Circuit Complexity and Multiplicative Complexity of Boolean Functions *

Arist Kojevnikov[1] and Alexander S. Kulikov[1]

St. Petersburg Department of Steklov Institute of Mathematics
{arist,kulikov}@logic.pdmi.ras.ru

Abstract. In this note, we use lower bounds on Boolean multiplicative complexity to prove lower bounds on Boolean circuit complexity. We give a very simple proof of a $7n/3 - c$ lower bound on the circuit complexity of a large class of functions representable by high degree polynomials over GF(2). The key idea of the proof is a circuit complexity measure assigning different weights to XOR and AND gates.

1 Introduction

Proving lower bounds on the circuit complexity of explicitly defined Boolean functions is one of the most famous and difficult problems in Theoretical Computer Science. Already in 1949 Shannon [1] showed by a counting argument that almost all Boolean functions have circuits of size $\Omega(2^n/n)$ only. Still, we have no example of an explicit function requiring super linear circuit size. Moreover, only a few proofs of linear lower bounds are known. Namely, Schnorr [2] proved a $2n-c$ lower bound for a class of functions with the property that by fixing the values of any two variables one gets at least three different subfunctions. Then Paul [3] proved a $2.5n - c$ lower bound for a modification of the storage access function. Stockmeyer [4] obtained the same lower bound for a class of symmetric functions satisfying a certain simple property. Finally, Blum slightly modified Paul's function and proved a $3n - o(n)$ bound on it. This bound was published in 1984 and is still the best result for circuits over the full binary basis B_2. The current record lower bound $5n - o(n)$ for the basis $U_2 = B_2 \setminus \{\oplus, \equiv\}$ was given in 2002 by Iwama and Morizumi [5].

All bounds mentioned above are proved by the gate elimination method. The main idea of this method is the following. One considers a Boolean function on n variables from a certain class of functions and shows (usually by a long case analysis) that for any circuit computing this function setting some variables to constants one obtains a subfunction of the same type and eliminates several gates. Usually, a gate is eliminated just because one of its inputs becomes a

* Research is partially supported by Federal Target Programme "Scientific and scientific-pedagogical personnel of the innovative Russia" 2009–2013, RFBR (08-01-00640 and 09-01-12137), RAS Program for Fundamental Research, Grant of the President of Russian Federation (MK-3912.2009.1 and NSh-5282.2010.1), and ANR, France (NAFIT ANR-08-EMER-008-01).

constant. By induction, one concludes that the original circuit must have many gates. Though this method is essentially the only known method for proving nontrivial lower bounds for general circuit complexity, as many authors note it is unlikely that it will allow to prove nonlinear bounds.

The multiplicative complexity of a Boolean function f is defined as the minimal number of AND gates in a circuit over $\{\wedge, \oplus, 1\}$ computing f. Again, by a counting argument one can show that almost all Boolean functions of n variables have multiplicative complexity $2^{n/2} - O(n)$ [6]. As for the constructive lower bounds, the situation is even worse than with circuit complexity. Namely, the best known lower bound is $n - 1$ [7]. This bound holds for any function representable by a polynomial over GF(2) of degree n. It is quite easy to see that, e.g., the conjunction of n variables has multiplicative complexity $n - 1$. Even a function with multiplicative complexity at least n is not known.

In this note, we prove a lower bound $7n/3 - c$ on the circuit complexity of a large class of functions representable by high degree polynomials over GF(2). The key idea of the proof is a circuit complexity measure assigning different weights to XOR and AND gates. Note that while the proven lower bound is weaker than Stockmeyer's $2.5n - c$ bound and Blum's $3n - o(n)$ bound, it is applicable for a much wider class of functions (Stockmeyer's proof works for symmetric functions only, Blum's bound works for a particular single function) and its proof is much simpler (in particular, contains almost no case analysis).

2 General Setting

2.1 Circuit Complexity and Gate Elimination

By B_n we denote the set of all Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$. A circuit over a basis $\Omega \subseteq B_2$ is a directed acyclic graph with nodes of in-degree 0 or 2. Nodes of in-degree 0 are marked by variables from $\{x_1, \ldots, x_n\}$ and are called inputs. Nodes of in-degree 2 are marked by functions from Ω and are called gates. There is also a special output gate where the result is computed. The size of a circuit is its number of gates. By $C_\Omega(f)$ we denote the minimum size of a circuit over Ω computing f. The two commonly studied bases are B_2 and $U_2 = B_2 \setminus \{\oplus, \equiv\}$. In this note we consider circuits over B_2 and denote $C_{B_2}(f)$ by just $C(f)$.

We call a function $f \in B^n$ degenerated if it does not depend essentially on some of its variables, i.e., there is a variable x_i such that the subfunctions $f|_{x_i=0}$ and $f|_{x_i=1}$ are equal. It is easy to see that a gate computing a degenerated function from B_2 can be easily eliminated from a circuit without increasing its size (when eliminating this gate one may need to change the functions computed at its successors). The set B_2 contains the following sixteen functions $f(x, y)$:

- six degenerate functions: 0, 1, x, $x \oplus 1$, y, $y \oplus 1$;
- eight functions of the form $((x \oplus a) \wedge (y \oplus b)) \oplus c$, where $a, b, c \in \{0, 1\}$; we call them type-$\wedge$ functions;

– two functions of the form $x \oplus y \oplus a$, where $a \in \{0, 1\}$; these are called type-$\oplus$ functions;

The main difference between type-$\wedge$ and type-$\oplus$ functions is that it is always possible to make a type-$\wedge$ binary function constant by assigning a Boolean value to any of its inputs (e.g., after replacing x by a one gets the constant c; in this case we say that a gate is trivialized), while this is not possible for a type-$\oplus$ function. Thus, if there is a variable feeding two type-$\oplus$ gates, then by assigning any value to it one eliminates only these two gates. However, if at least one of the two successors of a variable is of type-$\wedge$, then by assigning an appropriate value to it one eliminates both these gates and also all successors of this type-$\wedge$ gate. This is the reason why currently best lower bounds for circuits over U_2 are stronger than those for circuits over B_2.

Fig. 1 shows an example on how assigning values to input variables affects a circuit. We first replace x_1 by 1. Then G_5 computes $G_3 \oplus 1$ and is not needed anymore, while G_6 computes $G_3 \oplus G_4 \oplus 1$, i.e., $G_3 \equiv G_4$. We now assign the value 0 to x_3. Then both G_1 and G_2 can be eliminated as they compute x_2.

2.2 Multiplicative Complexity and Polynomials over GF(2)

The multiplicative complexity of a Boolean function f is defined as the minimal number of AND gates in a circuit over $\{\wedge, \oplus, 1\}$ computing f. It is easy to see that the minimal possible number of type-$\wedge$ gates in a circuit over B_2 computing f is exactly the multiplicative complexity of f.

Any function $f \in B_n$ can be easily represented as a multilinear polynomial over GF(2), i.e., a XOR (sum) of conjunctions (monomials). To obtain this representation one can take a sum of all "literal monomials" (i.e., conjunctions of literals, where a literal is either a variable or its negation) corresponding to elements of $f^{-1}(1)$. E.g., the polynomial over GF(2) for the majority function of three bits is

$$x_1x_2x_3 + (1 + x_1)x_2x_3 + x_1(1 + x_2)x_3 + x_1x_2(1 + x_3) = x_1x_2 + x_2x_3 + x_1x_3\,.$$

It is well known that such a representation is unique. Indeed, each function has a representation by a polynomial and the number of functions of n variables is equal to the number of multilinear polynomials over GF(2) (and is equal to 2^{2^n}). We denote this polynomial by $\chi(f)$. The important characteristics of a function f is the degree of $\chi(f)$ denoted by $\deg(f)$. Clearly, if a circuit contains only a few type-$\wedge$ gates, then it cannot compute a function of high degree. For example, to compute the conjunction of n variables $n - 1$ type-$\wedge$ gates are necessary and sufficient.

Lemma 1 ([7]). *Any circuit computing a Boolean function $f \in B_n$ contains at least $(\deg(f) - 1)$ type-$\wedge$ gates.*

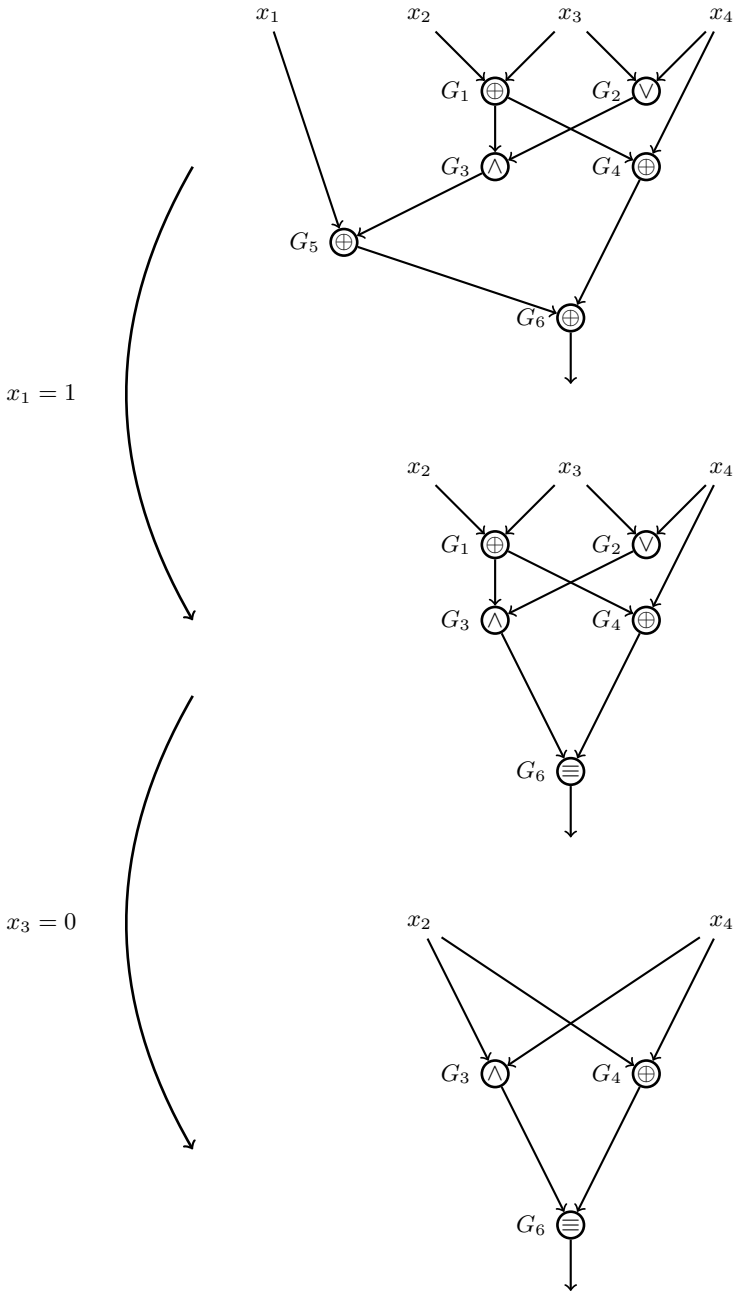

Fig. 1. Example of gate elimination.

3 $7n/3$ Lower Bound

In this section, we give a proof of a $7n/3 - c$ lower bound. It is as simple as Schnorr's proof of a $2n - c$ lower bound [2]. We first define a class of functions S_n^k similar to the one used by Schnorr. We then define a circuit complexity measure by putting different weights to type-$\oplus$ and type-$\wedge$ gates and prove a lower bound for it. This lower bound is then used to prove a lower bound on the circuit complexity of a class of functions obtained by taking the sum of functions from S_n^k and a function of full degree.

Definition 1. *For a constant k, let S_n^k be the class of all functions $f \in B_n$, $n \geq k$, with the following properties:*

1. *for any two variables x_i and x_j one obtains at least three different subfunctions of f by fixing the values of x_i and x_j;*
2. *for any variable x_i and any constant $c \in \{0,1\}$, $f|_{x_i=c} \in S_{n-1}^k$.*

A natural function satisfying both these properties is a modular function $\mathrm{MOD}_{m,r}^n$ defined as follows:

$$\mathrm{MOD}_{m,r}^n(x_1,\ldots,x_n) = 1 \text{ iff } \sum_{i=1}^{n} x_i \equiv r \pmod{m}\,.$$

It is easy to see that, for any $m \geq 3$ and r, $\mathrm{MOD}_{m,r}^n \in S_n^{m+2}$. Indeed,

$$\mathrm{MOD}_{m,r}^n|_{x_i=0,x_j=0} = \mathrm{MOD}_{m,r}^{n-2},\ \mathrm{MOD}_{m,r}^n|_{x_i=1,x_j=1} = \mathrm{MOD}_{m,r-2}^{n-2}\,,$$
$$\mathrm{MOD}_{m,r}^n|_{x_i=0,x_j=1} = \mathrm{MOD}_{m,r}^n|_{x_i=1,x_j=0} = \mathrm{MOD}_{m,r-1}^{n-2}\,.$$

For $m \geq 3$ and $n \geq m+2$, $\mathrm{MOD}_{m,r}^{n-2}$, $\mathrm{MOD}_{m,r-1}^{n-2}$, $\mathrm{MOD}_{m,r-2}^{n-2}$ are not constant and pairwise different (note that for $m = 2$ this is not true as $\mathrm{MOD}_{2,r-2}^{n-2} = \mathrm{MOD}_{2,r}^{n-2}$). Also, $\mathrm{MOD}_{m,r}^n|_{x_i=c} = \mathrm{MOD}_{m,r-c}^{n-1}$.

In order to prove the stated lower bound we use the following circuit complexity measure: $\mu(D) = 3X(D) + 2A(D)$, where $X(D)$ and $A(D)$ denote, respectively, the number of type-$\oplus$ and type-$\wedge$ gates in a circuit D.

Lemma 2. *For any circuit D computing a function $f \in S_n^k$,*

$$\mu(D) \geq 6(n-k-1)\,.$$

Proof. We prove the statement by induction on n. The case $n \leq k+1$ is obvious. Assume that $n > k+1$ and let D be an optimal (w.r.t. μ) circuit computing f. We show that it is possible to assign a value to one of the variables such that μ is reduced by at least 6. Since the resulting subfunction belongs to S_{n-1}^k, the required inequality follows by induction. Note that the resulting subfunction is not a constant (otherwise it would not have three different subfunctions w.r.t. any two variables). Thus, if a gate is replaced by a constant during such a

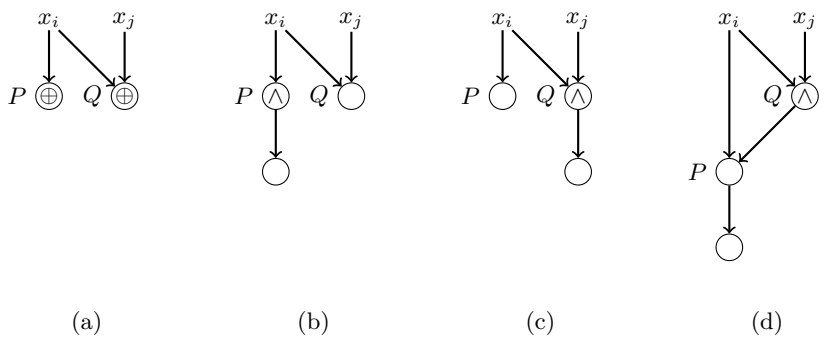

Fig. 2. Different cases of Lemma 2.

substitution, then this gate is not an output and hence has at least one successor that is also eliminated.

All degenerate gates can be eliminated from D without increasing $\mu(D)$. Let Q be a top-gate of D and x_i and x_j be its input variables. These variables are different as Q is non-degenerate. Since there are at least three different subfunctions of f w.r.t. x_i and x_j, one of them must feed at least one other gate. W.l.o.g. we assume that x_i feeds also a gate $P \neq Q$. There exist two cases.

1. If both P and Q are type-$\oplus$ gates (see Fig. 2(a); note that only the types of gates are shown, but not the exact functions computed at them), we just assign a value to x_i. Clearly μ is reduced by at least 6.
2. If one of P and Q is a type-$\wedge$ gate (Fig. 2(b–d)), we assign to x_i the value which trivializes this gate. This eliminates both P and Q and at least one of their successors. Note that P can be a successor of Q (Fig. 2(d)). In this case after assigning x_i the right constant, not only Q becomes a constant, but also the gate P, as both its inputs are constants. Hence all successors of P are eliminated. Since at least three gates are eliminated, μ is again reduced by at least 6. □

Lemma 3. *Let* $f \in S_n^k$ *and* $\deg(f) = n$. *Then* $C(f) \geq 7n/3 - c(k)$.

Proof. Let D be a circuit computing f. Recall that the number of gates in a circuit D is at least $X(D) + A(D)$. The required inequality then follows from the following two inequalities:

$$3X(D) + 2A(D) \geq 6n - 6(k+1)\,,$$
$$A(D) \geq n - 1\,.$$

□

The lemma above provides a $7n/3 - c$ lower bound for functions from S_k^n of degree n. In fact, it can be used to prove the same lower bound for a much

wider class of functions. Assume that $f \in S_n^k$ and $\deg(f) < n$. Consider now *any* function $g \in B_{n-1}$ of degree $n-1$ (note that the number of such functions is $2^{2^{n-1}-1}$, as we only need that the monomial $x_1 x_2 \dots x_{n-1}$ is present in $\chi(g)$) and define $h \in B_n$ as follows:

$$h(x_1, \dots, x_n) = f(x_1, \dots, x_n) \oplus x_n g(x_1, \dots, x_{n-1}) .$$

Consider a circuit D computing h. Since $\deg(h) = n$, $A(D) \geq n-1$. Note also that $h|_{x_n=0} = f|_{x_n=0} \in S_{n-1}^k$. Hence $\mu(D) \geq \mu(D|_{x_n=0}) \geq 6(n-1) - 6(k+1)$ and a lower bound $7n/3 - c(k)$ for the size of D follows.

4 Further Directions

It would be interesting to give an example of an explicit function with multiplicative complexity at least an, for $a > 1$. Such a function would immediately give a stronger than $7n/3$ lower bound for circuit complexity. As said above, the best known lower bound $n-1$ follows from the fact that a function is represented by a polynomial of high degree over GF(2). Such is, e.g., the function $\mathrm{MOD}_{3,r}^n$: $\deg(\mathrm{MOD}_{3,r}^n) \geq n-1$, for any r and $n \geq 3$. Thus, at least $n-2$ type-$\wedge$ gates are needed just to compute a high degree monomial in $\chi(\mathrm{MOD}_{3,r}^n)$. It is natural to expect that to compute the $\mathrm{MOD}_{3,r}^n$ function more type-$\wedge$ gates are needed in the worst case. Interestingly, this is not true: Boyar et al. [6] showed that the multiplicative complexity of any symmetric function is at most $n + o(n)$.

As to the lower bounds on circuit complexity, it would be interesting to improve lower bounds $3n - o(n)$ by Blum [8] and $5n - o(n)$ by Iwama and Morizumi [5] on the circuit complexity over B_2 and U_2, respectively. An (apparently) easier problem is to close one of the following gaps (see [4], [9], [10]):

$$2.5n - c \leq C_{B_2}(\mathrm{MOD}_3^n) \leq 3n + c,\ 4n - c \leq C_{U_2}(\mathrm{MOD}_4^n) \leq 5n + c .$$

Acknowledgments

We would like to thank Edward A. Hirsch for helpful comments.

References

1. Shannon, C.E.: The synthesis of two-terminal switching circuits. Bell System Technical Journal **28** (1949) 59–98
2. Schnorr, C.: Zwei lineare untere Schranken für die Komplexität Boolescher Funktionen. Computing **13** (1974) 155–171
3. Paul, W.J.: A $2.5n$-lower bound on the combinational complexity of Boolean functions. SIAM Journal of Computing **6**(3) (1977) 427–433
4. Stockmeyer, L.J.: On the combinational complexity of certain symmetric Boolean functions. Mathematical Systems Theory **10** (1977) 323–336

5. Iwama, K., Morizumi, H.: An explicit lower bound of $5n-o(n)$ for boolean circuits. In: Proceedings of 27th International Symposium on Mathematical Foundations of Computer Science (MFCS). Lecture Notes in Computer Science, Springer (2002) 353–364
6. Boyar, J., Peralta, R., Pochuev, D.: On The Multiplicative Complexity of Boolean Functions over the Basis $(\wedge, \oplus, 1)$. Theoretical Computer Science **235**(1) (2000) 1–16
7. Schnorr, C.P.: The Multiplicative Complexity of Boolean Functions. In: Proceedings of Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. Volume 357 of Lecture Notes in Computer Science., Springer (1989) 45–58
8. Blum, N.: A Boolean function requiring $3n$ network size. Theoretical Computer Science (28) (1984) 337–345
9. Demenkov, E., Kojevnikov, A., Kulikov, A.S., Yaroslavtsev, G.: New upper bounds on the Boolean circuit complexity of symmetric functions. Information Processing Letters **110**(7) (2010) 264–267
10. Zwick, U.: A $4n$ lower bound on the combinational complexity of certain symmetric boolean functions over the basis of unate dyadic Boolean functions. SIAM Journal on Computing **20** (1991) 499–505