A simple function that requires exponential size read-once branching programs

Anna Gál *

Abstract

We present a Boolean function in n variables that is computable in depth 2 monotone AC^0 but requires $2^{\Omega(\sqrt{n})}$ size read-once branching programs. The function we consider is defined by the blocking sets of a finite projective plane.

Keywords Computational complexity, read-once branching programs, projective planes.

1 Introduction

A number of papers have presented lower bounds for read-once branching programs. Exponential lower bounds are given for explicit functions in [2, 3, 5, 8, 11, 12, 13, 14, 16, 17, 18, 21]. Results for the more general case of read-k-times branching programs appear in [4, 9].

We list some functions which have previously been shown to require exponential size read-once branching programs (this is not intended to be a complete list): the *Hamiltonian-Circuit* and the *Perfect-Matching* problems [5], the function taking value 1 if and only if the input graph on m vertices

*DIMACS (Center for Discrete Mathematics and Theoretical Computer Science) and Department of Computer Science, Princeton University, Princeton, NJ 08544-2087. E-mail: panni@cs.princeton.edu

is $m/2$-regular [16], integer multiplication [13], the *Clique-Only* function, defined as having value 1 if the input represents the edges of a graph on m vertices which is an $m/2$ size clique [14, 18, 21]. In contrast, the *Clique-Only* function can be computed by polynomial size read-twice branching programs [18] and by NC^1 circuits [19] (polynomial size logarithmic depth constant fan-in circuits). In the above examples, the exponential lower bounds are of the form $2^{\Omega(\sqrt{n})}$, where n is the number of variables. A lower bound of $2^{\Omega(n)}$ for an n-variable function is given in [2, 3] for the *Triangle-Parity* problem, i.e., for the function taking the value 1 if and only if the input graph contains an odd number of triangles.

The class AC^0 is the class of families of Boolean functions that can be computed by polynomial size constant depth unbounded fan-in Boolean circuits. As a consequence of [1, 6] none of the above families of functions belongs to AC^0 (see [7, 20] for stronger results).

Jukna [8] and Krause et al. [12] exhibit a function which is AC^0 computable and at the same time requires exponential size read-once branching programs. They consider the *Exact-Perfect-Matching* function taking the value 1 if and only if the input graph consists of a perfect matching. They prove that this function requires $2^{\Omega(\sqrt{n})}$ size read-once branching programs, where n is the number of variables.

In this paper we consider a simple monotone function defined by the blocking sets of a finite projective plane and prove that it requires exponential size read-once branching programs. Our lower bound is based on elementary properties of projective planes.

2 Preliminaries

A *branching program* is a directed acyclic graph with a source node called START, and two sinks called ACCEPT and REJECT. Every vertex that is not a sink has outdegree 2, and the two edges leaving a given vertex are labeled by complementary literals $x_i, \bar{x}_i$ for some variable x_i $(1 \leq i \leq n)$. For every input string $(x_1, \ldots, x_n)$, $x_i \in \{0, 1\}$ the label of each edge evaluates to 0 or 1 depending on the value of the corresponding variable. A given input string is accepted by the program if and only if there is a directed path from START to ACCEPT along which all edge-labels take value 1 under this input. The branching program is said to compute the Boolean function

$f : \{0,1\}^n \to \{0,1\}$ which takes the value 1 precisely on the strings accepted by the branching program. The *size* of a branching program is the number of its nodes. A branching program is *read-once* if every variable occurs at most once along each source-sink path.

Let f be a Boolean function on the set of variables X. Consider a partition of X into two parts Y and $B = X \setminus Y$. For every fixed assignment σ of truth values to the variables in Y we get a subfunction f_σ on the remaining variables. Let $N(f, Y)$ denote the number of different subfunctions we obtain under all possible assignments to Y. Note that both $2^{|Y|}$ and $2^{2^{|B|}}$ are upper bounds for $N(f, Y)$.

Some variant of the following observation has been used in most papers on this subject. It is a special case of general results in [8, 11, 16] and is in fact implicit already in the method of Wegener [18] and Dunne [5].

Lemma([18, 5, 8, 11, 16]) Let f be a Boolean function of n variables. Assume that m is an integer, $1 \leq m \leq n$, such that for any m-element subset Y of the variables $N(f, Y) = 2^m$ holds. Then the size of any read-once branching program computing f is at least $2^m - 1$.

For completeness we include a proof.

Proof: We will show that the first $m - 1$ levels of any read-once branching program computing f must form a complete binary tree.

Let Z be an arbitrary set of at most $m-1$ variables, and let σ be a truth assignment to the variables in Z. Then the subfunction f_σ must depend on each of the remaining $n - |Z|$ variables, since otherwise Z could be extended to a set Y of m variables with $N(f, Y) < 2^m$. This also means that any path leading to a sink (ACCEPT or REJECT) in a branching program computing f must have length at least m.

Let v be an arbitrary node of a read-once branching program computing f. Suppose that there is a path P of length at most $m - 1$ leading from START to v. Let Y_P be the set of variables queried along the path P. We show that for any other path P' leading to v, $Y_{P'} = Y_P$ must hold. First we observe that the subprogram starting at v must depend on all the variables outside Y_P. This means that $Y_{P'} \subseteq Y_P$ since the branching program is read-once. But then we have $|Y_{P'}| \leq m - 1$, and by the same reasoning it follows that $Y_P \subseteq Y_{P'}$.

Let Z be an arbitrary set of at most $m-1$ variables. Suppose that there is a variable $z \in Z$ such that two different paths leading to the same vertex

and querying exactly the variables in Z evaluate z differently. Then we could find two different assignments σ_1 and σ_2 to the variables in Z such that the subfunctions f_{σ_1} and f_{σ_2} are the same, which is not possible. □

3 The construction

We use finite projective planes to exhibit an AC^0 computable function that requires exponential size read-once branching programs.

Let $\Pi = (P, L)$ be a projective plane of order q. (P is the set of points and L is the set of lines, viewed as subsets of P.) Let $n = q^2 + q + 1$ and $m = q + 1$. So $|P| = |L| = n$, each line has m points, and each point is incident with m lines.

We assign a variable x_i to each point $i \in P$, and define the following Boolean function.

Definition

$$f_\Pi(x_1, \ldots, x_n) = \bigwedge_{\lambda \in L} \bigvee_{i \in \lambda} x_i \,.$$

The value of this function is 1 if and only if the input, viewed as a subset of P, is a blocking set of the projective plane Π, that is it contains at least one point from each line of Π.

Theorem The size of any read-once branching program computing f_Π is at least $2^{\Omega(\sqrt{n})}$.

The proof is based on the following straightforward combinatorial property of projective planes.

Fact Let $J = \{p_1, \ldots, p_t\}$ be a set of $t \leq m$ distinct points of Π. Then there exist distinct lines $\lambda_1, \ldots, \lambda_t$ such that for $1 \leq i, j \leq t$ we have $p_i \in \lambda_j$ if and only if $i = j$.

Proof: Recall that there are exactly m lines that contain any given point. Let us consider an arbitrary point $p_i \in J$, and the m lines that contain it. Since any two lines intersect in at most one point, each of the other $t - 1 \leq m - 1$ points of the set J belong to at most one of these lines. Thus at least one of the m lines containing p_i will contain no other point from the set J. □

Proof of the Theorem: We show that for every q-element subset A of the variables, $N(f_\Pi, A) = 2^q$ holds, i.e. each truth assignment to the variables in

A yields a different subfunction on the remaining variables. Since each line λ defines a clause $\bigvee_{i\in\lambda} x_i$ of the function f_Π, it follows from the Fact that for an arbitrary q-element subset A of the variables there exist q clauses such that each variable from A appears in exactly one of them, and each variable appears in a different clause. Assume w.l.o.g. that $A = \{x_1, \ldots, x_q\}$, and the corresponding clauses are $\lambda_1, \ldots, \lambda_q$.

Let σ_1 and σ_2 be different truth assignments to the variables in A. Suppose they differ in the value of x_i, i.e., $x_i = 0$ in σ_1 and $x_i = 1$ in σ_2. Let us consider the assignment ξ to the variables outside A that sets each variable in the clause λ_i containing x_i to 0 and sets all other variables outside A to 1. Since any other clause λ_j has only one variable in common with the clause λ_i and there are at most $q-1$ variables in A that do not appear in λ_i, λ_j must contain at least one point which is neither in A nor in the clause λ_i thus it is set to 1 by ξ. Then we have $f_{\sigma_1}(\xi) = 0$ and $f_{\sigma_2}(\xi) = 1$. This shows that different truth assignments to the variables in A yield different subfunctions of f_Π.

The bound then follows from the Lemma. □

4 Remarks

The branching programs we considered in this note are called *deterministic* branching programs. There are several generalizations of this model that allow to use constants as labels on the edges of the program, or allow out-degree larger than 2 and let different variables to appear as labels on edges leaving from the same vertex. We refer to these more general models as *nondeterministic* branching programs. For precise definitions see [4, 15].

We note that for deterministic read-once branching branching programs it is the same whether we require each variable to appear at most once along each source-sink path (*syntactic* read-once branching programs) or we have the read-once requirement only on consistent paths, that is on paths that do not have both a variable and its negation present as labels (*nonsyntactic* read-once branching programs). However, the above two restrictions are not equivalent anymore in some of the nondeterministic read-once branching program models, as shown by Jukna [9].

In this note we have shown that the function f_Π requires exponential size deterministic read-once branching programs. It is easy to see that the com-

plement of the function f_Π can be computed by small (linear size) nondeterministic read-once branching programs. Jukna and Razborov [10] considered a modification of the function f_Π and showed that it requires exponential size deterministic read-once branching programs, but both the modified function and its complement can be computed by small nondeterministic read-once branching programs. It remains open to understand what is the nondeterministic read-once branching program complexity of the function f_Π.

Acknowledgements

I would like to thank László Babai and János Simon for helpful discussions. I would also like to thank Stasys Jukna, Stephen Ponzio and the anonymous referees for their comments and corrections of the manuscript.

References

[1] M. Ajtai, $\sum_1^1$-*formulae on finite structures*, Annals of Pure and Applied logic 24, (1983) pp. 1-48.

[2] M. Ajtai, L. Babai, P. Hajnal, J. Komlós, P. Pudlak, V. Rödl, E. Szemerédi and G. Turán *Two lower bounds for branching programs*, In Proceedings of the 18th ACM STOC, 1986, pp. 30-38.

[3] L. Babai, P. Hajnal, E. Szemerédi and G. Turán *A lower bound for read-once branching programs,* JCSS v. 35 n.2, Oct. 1987 pp. 153-162.

[4] A. Borodin, A. Razborov and R. Smolensky *On lower bounds for read-k-times branching programs,* Computational Complexity, 3, 1993, pp. 1-18.

[5] P. E. Dunne *Lower bounds on the complexity of 1-time only branching programs,* in FCT 85, Lecture Notes in CS n. 199, Springer, New York, 1985, pp. 90-99.

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

[7] J. HASTAD. *Almost optimal lower bounds for small depth circuits.* In Advances in Computing Research, 5: Randomness and Computation, Micali, S., ed. JAI Press, 1989, pp. 143-170.

[8] S. JUKNA, *Entropy of contact circuits and lower bound on their complexity,* Theor. Comput. Sci., 57 (1988), pp. 113-129.

[9] S. JUKNA, *A note on read-k-times branching programs*, RAIRO Theoretical Informatics and Applications, vol. 29, Nr. 1 (1995), pp. 75-83.

[10] S. JUKNA AND A. RAZBOROV, *A note on read-once branching programs for blocking sets in projective planes,* Manuscript, November 1996.

[11] M. KRAUSE, *Exponential lower bounds on the complexity of real time and local branching programs,* J. Inform. Proc. Cybern. (EIK), 24:3 1988, pp. 99-110

[12] M. KRAUSE, C. MEINEL AND S. WAACK, *Separating the eraser Turing machine classes L_e, NL_e, $co-NL_e$ and P_e*, Theor. Comput. Sci., 86 (1991), pp. 267-275.

[13] S. PONZIO, *A lower bound for integer multiplication with read-once branching programs,* In Proceedings of the 27th STOC, 1995, pp. 130-139.

[14] P. PUDLAK AND S. ZAK, *Space complexity of computations*, Tech. Rep. Univ. Prague, 1983.

[15] A. RAZBOROV, *Lower bounds for deterministic and nondeterministic branching programs,* In Proceedings of the 8th FCT, Lecture Notes in Computer Science, 529, 1991, pp. 47-60.

[16] J. SIMON AND M. SZEGEDY, *A new lower bound theorem for read-only-once branching programs and its applications*, In "Advances in Computational Complexity Theory", DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 13, 1993, pp. 183-193.

[17] P. SAVICZKY AND S. ZAK, *A large lower bound for 1-branching programs,* ECCC Technical Report, TR96-036.

[18] I. WEGENER, *On the complexity of branching programs and decision trees for clique functions,* Journal of the ACM, 35, 1988, pp. 461-471.

[19] I. WEGENER, *The Complexity of Boolean Functions*, Wiley-Teubner 1987.

[20] A. Yao, *Separating the polynomial hierarchy by oracles* In Proceedings of the 26th FOCS, 1985, pp. 1-10.

[21] S. Zak, *An exponential lower bound for one time only branching programs,* Proc. of MFCS'84, Springer Lect. Notes in Comp. Sci. 176, 1984, pp. 562-566.