_2$, $\widehat{\mathsf{LT}}_3$, etc.) is known for most arithmetic operations; see Table 1 for an overview, as well as similar surveys by Wegener [11] and Goldmann and Karpinski [3].

Operation	**Lower Bound**	**Upper Bound**
Addition	2, folklore	2 [8]
Multiple addition	2, folklore	2 [10]
Multiplication	3 [4]	3 [10]
Multiple multiplication	3 [4]	4 [10]
Division	3 [11, 5]	3 [10]
Squaring	3 [11]	3 [10]
Powering	3, this paper	3 [10]

Table 1. Best upper and lower bounds on the depth of poly-size majority circuits for basic arithmetic operations. All operations take n-bit integers as arguments. *Addition* and *multiplication* take two arguments. *Multiple addition* and *multiple multiplication* take $\mathsf{poly}(n)$ arguments. *Division* returns the quotient $\lfloor X/Y \rfloor$ of two integers X, Y. *Squaring* computes X^2 for an integer X. *Powering* computes the m-th power X^m of an integer X, where $2 \leqslant m \leqslant \mathsf{poly}(n)$.

This paper addresses one of the remaining questions in the complexity of arithmetic operations. We study the powering function, defined as $\mathrm{POW}_m(Z) = Z^m$ for an n-bit integer input Z and a fixed integer exponent m with $2 \leqslant m \leqslant \mathsf{poly}(n)$. Siu and Roychowdhury [10] gave an explicit depth-3 polynomial-size majority circuit for computing POW_m, thus proving that $\mathrm{POW}_m \in \widehat{\mathsf{LT}}_3$. A natural question to ask is whether depth 3 is in fact optimal. A concise proof due to Wegener [11] shows that $\mathrm{POW}_2 \notin \widehat{\mathsf{LT}}_2$, meaning that Siu and Roychowdhury's construction has optimal depth for the special case $m = 2$. Wegener leaves open the question for general m. We settle this issue, proving that $\mathrm{POW}_m \notin \widehat{\mathsf{LT}}_2$ for all m. We actually prove the following more delicate result:

Theorem 1.1 (Complexity of powering) *Let $\mathrm{POW}_m(Z) = Z^m$ denote the powering function, where Z is an n-bit integer and m is an arbitrary integer with $2 \leqslant m \leqslant \mathsf{poly}(n)$. Then any majority vote of linear threshold gates that computes POW_m has size $2^{\Omega(n/\log n)}$.*

Our proof uses a reduction from IP (inner product mod 2), a function whose hardness for threshold circuits has been extensively studied. Given a circuit C that computes powering, we show how to compute IP by padding the input to C with zeroes in a suitable way. This general approach is a natural one to pursue since reductions from IP underlie all other lower bounds in Table 1, except those for addition and multiple addition.

The second part of this paper (Section 4) documents a counterintuitive fact we discovered while studying the complexity of powering. Suppose we have an arithmetic function $f : \mathbb{N} \to \mathbb{N}$ with $f \notin \widehat{\mathsf{LT}}_2$, i.e., not every output bit of f can be computed by an $\widehat{\mathsf{LT}}_2$ circuit. Can we claim that likewise $kf \notin \widehat{\mathsf{LT}}_2$ for every rational constant $k > 0$? (Here kf stands for the usual product $k \cdot f(x)$.) One is tempted to answer "yes," but we show several counterexamples. We obtain similar results for $\widehat{\mathsf{LT}}_1$. Specifically, we prove the following:

Theorem 1.2 ($\widehat{\mathsf{LT}}_1$ and $\widehat{\mathsf{LT}}_2$ not closed under scaling) *There are (explicitly given) functions $f, g, h : \{0,1\}^{2n} \to \{0,1\}^{\mathsf{poly}(n)}$ such that*

(a) $f \notin \widehat{\mathsf{LT}}_1$ *but* $5f \in \widehat{\mathsf{LT}}_1$;
(b) $g \notin \widehat{\mathsf{LT}}_1$ *but* $\frac{1}{5}g \in \widehat{\mathsf{LT}}_1$ (*here g always outputs a multiple of* 5);
(c) $h \notin \widehat{\mathsf{LT}}_2$ *but* $(1+2^n+\cdots+2^{(n-1)n})^{-1}h \in \widehat{\mathsf{LT}}_2$ (*here h always outputs a multiple of* $1+2^n+\cdots+2^{(n-1)n}$).

In other words, multiplying a hard arithmetic function by a constant can make it easy. This unexpected fact shows that $\widehat{\mathsf{LT}}_1$ and $\widehat{\mathsf{LT}}_2$ lack basic closure properties when it comes to arithmetic functions, which complicates proving good lower bounds.

2 Preliminaries

We view Boolean functions as mappings $\{0,1\}^n \to \{0,1\}$, and arithmetic functions as mappings $\{0,1\}^n \to \{0,1\}^{\mathsf{poly}(n)}$. A *linear threshold gate* with Boolean inputs $f_1, f_2, \ldots, f_t \in \{0,1\}$ is a Boolean function of the form

$$a_1 f_1 + a_2 f_2 + \cdots + a_t f_t \geqslant \theta$$

for some reals $a_1, a_2, \ldots, a_t, \theta$. In particular, every a majority gate is a linear threshold gate. The *size* of a circuit is the number of gates in it. An arithmetic function f is *computable in* $\widehat{\mathsf{LT}}_d$ if every output bit of f is computable by an $\widehat{\mathsf{LT}}_d$ circuit.

Our main result exploits a reduction from *inner product mod* 2, defined as

$$\mathrm{IP}(x_1, y_1, x_2, y_2, \ldots, x_{n/2}, y_{n/2}) = \sum x_i y_i \bmod 2.$$

The complexity of IP for depth-2 threshold circuits is well understood:

Theorem 2.1 (Nisan [6]) *Any majority vote of linear threshold gates that computes* IP *on n variables has size* $2^{\Omega(n)}$.

We follow the typographical convention of denoting $\mathsf{poly}(n)$-bit integers by capital letters (such as Z), and $O(\log n)$-bit integers by small letters (such as m).

3 Complexity of Powering

As outlined in the Introduction, our strategy will be to reduce the task of computing IP to that of computing POW_m. We accomplish this by first reducing IP to an intermediate problem, that of "generalized" squaring, and then reducing from squaring to POW_m. This first reduction, Lemma 3.1 below, is a slight generalization of Wegener's IP-to-squaring reduction [11].

Lemma 3.1 (From IP to squaring; cf. Wegener [11]) *Consider the function $f(Z) = kZ^2$, where $k \geqslant 1$ is an integer and Z is an n-bit input. Any circuit that computes $f(Z)$ can also compute* IP *on $\Omega\left(\frac{n}{\log k + \log n}\right)$ bits.*

Proof: We are given a circuit that computes $f(Z) = kZ^2$. To compute $\mathrm{IP}(x_1, y_1, \ldots, x_t, y_t)$, we construct the input Z as follows:

$$Z = y_1 \underbrace{00\ldots0}_{\ell} y_2 \underbrace{00\ldots0}_{\ell} \ldots\ldots \underbrace{00\ldots0}_{\ell} y_t \underbrace{00\ldots0}_{\ell} x_t \underbrace{00\ldots0}_{\ell} \ldots\ldots \underbrace{00\ldots0}_{\ell} x_2 \underbrace{00\ldots0}_{\ell} x_1.$$

Each pair of consecutive bits above is separated by $\ell = 2 + \lceil \log k + \log t \rceil$ zeroes. It is easy to check that the binary representation of kZ^2 will contain the integer

$$2k(x_1y_1 + x_2y_2 + \cdots + x_ty_t)$$

as a block starting in the $(2t-1)(\ell+1)$th bit position. In particular, the binary representation of kZ^2 features $\mathrm{IP}(x_1, y_1, \ldots, x_t, y_t)$ in some fixed bit position. To finish the proof, we set t as large as possible while keeping the bit length of Z at most n. This yields $t = \Omega(n/(\log k + \log n))$. $\square$

We now proceed to the crux of our proof, the squaring-to-powering reduction.

Lemma 3.2 (From squaring to powering) *Consider the function $\mathrm{POW}_m(Z) = Z^m$, where $m \geqslant 2$ is an integer and Z is an n-bit input. Any circuit that computes $\mathrm{POW}_m(Z)$ can also compute $f(W) = \binom{m}{2} W^2$, where W is up to $\frac{1}{3}n - \frac{2}{3}\lceil \log m \rceil$ bits long.*

Proof: We are given a circuit that computes $\mathrm{POW}_m(Z) = Z^m$. To compute $f(W) = \binom{m}{2} W^2$ on t bits, we construct its input as follows:

$$Z = W \underbrace{000\ldots0001}_{\ell}.$$

Then

$$
\begin{aligned}
Z^m &= (2^\ell W+1)^m = \sum_{i=0}^{m}\binom{m}{i}2^{i\ell}W^i \\
&= \underbrace{1+2^\ell mW}_{A} + 2^{2\ell}\underbrace{\binom{m}{2}W^2}_{f(W)} + \underbrace{2^{3\ell}\sum_{i=3}^{m}\binom{m}{i}2^{(i-3)\ell}W^i}_{B}.
\end{aligned}
$$

It is easy to verify that for $\ell = 2t + 2\lceil \log m \rceil$, the integers A, $f(W)$, and B above occur as disjoint blocks in the binary representation of Z^m in positions $0, 2\ell$, and 3ℓ, respectively. In particular, the circuit computes $f(W)$ as desired. To finish the proof, we set t as large as possible while keeping the bit length of Z at most n. This yields $t = \frac{1}{3}n - \frac{2}{3}\lceil \log m \rceil$. □

Combining Lemmas 3.1 and 3.2 immediately yields the following IP-to-powering reduction:

Theorem 3.3 *Any circuit that computes* $\mathrm{POW}_m(Z) = Z^m$, *where* $m \geqslant 2$ *is an integer and* Z *is an* n*-bit input, can also compute* IP *on* $\Omega\left(\frac{n}{\log m + \log n}\right)$ *bits.*

Recalling the hardness result for IP from Section 2, we readily deduce our main theorem:

Theorem 1.1. (Restated from page 2.) *Let* $\mathrm{POW}_m(Z) = Z^m$ *denote the powering function, where* Z *is an* n*-bit integer and* m *is an arbitrary integer with* $2 \leqslant m \leqslant \mathsf{poly}(n)$. *Then any majority vote of linear threshold gates that computes* POW_m *has size* $2^{\Omega(n/\log n)}$.

Proof: Immediate from Theorems 2.1 and 3.3. □

4 $\widehat{\mathsf{LT}}_1$ and $\widehat{\mathsf{LT}}_2$ Not Closed Under Scaling

In Section 3, we used a reduction from IP to prove that the generalized squaring function kZ^2 is hard for $\widehat{\mathsf{LT}}_2$. It may seem that we could have achieved the same result simply by invoking Wegener's proof [11] that the squaring function Z^2 is hard for $\widehat{\mathsf{LT}}_2$. Unfortunately, this approach does not work: there are functions $f : \{0,1\}^n \to \{0,1\}^{\mathsf{poly}(n)}$ such that $f \notin \widehat{\mathsf{LT}}_2$ but $kf \in \widehat{\mathsf{LT}}_2$ for some rational $k > 0$. Demonstrating this phenomenon is the subject of this section.

We adopt vector notation for specifying a concrete function $f : \{0,1\}^n \to \{0,1\}^{\mathsf{poly}(n)}$. For example, we write $f(x) = (x_1 \wedge x_2, 0, x_1 \oplus x_2)$ to define a function $f : \{0,1\}^n \to \{0,1\}^3$ whose output bits are, from most to least significant,

$x_1 \wedge x_2, 0, x_1 \oplus x_2$.

Theorem 1.2. (Restated from page 3.) *There are (explicitly given) functions* $f, g, h : \{0,1\}^{2n} \to \{0,1\}^{\mathsf{poly}(n)}$ *such that*

(a) $f \notin \widehat{\mathsf{LT}}_1$ *but* $5f \in \widehat{\mathsf{LT}}_1$;
(b) $g \notin \widehat{\mathsf{LT}}_1$ *but* $\frac{1}{5}g \in \widehat{\mathsf{LT}}_1$ (*here g always outputs a multiple of* 5);
(c) $h \notin \widehat{\mathsf{LT}}_2$ *but* $(1+2^n+\cdots+2^{(n-1)n})^{-1}h \in \widehat{\mathsf{LT}}_2$ (*here h always outputs a multiple of* $1+2^n+\cdots+2^{(n-1)n}$).

Proof: To prove (a), consider the function $f(x) = (x_1, 0, x_1 \oplus x_2, 0, x_2)$. Then $f \notin \widehat{\mathsf{LT}}_1$ because the parity $x_1 \oplus x_2$ cannot be computed by a linear threshold gate [1]. However, $5f(x) = 4f(x) + f(x) = (x_1, x_1 \wedge \overline{x_2}, x_2, \overline{x_1} \wedge x_2, x_1, 0, x_2)$, and thus $5f \in \widehat{\mathsf{LT}}_1$.

To prove (b), consider the function $g(x) = (x_1, x_1 \wedge x_2, x_1 \oplus x_2, 0, x_2)$. Clearly $g \notin \widehat{\mathsf{LT}}_1$ because $x_1 \oplus x_2 \notin \widehat{\mathsf{LT}}_1$. However, $g(x) = (0,0,x_1,0,x_2) + (x_1,0,x_2,0,0) = 5(0,0,x_1,0,x_2)$, and thus $\frac{1}{5}g(x) = (0,0,x_1,0,x_2)$, which is in $\widehat{\mathsf{LT}}_1$.

To prove (c), consider the function

$$h'(x,y) = (x_n \wedge y_n, \underbrace{0,0,\ldots,0}_{n-1}, x_{n-1} \wedge y_{n-1}, \underbrace{0,0,\ldots,0}_{n-1}, \ldots, \underbrace{0,0,\ldots,0}_{n-1}, x_1 \wedge y_1),$$

which is in $\widehat{\mathsf{LT}}_2$. Note that the number $k \coloneqq 1+2^n+\cdots+2^{(n-1)n}$ has the binary representation

$$k = \overbrace{1\underbrace{00\ldots0}_{n-1}1\underbrace{00\ldots0}_{n-1}\ldots1\underbrace{00\ldots0}_{n-1}1}^{n \text{ ones}}.$$

Now define $h(x,y) = kh'(x,y)$. It is easy to verify that the $(n-1)n$th bit of $h(x,y)$ is $\mathrm{IP}(x,y)$, and thus $h \notin \widehat{\mathsf{LT}}_2$ by Theorem 2.1. To summarize, $h \notin \widehat{\mathsf{LT}}_2$ but $\frac{1}{k}h = h' \in \widehat{\mathsf{LT}}_2$, as desired. $\square$

Acknowledgments

I would like to thank Adam Klivans and Ingo Wegener for helpful feedback on earlier versions of this manuscript.

References

[1] J. Aspnes, R. Beigel, M. Furst, and S. Rudich. The expressive power of voting polynomials. In *STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 402–409, New York, NY, USA, 1991. ACM Press.

[2] M. Goldmann, J. Håstad, and A. A. Razborov. Majority gates vs. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992.

[3] M. Goldmann and M. Karpinski. Simulating threshold circuits by majority circuits. *SIAM J. Comput.*, 27(1):230–246, 1998.

[4] A. Hajnal, W. Maass, P. Pudlák, G. Turán, and M. Szegedy. Threshold circuits of bounded depth. *J. Comput. Syst. Sci.*, 46(2):129–154, 1993.

[5] T. Hofmeister and P. Pudlák. A proof that division is not in TC_2^0. Research Report 447, University of Dortmund, Dept. of Computer Science, 1992.

[6] N. Nisan. The communication complexity of threshold gates. In *Proceedings of "Combinatorics, Paul Erdos is Eighty"*, pages 301–315, 1993.

[7] A. A. Razborov. On small depth threshold circuits. In *SWAT '92: Proceedings of the Third Scandinavian Workshop on Algorithm Theory*, pages 42–52, London, UK, 1992. Springer-Verlag.

[8] K.-Y. Siu and J. Bruck. On the power of threshold circuits with small weights. *SIAM J. Discrete Math.*, 4(3):423–435, 1991.

[9] K.-Y. Siu, J. Bruck, T. Kailath, and T. Hofmeister. Depth efficient neural networks for division and related problems. *IEEE Transactions on Information Theory*, 39(3):946–956, 1993.

[10] K.-Y. Siu and V. P. Roychowdhury. On optimal depth threshold circuits for multiplication and related problems. *SIAM J. Discrete Math.*, 7(2):284–292, 1994.

[11] I. Wegener. Optimal lower bounds on the depth of polynomial-size threshold circuits for some arithmetic functions. *Inf. Process. Lett.*, 46(2):85–87, 1993.