Lower Bounds for Boolean Circuits of Bounded Negation Width[☆]

Stasys Jukna[a,1], Andrzej Lingas[b,2,*]

[a]*Faculty of Mathematics and Computer Science, Vilnius University, Vilnius, Lithuania.*
[b]*Department of Computer Science, Lund University, Lund, Sweden*

Abstract

The negation width of a Boolean AND, OR, NOT circuit computing a *monotone* Boolean function f measures the "amount of negation" used by the circuit, and is the minimum number w such that the unique formal DNF produced (purely syntactically) by the circuit contains each prime implicant of f extended by at most w solely negated variables. The negation width of monotone circuits is zero. We first show that already a moderate allowed negation width can substantially decrease the size of monotone circuits. Our main result is a general reduction from non-monotone circuits of bounded negation width to monotone circuits: if a monotone Boolean function f can be computed by a non-monotone circuit of negation width w, then f can be also computed by a monotone circuit of size s times $4\min\{w^m, m^w\}\log M$, where m is the maximum length of a prime implicant and M is the total number of prime implicants of f.

1. Introduction

Understanding the power of negations in computations is one of the most basic tasks in computational complexity. While strong, even exponential, lower bounds for explicit monotone Boolean functions are already known for *monotone* Boolean $\{\lor, \land\}$ circuits, we can currently prove only depressingly small (linear) lower bounds on the size of $\{\lor, \land, \neg\}$ circuits when there are no restrictions on the number or the usage of negation gates. So, it is important to understand the power of negations in Boolean circuits.

It is well known and easy to show that requiring all NOT gates to appear only at input variables is not a real restriction. Circuits fulfilling this restriction are known as *DeMorgan circuits*. Such a circuit uses fanin-2 OR and AND gates, while inputs are variables $x_1, \dots, x_n$ and their negations $\overline{x}_1, \dots, \overline{x}_n$; to simplify notation, we will sometimes write $\overline{x}_i$ instead of $\neg x_i$. By just doubling the circuit size and using DeMorgan laws, any circuit over $\{\lor, \land, \neg\}$ of size s can be converted to a DeMorgan circuit computing the same function and having size at most $2s$ (see, for example, [7, Theorem 3.1]). A *formula* is a circuit with all gates of fanout 1 (the underlying graph is a tree). A *monotone circuit* is a DeMorgan circuit without negated input variables.

The effect of negations on the size or depth of $\{\lor, \land, \neg\}$ circuits was mainly considered by either restricting the total *number* of used negation gates, or by restricting the *usage* of negated input variables in DeMorgan circuits.

[☆]Preliminary version appeared in [17].
[*]Corresponding author
Email addresses: stjukna@gmail.com (Stasys Jukna), Andrzej.Lingas@cs.lth.se (Andrzej Lingas)
[1]Research supported by the DFG grant JU 3105/1-1 (German Research Foundation).
[2]Research supported in part by VR grant 2017-03750 (Swedish Research Council).

There is an extensive literature on the research in the first direction (when the total number of NOT gates is bounded); here negations can be applied not only to input variables. Markov [23] has shown that every Boolean function f of n variables (even any multi-output Boolean function) can be computed by a circuit using only $\log n$ NOT gates. Fischer [8] has shown that one can always reduce the number of NOT gates to the Markov $\log n$ bound with only a slight increase in circuit size. Morizumi [24, 25, 26] proved versions of Markov's theorem for Boolean formulas, as well as nondeterministic and probabilistic circuits. Amano and Maruoka [4] have shown that $(1/6)\log\log n$ NOT gates are not enough to compute the $(n/2)$-clique function by a polynomial size circuit. In [15], a (multi-output) monotone Boolean function is given which can be computed by poly-size circuit with $\log n$ NOT gates, but requires an exponential size if only $\log n - \mathcal{O}(\log\log n)$ NOT gates are available; this is by only an additive $\log\log n$ factor away from the Markov–Fischer bound. We refer to [16, Chapter 10] and the papers cited therein for this line of research; see also [33, 5, 11, 10] for more recent developments in this direction.

Another line of research (which we follow in this paper) was to restrict the "amount of negation" in DeMorgan circuits; here negations are only applied to input variables. One of the first results in this direction was proved by Raz and Wigderson [29, Theorem 4.1]: if $w \leq n^{2-\epsilon}$ for a constant $\epsilon > 0$, then any DeMorgan circuit with at most w negated input variables computing the s-t connectivity function of n-vertex graphs must have depth $\Omega(\log^2 n)$. Guo et al. [11] have proved that any DeMorgan circuit with at most w negated input variables computing a monotone Boolean function f must have depth at least the monotone circuit depth of f minus w. Koroth and Sarma [21] relax this restriction on the total number of allowed negated input variables, and say that a not necessarily DeMorgan $\{\vee, \wedge, \neg\}$ circuit has *orientation weight* w if the function computed at each gate is monotone in all but at most w variables. Informally, this means that the function computed at any gate of the circuit must be computable by a DeMorgan circuit using at most w negated input variables. They prove that the depth of a circuit of orientation weight w computing a monotone function f is at least the minimum depth of a monotone circuit computing f divided by $4w + 1$.

In this paper, as the measure of the "amount of negation" in DeMorgan circuits, we consider their "negation width" (see Definition 1 below). This measure, without giving it a name, was introduced by Amano and Maruoka [3, Section 4]. They used a modification of Razborov's Method of Approximation [31, 32] to show that DeMorgan circuits of small negation width for the Clique function must still be large; we recall their result more exactly in Section 7.

Our main results (Theorems 1 and 2) give a *general* and fairly simple reduction of DeMorgan circuits of bounded negation width to monotone circuits, from which the bound of [3], as well as new lower bounds, follow (see Section 7).

Notation. We use standard terminology regarding Boolean functions (see, for example, [38]). In particular, a *term* is an AND of *literals*, each being a variable or its negation. The *length* of a term is the number of distinct literals in it. A term is a *zero term* if it contains a variable and its negation. A *DNF* (disjunctive normal form) is an OR of terms. An *implicant* of a Boolean function $f(x_1, \dots, x_n)$ is a nonzero term p such that $p \leq f$ holds, that is, $p(a) \leq f(a)$ holds for all $a \in \{0,1\}^n$. An implicant p of f is a *prime implicant* of f if no proper subterm q of p has this property, that is, if $p \leq q \leq f$, then $q = p$. The set of all prime implicants of f will be denoted by $PI(f)$. A Boolean function f is *monotone* if $a \leq b$ implies $f(a) \leq f(b)$. Note that if f is monotone, then all prime implicants of f are positive, that is, consist solely of not negated variables.

1.1. DeMorgan circuits and their associated DNFs

Our goal is to understand to what extent the usage of negated input variables can decrease the size or the depth of DeMorgan circuits computing *monotone* Boolean functions. As a measure of the "amount of negation" in DeMorgan circuits, we will use their "negation width." This measure is motivated by a simple observation that every DeMorgan circuit F not only computes a particular Boolean function f but also *produces* (purely syntactically) a unique set $T(F)$ of terms in a natural way:

- if $F = z$ is an input literal, then $T(F) = \{z\}$;
- if $F = F_1 \lor F_2$, then $T(F) = T(F_1) \cup T(F_2)$;
- if $F = F_1 \land F_2$, then $T(F) = \{t_1 \land t_2 \colon t_i \in T(F_i), i = 1, 2\}$.

During the production of terms, we use the "shortening" axiom $x \land x = x$, but do not use the "annihilation" axiom $x \land \overline{x} = 0$. So, $T(F)$ can contain *zero terms*, that is, terms with a variable x_i and its negation $\overline{x}_i$.[3] Easy induction on the circuit size shows that the Boolean function f computed by a circuit F is the function computed as the OR of all produced terms, that is, $f = \bigvee_{t \in T(F)} t$.

For example, the DeMorgan formula $F = xy \lor (x \lor y)(\overline{x} \lor \overline{y})z$ computes the threshold-2 function $f(x, y, z) = xy \lor xz \lor yz$ of three variables, and the set of terms produced by the circuit F is $T(F) = \{xy, x\overline{y}z, y\overline{x}z, x\overline{x}z, y\overline{y}z\}$, where the latter two terms are zero terms.

If a circuit F computing a monotone Boolean function f is *monotone* (has no negated inputs at all), then we clearly have $PI(f) \subseteq T(F)$, that is, every prime implicant of f must then be produced by the circuit. But even then, the equality $T(F) = PI(f)$ does not need to hold: already in 1981, Okolnishnikova [27] exhibited an explicit monotone Boolean function f of n variables which can be computed by a monotone circuit of size $\mathcal{O}(n)$, but any monotone circuit F satisfying $T(F) = PI(f)$ must have $2^{\Omega(n^{1/4})}$ gates.

The situation when the function f computed by a DeMorgan circuit F is monotone, but the circuit F itself is *not* necessarily monotone, is even more subtle: then even the inclusion $PI(f) \subseteq T(F)$ does not need to hold. For example, the circuit $F = x\overline{y} \lor y$ computes the function $f = x \lor y$ and produces the set $T(F) = \{x\overline{y}, y\}$, whereas $PI(f) = \{x, y\}$. Super-polynomial lower bounds on the size of non-monotone DeMorgan circuits F when $PI(f) \not\subseteq T(F)$ but $|T(F) \cap PI(f)| \geq |PI(f)|^{\delta}$ for a constant $\delta > 0$ were proved in [13] by extending Razborov's method of approximations.

1.2. *Negation width of circuits*

In the present paper, we consider the situation when $T(F) \cap PI(f)$ may be even empty, that is, when even none of the prime implicants is produced by the circuit at all. Our restriction is motivated by the following well-known "folklore" observation; see, for example, the book [6, p. 37]. The *positive factor* of a term $t = \bigwedge_{i \in S} x_i \land \bigwedge_{j \in T} \overline{x}_j$ is the AND $t_+ = \bigwedge_{i \in S} x_i$ of all its unnegated variables. The term t is *nonzero* if $S \cap T = \emptyset$ holds.

Fact 1 (Folklore). *If the Boolean function f computed by a DeMorgan circuit F is monotone, then every prime implicant of f is the positive factor of at least one nonzero term produced by F.*

That is, for every prime implicant p of f, either p itself or some its nonzero extension by solely negated variables must be produced by the circuit.

[3] At a "functional" level, zero terms are redundant: they contribute nothing to the values of the computed function. The only reason to keep them in $T(F)$ is to ensure that "syntactical" changes of circuits (replacements of some input gates by constants), which we will latter make, do not turn some previously zero terms into nonzero terms.

Proof. Since the circuit F computes f, we have $f = \bigvee_{t \in T(F)} t$. Fix a prime implicant $p = \bigwedge_{i \in A} x_i$ of f. Take an input vector $a \in \{0,1\}^n$ with $a_i = 1$ for $i \in A$, and $a_i = 0$ for $i \notin A$. On this vector, we have $p(a) = 1$ and, hence, also $f(a) = 1$. So, $t(a) = 1$ must hold for some nonzero term $t = \bigwedge_{i \in S} x_i \wedge \bigwedge_{i \in T} \overline{x}_i$ in $T(F)$. This yields the inclusion $S \subseteq A$. Let $t_+ = \bigwedge_{i \in S} x_i$ be the positive factor of the term t. Since the function f is monotone, and t is an implicant of f, t_+ of t is also an implicant of f. Since $t_+(a) = 1$, and since p is a *prime* implicant, $S = A$ follows. □

If the circuits are only allowed to use at most w negated input variables, say $\overline{x}_1, \ldots, \overline{x}_w$ (the restriction considered in [29, 11]), then the extensions $t = p\overline{x}_{i_1} \cdots \overline{x}_{i_l}$ of *all* prime implicants p can only use these negated inputs. We now relax this and allow *different* subsets of up to w negated inputs out of all $\overline{x}_1, \ldots, \overline{x}_n$ to be used for *different* prime implicants p. That is, we only require that for every prime implicant p of f, the circuit produces some its nonzero extension $t = p\overline{x}_{i_1} \cdots \overline{x}_{i_l}$ with $l \leq w$. For different prime implicants p, the produced extensions t may have different negated variables.

Definition 1 (Negation width). The *negation width* of a DeMorgan circuit F computing a monotone Boolean function f is the minimum number w such that for every prime implicant p of f, the circuit produces either p or some its nonzero extension by at most w negated variables.

That is, the circuit F has negation width w if for every prime implicant $p = \bigwedge_{i \in S} x_i$ of f the circuit produces (at the output gate) some term $t = \bigwedge_{i \in S} x_i \wedge \bigwedge_{j \in T} \overline{x}_j$ with $T \cap S = \emptyset$ and $|T| \leq w$.

There are no other restrictions on the remaining produced terms, except the trivial one that the function computed as the OR of all produced terms must coincide with the function f. In particular, the circuit is allowed to produce terms with much more than w negated variables, as well as arbitrary zero terms. Important only is that every prime implicant of f has *at least one* extension with at most w negated variables.

Note that the negation width w of any DeMorgan circuit computing f satisfies $0 \leq w \leq n - m$, where n is the total number of variables, and m is the minimum length of a prime implicant of f. Also, minimal circuits of negation width $w = 0$ are essentially monotone circuits: just replace each negated input gate $\overline{x}_i$ by constant 0.

As we already mentioned, the negation width (without using this term) was already considered by Amano and Maruoka [3, Sect. 4]. Examples of sufficient (but by far not necessary) conditions for a circuit to have negation width at most w are any of the following.

- The circuit has at most w negated input variables; such circuits were considered, for example, by Raz and Wigderson [29], and Guo et al.[11].
- No input-output path has more than[4] $\log w$ AND gates; such circuits computing quadratic forms (multi-output functions) were considered in [22].
- No nonzero term produced by the circuit contains more than w distinct negated variables. Note that this restriction is a relaxation of both two previous restrictions.

None of these sufficient conditions is necessary. In particular, the negation width restricts neither the total number of negated input variables nor the length of produced *zero* terms. Also, at intermediate gates, the circuit can use any number of negated input variables, can produce very long terms, and then cancellate them, that is, turn them into zero terms: such terms do not affect the negation width.

[4]All logarithm in this paper are to the base 2.

Remark 1. At this point, it is worth to mention that DeMorgan circuits computing monotone Boolean functions f *more efficiently* than monotone circuits *must* use cancellations (must produce zero terms): otherwise, we could just replace all negated inputs by constants 1, and the resulting monotone circuit would still compute f.

1.3. Motivation

Our motivation to consider circuits of bounded negation width w is that allowance of even moderate negation width $w = n^{\epsilon}$ for an arbitrarily small constant $\epsilon > 0$ *can* reduce the size of monotone circuits (those of negation width $w = 0$). We demonstrate this by three examples. The first two examples also show that our restriction on the negation width is *properly* weaker than the restriction on the total number of allowed negated input variables, as considered in [29, 11]. The third example shows that the reduction in size can be even super-polynomial.

Example 1 (Threshold functions). The threshold-k function Th_k^n accepts a Boolean input of length n iff it contains at least k ones. The smallest known monotone circuits for Th_k^n have size of order $n \log k$ (such circuits are constructed in [20] using the AKS sorting network [1]). On the other hand, for $k \leq n^{1/3}$, the function Th_k^n *can* be computed by a DeMorgan circuit of *linear* size $\mathcal{O}(n)$ if the negation width $w = k^3$ is allowed (see Lemma 6 in Appendix B).

Now, consider a DeMorgan circuit computing Th_k^n and using at most $w = (1-\epsilon)n$ negated input variables, for a constant $\epsilon > 0$. Assign the value 0 to all variables whose negations are used by the circuit. The resulting *monotone* circuit computes the threshold-k function for inputs of length $n - w \geq \epsilon n$ and, hence, most likely has size $\Omega(n \log k)$. Thus, apparently, DeMorgan circuits for Th_k^n with almost the maximal number $w = (1-\epsilon)n$ of allowed negated input gates cannot lead to substantial speed ups.

Example 2 (Triangle function). The *triangle function* $\mathrm{Clique}(n, 3)$ has $\binom{n}{2}$ variables, one for each edge of the complete graph K_n on $\{1, \ldots, n\}$, and accepts a subgraph G of K_n iff G contains a triangle. It is known that this function requires monotone circuits of almost cubic size $n^{3-o(1)}$ [31, 2]. In Appendix C we show that, if the negation width $w = n^{\epsilon}$ is allowed, then the triangle function can already be computed using a *sub-cubic* number $\mathcal{O}(n^{3-\epsilon/4})$ of gates. For example, if the negation width $w = \sqrt{n}$ is allowed, then only about $n^{2.875}$ gates are enough.

Now, consider a DeMorgan circuit F computing $\mathrm{Clique}(n, 3)$ and using at most $w = n^2/7$ negated variables. Again set to 0 all the variables whose negations are used by the circuit. The resulting monotone circuit computes some function $f \leq \mathrm{Clique}(n, 3)$ with at least $M = \binom{n}{3} - nw = \Omega(n^3)$ prime implicants (triangles). The argument of Alon and Boppana [2, Lemma 3.14] implies that every monotone circuit for any such function f must use at least about $M/\log^3 n = \Omega(n^{3-o(1)})$ gates. Hence, the circuit F must also use at least so many gates. This shows that even if about the maximum number $w = n^2/7$ of negated inputs is allowed, no DeMorgan circuit can compute $\mathrm{Clique}(n, 3)$ in sub-cubic size.

Example 3 (Logical permanent). The *logical permanent* function Per_m is a monotone Boolean function of m^2 variables which takes a Boolean $m \times m$ matrix Y as input, and outputs 1 iff Y contains m 1-entries no two of which lie in the same row or the same column. Let $0 < \epsilon < 1/2$ be an arbitrarily small constant, and assume for simplicity that both $m = n^{\epsilon}$ and $r = n^{1-\epsilon}$ are integers. Consider the monotone Boolean function $f(X)$ whose variables are arranged into an $n \times n$ matrix X. Split X into r^2 disjoint $m \times m$ submatrices. The function f accepts X iff $\mathrm{Per}_m(Y) = 1$ holds for at least one of these submatrices Y. The monotone circuit complexity of f is at least the monotone circuit complexity of Per_m which, as shown by Razborov [32], is $m^{\Omega(\log m)} = n^{\Omega(\log n)}$.

On the other hand, it is well known that Per_m can be computed by a DeMorgan circuit of size polynomial in m; see, for example, Hopcroft and Karp [12]. The negation width of such a circuit is clearly at most the number m^2 of its input variables. So, since at OR gates the negation width is not increased, we obtain a DeMorgan circuit for f of size $r^2 \cdot m^{\mathcal{O}(1)} = n^{\mathcal{O}(1)}$ and negation width $w \leq m^2 = n^{2\epsilon}$.

Using the monotone circuit lower bound of Tardos [36] instead of the logical permanent, one can show that, on some explicit monotone Boolean functions, an even *exponential* (not only super-polynomial) gap between the size of monotone circuits and circuits of moderate negation width can be achieved. We are not aware of any separating examples for restrictions on the use of negations considered in [29, 21, 11]: restricted number of allowed negated input variables (as in [29, 11]), or restricted orientation weight (as in [21]).

Thus, allowance of nonzero negation width *can* decrease the size of monotone circuits. A natural question therefore is: by how much the monotone circuit size can be decreased? In this paper, we give an answer: the size of monotone circuits *cannot* be decreased by a factor larger than $\Delta(m, w) = 4 \cdot \min\{w^m, m^w\} \cdot \log M$, where w is the allowed negation width, m is the maximal length of a prime implicant of the computed monotone Boolean function f, and M is the total number of prime implicants of f.

2. Our results

Our lower bounds will depend on two parameters: $m \geq 3$ is the maximum length of a prime implicant of a given monotone Boolean function f, and $w \geq 1$ is the maximum allowed negation width of DeMorgan circuits computing f. Having these two parameters, we define

$$\Delta(m, w) := 4 \cdot \min\{w^m, m^w\}\,.$$

Easy induction shows that $r^s \geq s^r$ holds for all $3 \leq r \leq s$ (see Claim 1 in Section 3). Hence, $\Delta(m, w) = 4w^m$ if $w \geq m$, and $\Delta(m, w) = 4m^w$ if $w \leq m$.

The first result reduces (non-monotone) DeMorgan circuits of bounded negation width to *monotone* circuits. A *monotone subcircuit* of a DeMorgan circuit is obtained by fixing the values of some variables to constant 0 (so that the corresponding negated input gates of F are replaced with constant 1), and replacing the remaining negated input gates of F with constant 0.

Theorem 1. *Let F be a DeMorgan circuit of negation width w computing a monotone Boolean function f, and let m be the maximum length of a prime implicant of f. Then an OR of at most $\tau = \Delta(m, w) \cdot \log |PI(f)|$ monotone subcircuits of F also computes f.*

Our proof is probabilistic. Given a DeMorgan circuit F computing a monotone Boolean function, we will define its random monotone subcircuit $\boldsymbol{F}_+$ by randomly replacing some of input gates by constants 0 and 1. When doing this, we will carefully chose the probability distribution in terms of the parameters m and w; see Section 3 for precise definition of $\boldsymbol{F}_+$. We then show that every single prime implicant $p \in PI(f)$ is produced by the circuit $\boldsymbol{F}_+$ with probability at least $\Delta(m, w)^{-1} = \frac{1}{\tau} \log |PI(f)|$. Thus, the probability that the OR of τ independent copies of $\boldsymbol{F}_+$ will produce *all* prime implicants $p \in PI(f)$ is nonzero.

Remark 2. Note that we only consider subcircuits of DeMorgan circuits of a very special form: we just replace some of the input literals by constants 0 and 1 (the structure of circuits remains untouched). So, the same simulation holds also for other circuit models as DeMorgan formulas, contact schemes, switching-and-rectifier networks as well as for DeMorgan circuits with unbounded fanin AND and OR gates.

Theorem 1 directly yields the following lower bounds on the size and depth of DeMorgan circuits of bounded negation width. For a monotone Boolean function f, let:

$C_w(f)$ = minimum size of a DeMorgan circuit of negation width w computing f;

$C_+(f)$ = minimum size of a monotone circuit computing f.

In the case of DeMorgan *formulas*, these measures are denoted by $L_w(f)$ and $L_+(f)$; in this case, the *size* of a formula is the number of leaves of the underlying tree. Let also $D_w(f)$ denote the minimum *depth* of a DeMorgan circuit of negation width w computing f, and let $D_+(f)$ denote the minimum depth of a monotone circuit computing f.

Corollary 1. *Let $w \geq 1$ and let f be a monotone Boolean function with all prime implicants of length at most m. Then*

$$C_w(f) \geq \frac{C_+(f)}{\tau} - 1\,, \quad L_w(f) \geq \frac{L_+(f)}{\tau} \quad \textit{and} \quad D_w(f) \geq D_+(f) - \log \tau\,,$$

where $\tau = \Delta(m, w) \cdot \log |PI(f)|$.

Remark 3 (Negation width and communication complexity). In Appendix D, we show that a lower bound $D_w(f) \geq D_+(f) - w \cdot \lceil \log(n+1) \rceil$ on the depth of DeMorgan circuits of negation width w can be *directly* proved using a communication complexity argument. This argument exposes the meaning of the negation width restriction from the communication point of view.

Remark 4 (AND-depth). The *AND-depth* of a DeMorgan circuit is the maximum number of AND gates along an input-output path. Since no circuit of AND-depth d can produce any terms with more than 2^d literals, the negation width w of any such circuit satisfies $w \leq 2^d$. So, Corollary 1 gives the same lower bound $C_+(f)/\tau - 1$ on the size of any DeMorgan circuit of AND-depth at most $\log w$.

Our second result concerns circuits of bounded *average* negation width. Let F be a DeMorgan circuit computing a monotone Boolean function f. The *negation width* of a prime implicant $p \in PI(f)$ in the circuit F is the minimum number w such that $T(F)$ contains a nonzero extension of p by at most w negated variables. Hence, the circuit F has negation width at most w if *every* prime implicant of f has negation width at most w in F. Average negation width relaxes this "every" requirement.

Definition 2 (Average negation width). The *average negation width* of the circuit F is the average, over all prime implicants $p \in PI(f)$, of the negation width of p in F.

Hence, if $w(f)$ denotes the negation width of a prime implicant $p \in PI(f)$ in the circuit F, then the average negation width of the circuit F is

$$w = \frac{1}{|PI(f)|} \sum_{p \in PI(f)} w(p)\,.$$

Definition 3. A monotone Boolean function h *τ-approximates* a monotone Boolean function f if there is an OR g of at least $|PI(f)|/\tau$ prime implicants of f such that $g \leq h \leq f$ holds.

Theorem 2. *Let f be a monotone Boolean function with all prime implicants of length at most m. Let $w \geq 1$ and $\tau = 2 \cdot \Delta(m, 2w)$. If every monotone circuit τ-approximating f requires at least s gates, then every DeMorgan circuit of average negation width w computing f must also have at least s gates.*

Remark 5. Note the difference between Corollary 1 and Theorem 2. To apply Corollary 1, one can *directly* use known lower bounds on the monotone circuit complexity of the function f *itself.* Theorem 2 is more general: it applies to circuits when only the *average* negation width is bounded, and we do not have the additional $\log |PI(f)|$ factor in the "blow down" parameter τ. However, in order to apply Theorem 2, one has to show that not only the function f itself but also any sufficiently close approximation of f requires large monotone circuits. So, one has to analyze the monotone lower bound *proofs* to ensure this latter property; we demonstrate this in the derivation of Corollary 6 in Section 7.

It is well-known that DeMorgan formulas can be balanced: every DeMorgan formula of size s can be simulated by a DeMorgan formula of depth at most $c \log s$ for some constant $c \geq 1$. This was first proved by Spira [34] with $c < 3.42$. Subsequently, the balance constant c was incrementally decreased by many authors; the best known $c \leq 1.73$ is due to Khrapchenko [19].

In our context (when the negation width of formulas is bounded), the following natural question arises: can also DeMorgan formulas of bounded negation width be balanced *without* increasing the negation width of the resulting (balanced) formulas? The question is nontrivial because Spira's argument, as well as subsequent ones introduce negation gates applied to sub-formulas (not just to input variables), which may result in a much larger negation width. Our third result answers the question in the affirmative.

Theorem 3. *For every monotone Boolean function f, and for every $w \geq 0$, we have $D_w(f) \leq 3 \log L_w(f)$.*

That is, if a monotone Boolean function f can be computed by a DeMorgan formula of size s and negation width w, then f can be also computed by a DeMorgan formula of depth at most $3 \log s$ and the same negation width w. For formulas of negation width $w = 0$ (i.e., for monotone formulas), this was shown by Wegener [37].

Organization. In Section 3, a special type of random subcircuits is introduced. Sections 4, 5 and 6 are devoted to the proofs of our main results (Theorems 1, 2 and 3); the proofs are fairly simple. In Section 7, we give some applications of these results to specific Boolean functions. Appendices include some surrounding results. Appendix A gives a general argument to decrease the negation width in non-monotone circuits. Appendix B illustrates this reduction for circuits computing threshold functions. Appendix C shows that already the circuits of moderate negation width for the triangle function are of sub-cubic size. Appendix D gives an alternative proof of a lower bound on the depth of circuits of bounded negation width using communication complexity arguments.

3. Random subcircuits

Let $f(x_1, \ldots, x_n)$ be a monotone Boolean function, and F be a DeMorgan circuit of negation width w computing f. The inputs of F are the variables $x_1, \ldots, x_n$ and their negations $\overline{x}_1, \ldots, \overline{x}_n$; the rest consists of AND and OR gates. Starting from the circuit F, our goal is to construct a *monotone* circuit computing f. Since the circuit F has negation width w, we know that for every prime implicant p of f the circuit must produce some nonzero extension $p\overline{x}_{i_1} \cdots \overline{x}_{i_l}$ of p by $l \leq w$ solely negated variables.

If the circuit F produces no zero terms (those containing a variable x_i and its negation $\overline{x}_i$), then we can just replace all negated input gates $\overline{x}_i$ by constants 1, and the resulting monotone circuit F_+ will also compute f. But, in general (when the circuit produces zero terms), we cannot do this: if, say, F produces some zero term $x_i\overline{x}_i q$, where the term $x_i q$ is nonzero but is *not* an implicant of f, then the resulting monotone circuit will (wrongly) accept some inputs which are rejected by f.

So, to cope with zero terms, we will use the idea suggested in [22]: instead of replacing a negated input gate $\overline{x}_i$ of the circuit F by constant 1 give the value 0 to the variable x_i itself. Note that then no zero term $t = x_j\overline{x}_jq$ produced by the circuit F can turn into a nonzero term: if $j = i$ then t turns into the constant 0, and if $j \neq i$, then the resulting term $x_j\overline{x}_jq'$ is still a zero term (or constant 0).

More formally, given a DeMorgan circuit $F(x_1, \dots, x_n, \overline{x}_1, \dots, \overline{x}_n)$ and a subset $I \subseteq [n] := \{1, \dots, x_n\}$, the *monotone subcircuit* F_+ of F (or *monotone I-subcircuit* to mention the used subset I) is obtained as follows.

1. First, set to 0 the values of all variables x_i with $i \in I$. That is, for every $i \in I$, the unnegated input gate x_i is replaced with constant 0, while the negated input gate $\overline{x}_i$ is replaced with constant 1.
2. Then replace with constant 0 each of the remaining negated input gates $\overline{x}_j$ for $j \notin I$.

Finally, eliminate constant input gates through repeated replacements of $0 \wedge u$ by 0, $1 \vee u$ by 1, and $0 \vee u$, $1 \wedge u$ by u. Schematically:

$$F(x, y, \overline{x}, \overline{y}) \overset{\text{Step 1}}{\mapsto} F(x, 0, \overline{x}, 1) \overset{\text{Step 2}}{\mapsto} F(x, 0, 0, 1) \mapsto F_+(x)\,.$$

Example 4. Consider the DeMorgan formula $F = (x_1 \vee x_2 \vee \overline{x}_3)(\overline{x}_1 \vee \overline{x}_2 \vee x_5)(x_3 \vee x_4 \vee \overline{x}_5)$, and the set $I = \{1, 4\}$. After the first step, we obtain the formula $(0 \vee x_2 \vee \overline{x}_3)(1 \vee \overline{x}_2 \vee x_5)(x_3 \vee 0 \vee \overline{x}_5)$. After the second step, we obtain the formula $(0 \vee x_2 \vee 0)(1 \vee 0 \vee x_5)(x_3 \vee 0 \vee 0)$ and, after the elimination of constants, the resulting monotone sub-formula of F is $F_+ = x_2x_3$.

Lemma 1. *Let F be a DeMorgan circuit computing a monotone Boolean function f, and g be the monotone Boolean function computed by a monotone subcircuit of F. Then $PI(g) \subseteq PI(f)$ and, consequently, $g \leq f$.*

Proof. Let F_+ be a monotone I-subcircuit of F for some $I \subseteq [n]$, and let g be the monotone Boolean function computed by F_+. Since the circuit F computes f, the set $T(F)$ of terms produced by F has the following two properties: every prime implicant of f is a positive part of at least one nonzero term in $T(F)$, and every nonzero term in $T(F)$ contains at least one prime implicant of f as a subterm.

At the 1st step of the construction of F_+, all variables x_i with $i \in I$ are evaluated to the constant 0 and, hence, all negated input gates $\overline{x}_i$ of F with $i \in I$ are replaced by constant 1. After this evaluation, all terms of $T(F)$ containing at least one unnegated literal x_i with $i \in I$ disappear from $T(F)$ (are evaluated to 0), and the negated literal $\overline{x}_i$ with $i \in I$ are removed from all remaining terms (these literals are evaluated to 1). At the 2nd step, all remaining terms with at least one negated variable (including the remaining zero terms) are removed. Thus, every term in $T(F_+)$ is either a prime implicant or contains some prime implicant of f as a subterm. In particular, $PI(g) \subseteq PI(f)$ holds. □

When going from a given DeMorgan circuit F computing a given monotone Boolean function f to its monotone I-subcircuit F_+ we set to 0 all variables x_i with $i \in I$. So, all prime implicants of f containing at least one of these variables are lost, are no longer produced by F_+. Our goal is to lose as few prime implicants as possible.

To achieve this, we will set to 0 *random* subsets of variables. Let ρ be a random restriction which randomly and independently sets each variable x_i to 0 with probability $0 < \epsilon < 1$, and leaves x_i unset with probability $1 - \epsilon$. Let $t = \bigwedge_{i \in S} x_i \wedge \bigwedge_{j \in T} \overline{x}_j$ be a term with $|S| \leq m$ unnegated and $|T| \leq w$ negated variables, and assume that t is a nonzero term, i.e., that $S \cap T = \emptyset$ holds. The *positive factor* of t is the term $t_+ = \bigwedge_{i \in S} x_i$. As observed in [22, Lemma 2], the (random) subterm $t\lceil_\rho$ of t resulting after the restriction ρ satisfies

$$\text{Prob}\left\{t\lceil_\rho = t_+\right\} \geq \alpha(\epsilon) := \epsilon^w(1 - \epsilon)^m\,. \tag{1}$$

Indeed, the probability that none of the $|S| \leq m$ unnegated variables of t is evaluated to 0 is $(1-\epsilon)^{|S|} \geq (1-\epsilon)^m$, while the probability that all $|T| \leq w$ negated variables of t are evaluated to 1 is $\epsilon^{|T|} \geq \epsilon^w$.

For fixed m and w, the maximum value of the function $\alpha(\epsilon) = \epsilon^w(1-\epsilon)^m$ is achieved at the point

$$\epsilon_0 = \frac{w}{m+w} = 1 - \frac{m}{m+w}.$$

Indeed, the first derivative of the function $g(\epsilon) = \ln \alpha(\epsilon) = w \ln \epsilon + m \ln(1-\epsilon)$ is $g'(\epsilon) = -w/\epsilon + m/(1-\epsilon)$, and this derivative is zero iff $\epsilon = w/(m+w)$. So, the maximal possible "survival" probability is

$$\alpha(\epsilon_0) = \left(1 - \frac{m}{m+w}\right)^w \left(1 - \frac{w}{m+w}\right)^m.$$

By a more careful choice of the probability ϵ, depending on whether there are more negated variables in the extensions of prime implicants (w is larger than m) or not, a more handy lower bound on the "survival" probability $\alpha(\epsilon) = \epsilon^w(1-\epsilon)^m$ can be obtained.

Namely, define the *random (m,w)-subcircuit* of a DeMorgan circuit F to be a monotone $\boldsymbol{I}$-subcircuit of F for the random subset $\boldsymbol{I} \subseteq [n]$ with each $i \in [n]$ included in $\boldsymbol{I}$ independently with probability:

$$\epsilon := \begin{cases} 1 - \frac{1}{w} & \text{if } w \geq m, \\ \frac{1}{m} & \text{if } w < m. \end{cases} \tag{2}$$

Thus, if a we have more negated variables than unnegated in a term ($w \geq m$), then we evaluate to 0 each variable with a slightly larger then optimal probability (our goal is to evaluate all negated literals to 1), and if we have more unnegated variables ($m > w$), then we evaluate to 0 each variable with a slightly smaller then optimal probability (our goal is to keep all these variables untouched).

In the following lemma, $m \geq 3$ and $w \geq 1$ are integers, and F is an arbitrary DeMorgan circuit of negation width w. Recall that $\Delta(m,w) := 4 \cdot \min\{w^m, m^w\}$.

Lemma 2. *Let $p \cdot r$ be a nonzero term, where p consists of $\leq m$ unnegated variables, and r consists of $\leq w$ negated variables. Let $\boldsymbol{F}_+$ be a random (m,w)-subcircuit of F. If $p \cdot r \in T(F)$, then*

$$\text{Prob}\,\{p \in T(\boldsymbol{F}_+)\} \geq 1/\Delta(m,w).$$

That is, if the term $p \cdot r$ is produced by the circuit F, then its positive factor p is produced by the subcircuit $\boldsymbol{F}_+$ with probability at least $1/\Delta(m,w)$.

Proof. By Eq. (1), we know that $\text{Prob}\,\{p \in T(\boldsymbol{F}_+)\} \geq \alpha := \epsilon^w(1-\epsilon)^m$ holds for any $0 < \epsilon < 1$ and, hence also for ϵ defined by Eq. (2). To show that $\alpha \geq 1/\Delta(m,w)$ holds (for this particular choice of ϵ), we use the following elementary estimates.

Claim 1. *For all integers $t \geq 2$ and $3 \leq r \leq s$, we have $(1-1/t)^t \geq 1/4$ and $r^s \geq s^r$.*

Proof. The first inequality follows from the fact that the sequence $a_t = (1-1/t)^t$ for $t = 2, 3, \ldots$ is non-decreasing. Namely, a_{t+1}/a_t is $t/(t+1)$ times

$$\left(\frac{t^2}{t^2-1}\right)^t = \left(1 + \frac{1}{t^2-1}\right)^t \geq \left(1 + \frac{1}{t^2}\right)^t \geq 1 + t \cdot \frac{1}{t^2} = \frac{t+1}{t},$$

where the second inequality follows from the Bernoulli inequality. So, $a_{t+1}/a_t \geq 1$, implying that $(1-1/t)^t \geq a_2 = 1/4$ holds for all integers $t \geq 2$.

To see the second inequality, observe that $r^s \geq s^r$ is equivalent to $r^{1/r} \geq s^{1/s}$. We claim that the sequence $r^{1/r}$ for $r = 3, 4, \ldots$ is non-increasing. This can be also shown by induction on r. Observe that the inequality $r^{1/r} \geq (r+1)^{1/(r+1)}$ is equivalent to $r^{r+1} \geq (r+1)^r$, or $r \geq (1+1/r)^r$. Since $(1+1/r)^r$ is at most the Euler number $\mathrm{e} < 3$, and since we assumed that $r \geq 3$, the inequality follows. □

We can now finish the proof of Lemma 2 as follows. If $w \geq m$, then $\epsilon = 1 - 1/w$, and Claim 1 yields

$$\alpha = (1-1/w)^w (1/w)^m \geq \tfrac{1}{4} w^{-m} \geq \tfrac{1}{4} m^{-w}\,.$$

If $3 \leq w < m$, then $\epsilon = 1/m$, and Claim 1 yields

$$\alpha = (1/m)^w (1-1/m)^m \geq \tfrac{1}{4} m^{-w} \geq \tfrac{1}{4} w^{-m}\,.$$

In both cases, we have $\alpha \geq \frac{1}{4}\max\{w^{-m}, m^{-w}\} = 1/\Delta(m,w)$. It remains to show that $\alpha \geq 1/\Delta(m,w)$ holds also for $w = 1$ and for $w = 2$. Recall that the second parameter (the maximum length of a prime implicant of f) satisfies $m \geq 3$, and Claim 1 yields $(1-1/m)^m \geq 1/4$. So, if $w = 1$, then $\alpha = (1/m)^w(1-1/m)^m \geq 1/(4m) = 1/\Delta(m,1)$, and if $w = 2$, then $\alpha \geq 1/(4m^2) = 1/\Delta(m,2)$. □

4. Proof of Theorem 1: reduction to monotone circuits

Let f be a monotone Boolean function with all prime implicants of length at most m, and suppose that f can be computed by a DeMorgan circuit F of negation width w. Our goal is to show that then there exist at most $\tau = \Delta(m,w)\cdot\log|PI(f)|$ monotone subcircuits of F whose OR also computes f.

Let $\boldsymbol{F}_+$ be a random (m,w)-subcircuit of F, and take τ independent copies $\boldsymbol{F}_+^1, \ldots, \boldsymbol{F}_+^\tau$ of $\boldsymbol{F}_+$. By Lemma 2, we have $\mathrm{Prob}\,\{p \in T(\boldsymbol{F}_+)\} \geq 1/\kappa$ for every prime implicant $p \in PI(f)$, where $\kappa = \Delta(m,w)$. Note that $\tau/\kappa = \log|PI(f)|$. Hence, for every prime implicant $p \in PI(f)$, we have

$$\mathrm{Prob}\left\{p \notin T(\boldsymbol{F}_+^i) \text{ for all } i = 1, \ldots, \tau\right\} \leq (1-1/\kappa)^\tau \leq \mathrm{e}^{-\tau/\kappa} < 2^{-\tau/\kappa} = |PI(f)|^{-1}\,.$$

By the union bound, the probability that some prime implicant of f is produced by *none* of the circuits $\boldsymbol{F}_+^1, \ldots, \boldsymbol{F}_+^\tau$ is strictly smaller than 1. Consequently, there must be a sequence $F_+^1, \ldots, F_+^\tau$ of realizations of these circuits such that *every* prime implicant of f is produced by at least one of these circuits. Consider the monotone Boolean function $h = h_1 \vee \cdots \vee h_\tau$, where h_i is the function computed by F_+^i. By Lemma 1, we have $h \leq f$. On the other hand, the inclusion $PI(f) \subseteq T(F_+^1) \cup \cdots \cup T(F_+^\tau)$ yields the converse inequality $f \leq h$. So, the OR of the circuits $F_+^1, \ldots, F_+^\tau$ computes $h = f$, as desired. □

5. Proof of Theorem 2: average negation width

Let f be a monotone Boolean function with all prime implicants of length at most m. Let F be a DeMorgan circuit of average negation width w computing f. Recall that a monotone Boolean function h *τ-approximates* a monotone Boolean function f if there is an OR g of at least $|PI(f)|/\tau$ prime implicants of f such that $g \leq h \leq f$ holds. Now suppose that every monotone circuit τ-approximating f for $\tau = 2\cdot\Delta(m,2w)$ requires at least s gates. Our goal is to show that then the circuit F must have at least s gates, as well.

Since the average negation width of F is w, there must be some subset $P \subseteq PI(f)$ of $|P| \geq \frac{1}{2}|PI(f)|$ prime implicants of f such that every p has negation width at most $2w$ in F. Let $\boldsymbol{F}_+$ be a random $(m,2w)$-subcircuit of F. By Lemma 2, we have

$$\mathrm{Prob}\,\{p \in T(\boldsymbol{F}_+)\} \geq 1/\Delta(m,2w) = 2/\tau$$

for every prime implicant $p \in P$. So, the expected number of prime implicants $p \in P$ produced by $\boldsymbol{F}_+$ is at least $(2/\tau)|P| \geq |PI(f)|/\tau$.

There must therefore be a realization F_+ of $\boldsymbol{F}_+$ such that the set $P' = P \cap T(F_+)$ has $|P'| \geq |PI(f)|/\tau$ terms. Let g be the OR of the terms in P', and h be the monotone Boolean function computed by F_+. Since $P' \subseteq T(F_+)$, we have $g \leq h$, while the inequality $h \leq f$ follows from Lemma 1. This means that the circuit F_+ τ-approximates f and, by our assumption about the function f, the monotone circuit F_+ and, hence, also the original (non-monotone) circuit F must have at least s gates, as desired. □

6. Proof of Theorem 3: a version of Spira's theorem

As before, for a DeMorgan circuit or formula F, $T(F)$ denotes the set of terms produced by F. Two formulas are *equivalent* if they compute the same function. Recall that the size of a formula is the number of its leaves.

Lemma 3. *For every monotone formula F of size s, there is an equivalent monotone formula F' of depth at most $3\log s$ such that $T(F) \subseteq T(F')$.*

Here, only the inclusion $T(F) \subseteq T(F')$ is new: that monotone formulas can be balanced was already shown by Wegener [37]. We will need this additional inclusion in the case of monotone formulas to balance (non-monotone) DeMorgan formulas of bounded negation width (in the proof of Theorem 3 below).

Proof. We argue by induction on s. The claim is trivially true for $s = 2$ (just take $F' = F$). Now assume that the claim holds for all formulas with fewer than s leaves, and prove it for formulas with s leaves. Take an arbitrary monotone formula F with s leaves. By walking from the output-gate of F we can find a sub-formula H such that H has $\geq s/2$ leaves but its left and right sub-formulas each have $< s/2$ leaves. Now replace the sub-formula H of F by constants 0 and 1, and let F_0 and F_1 be the resulting formulas. It is clear that then the formula $(H \wedge F_1) \vee (\neg H \wedge F_0)$ is equivalent to F. The key observation (already made by and Wegener [37]) is that, due to the monotonicity of F, $F_0(x) = 1$ implies $F_1(x) = 1$. Thus the formula $(H \wedge F_1) \vee F_0$ is equivalent to F: if $F_0(x) = 1$ but $\neg H(x) = 0$, then $H(x) \wedge F_1(x) = 1$.

The formulas F_0 and F_1 as well as the left and right sub-formulas of H each have at most $s/2$ leaves. By the induction hypothesis, F_0 and F_1 can be replaced by formulas F_0' and F_1' of depth at most $3\log(s/2)$, and the formula H can be replaced by a formula H' of depth at most $1 + 3\log(s/2)$ (we take $+1$ because $3\log(s/2)$ is the upper bound on the depth of the two sub-formulas of H) such that

$$T(F_1) \subseteq T(F_1')\,, \quad T(F_0) \subseteq T(F_0') \quad \text{and} \quad T(H) \subseteq T(H')\,. \tag{3}$$

Thus, the resulting entire formula

$$F' = (H' \wedge F_1') \vee F_0' \tag{4}$$

is equivalent to F and has depth at most $2 + 1 + 3\log(s/2) = 3\log s$.

It remains to show that the set $T(F')$ of terms produced by the (balanced) formula F' satisfies $T(F') \supseteq T(F)$. Let F_z be the formula obtained from F by replacing the sub-formula H by a new variable z. Then the set of terms produced by F_z has the form $T(F_z) = (\{z\} * Q) \cup R$, where Q is some set of terms, R consists of all terms in $T(F_z)$ with no occurrences of the variable z, and $T_1 * T_2$ stands for the set of terms $\{t_1 \wedge t_2 \colon t_1 \in T_1, t_2 \in T_2\}$. That is, $\{z\} * Q$ is the set of all terms produced by F_z that contain the variable z, and R is the set of all terms produced by F_z that do not contain this variable. This yields

$$T(F) = [T(H) * Q] \cup R\,, \;\; T(F_1) = Q \cup R \;\text{ and }\; T(F_0) = R\,. \tag{5}$$

So,

$$\begin{aligned} T(F') &\overset{(4)}{=} [T(H') * T(F_1')] \cup T(F_0') \overset{(3)}{\supseteq} [T(H) * T(F_1)] \cup T(F_0) \\ &\overset{(5)}{=} [T(H) * (Q \cup R)] \cup R \supseteq [T(H) * Q] \cup R \overset{(5)}{=} T(F)\,, \end{aligned}$$

as desired. □

Proof of Theorem 3. Let f be a monotone Boolean function, and $w \geq 0$. Suppose that f can be computed by a DeMorgan formula $G = G(x, \overline{x})$ of size s and negation width w. Our goal is to show that then f can be also computed by a DeMorgan formula of depth at most $3 \log s$ and the same negation width w.

Replace all negated input variables $\overline{x}_i$ in G by new variables y_i, and consider the monotone formula $F = G(x, y)$. Since the formula G has negation width w, we know that the monotone formula F has the following property:

($*$) for every prime implicant $p = \bigwedge_{i \in S} x_i$ of f there is a term $p \cdot r \in T(F)$ with $r = \bigwedge_{j \in T} y_j$, $T \cap S = \emptyset$ and $|T| \leq w$.

Apply Lemma 3 to the monotone formula $F(x, y)$. This gives us an equivalent monotone formula $F'(x, y)$ of depth at most $3 \log s$ whose set $T(F')$ of produced terms contains all terms produced by the formula F. This latter property implies that the (balanced) formula F' also has property ($*$). So, if we replace back in $F'(x, y)$ the input variables y_i by negated variables $\overline{x}_i$, the obtained (also balanced) DeMorgan formula $F''(x, \overline{x})$ computes our function f and has negation width w, as desired. □

7. Explicit lower bounds

Below we demonstrate our general bounds given by Corollary 1 on the *clique functions.* Such a function $f = \mathrm{Clique}(n, k)$ has $\binom{n}{2}$ variables, one for each edge of the complete graph K_n on $[n] = \{1, \ldots, n\}$. Every assignment of Boolean values to these variables specifies a subgraph of K_n, and the function f accepts the assignment iff the specified graph contains a complete graph on k or more vertices; note that (to avoid floorings and ceilings) we do not require k to be an integer. Recall that $C_w(f)$ and $D_w(f)$ denote the minimum size and the minimum depth of a DeMorgan circuit of negation with $\leq w$ computing f. Corollary 1 gives the lower bounds

$$C_w(f) \geq \frac{C_+(f)}{\tau} - 1 \quad \text{and} \quad D_w(f) \geq D_+(f) - \log \tau\,, \tag{6}$$

for any $w \geq 1$ and any monotone Boolean function f with all prime implicants of length at most m, where

$$\tau := \Delta(m, w) \cdot \log |PI(f)| \quad \text{with} \quad \Delta(m, w) := \begin{cases} 4w^m & \text{if } w \geq m; \\ 4m^w & \text{if } w < m. \end{cases}\,. \tag{7}$$

7.1. Triangles

The *triangle function* $\mathrm{Clique}(n,3)$ accepts an input graph iff it contains a triangle. For this function, we can show an almost optimal lower bound in the class of circuits of bounded negation width.

Corollary 2. *Let $f = \mathrm{Clique}(n,3)$, and $w = n^{\epsilon}$ for $\epsilon > 0$. There are absolute constants $c_1, c_2 > 0$ such that, $n^{3-4\epsilon} \leq C_w(f) \leq c_2 n^{3-\epsilon/4}$.*

Proof. The upper bound is proved in Appendix C. To derive the lower bound, we use the lower bound $C_+(f) = \Omega\left(n^3/\log^3 n\right)$ on the monotone circuit complexity of f proved by Alon and Boppana [2, Lemma 3.14]. Monomials of f have length $m = 3 \leq w$, so $\Delta(m,w) = 4w^m = 4n^{3\epsilon}$. Since f has $|PI(f)| = \binom{n}{3} \leq n^3$ prime implicants, the parameter τ in Eq. (7) is at most a constant times $4w^m \cdot \log|PI(f)| \leq 12n^{3\epsilon}\log n$, and Eq. (6) yields the desired lower bound $C_w(f) \geq C_+(f)/\tau - 1 = \Omega(n^{3-4\epsilon})$. ◻

7.2. Large cliques

For clique functions $f = \mathrm{Clique}(n,k)$ with $k = n^{2/3}$, Amano and Maruoka [3, Theorem 4.2] proved a lower bound $C_w(f) = 2^{\Omega(n^{1/3})}$ as long as the allowed negation width is $w \leq \sqrt{k}/n^{\epsilon} = n^{1/3-\epsilon}$ for an arbitrarily small constant $\epsilon > 0$. They obtained this bound via appropriate extension of Razborov's method of approximations *itself* to circuit of bounded negation width.

On the other hand, when combined with our general reduction to monotone circuits, as given by Corollary 1, the same lower bound can be directly derived from known lower *bounds* of the monotone circuit complexity of clique functions. Namely, Alon and Boppana [2, Theorem 3.9] have shown that $C_+(f) = 2^{\Omega(\sqrt{k})}$ holds for any $3 \leq k \leq (n\log n)^{2/3}/4$. In particular, for the clique size $k = n^{2/3}$ considered in [3], we have $C_+(f) = 2^{\Omega(n^{1/3})}$. Since f has $|PI(f)| = \binom{n}{k} \leq n^k$ prime implicants, each of length $m = \binom{k}{2} \leq k^2$, the parameter τ in Eq. (7) is at most a constant times $km^w \log n \leq k^{2w+1}\log n$. Since $w\log k = \mathcal{O}(n^{1/3-\epsilon}\log n) = o(n^{1/3})$, we have $\tau \leq 2^{o(n^{1/3})}$, and the desired lower bound $C_w(f) \geq C_+(f)/\tau - 1 = 2^{\Omega(n^{1/3})}$ follows.

Note that, in the aforementioned result of [3], the allowed negation width w is *smaller* than the clique size k. When combined with the lower bound of Alon and Boppana [2, Theorem 3.16] for cliques of small (up to logarithmic) size, Corollary 1 directly yields super-polynomial lower bounds also when the allowed negation width is much larger, even *exponential*, in the clique size.

Corollary 3. *Let $f = \mathrm{Clique}(n,k)$ for $k = (\log n)^{1/3}$. Then $C_w(f) = n^{\Omega(k)}$ for $w = 2^k$.*

Proof. Let $f = \mathrm{Clique}(n,k)$. It is shown in [2, Theorem 3.16] that $C_+(f) \geq n^k/(8k^2 e^k \log n)^k$ holds for any $3 \leq k \leq \frac{1}{4}\log n$. In particular, for $k = (\log n)^{1/3}$, we have $C_+(f) = n^{\Omega(k)}$. On the other hand, since f has $|PI(f)| = \binom{n}{k} \leq n^k$ prime implicants, each of length $m = \binom{k}{2} \leq k^2$, the parameter τ in Eq. (7) is at most a constant times $w^m \cdot \log|PI(f)| \leq 2^{k^3} k\log n \leq n\log^2 n$, and Eq. (6) yields $C_w(f) \geq C_+(f)/\tau - 1 = n^{\Omega(k)}$. ◻

The following lower bound holds for the *depth* of DeMorgan circuits of bounded negation width.

Corollary 4. *Let $f = \mathrm{Clique}(n, n/2)$. Then $D_w(f) = \Omega(n)$ for $w = o(n/\log n)$.*

Proof. Raz and Wigderson [30, Corollary 4.1] have proved that $D_+(f) = \Omega(n)$. Since f has $|PI(f)| = \binom{n}{n/2} \leq 2^n$ prime implicants, each of length $m = \binom{n/2}{2} \leq n^2$, the logarithm of the parameter τ in Eq. (7) is at most a constant times $w\log m + \log\log|PI(f)| = \mathcal{O}(w\log n)$. Eq. (6) yields $D_w(f) \geq D_+(f) - \log\tau = D_+(f) - \mathcal{O}(w\log n) = \Omega(n)$, as desired. ◻

The following lower bound holds for the size of DeMorgan *formulas* of bounded negation width.

Corollary 5. *Let* $f = \mathrm{Clique}(n, n/2)$. *Then* $L_w(f) = 2^{\Omega(n)}$ *for* $w = o(n/\log n)$.

Proof. The desired lower bound follows directly from Corollary 4 and our refinement of Spira's depth-reduction given in Theorem 3. □

Remark 6. Recently, Pitassi and Robere [28] gave an explicit monotone Boolean function f of n variables such that $D_+(f) = \Omega(n)$. Together with the lower bound in Eq. (6), this implies that any (non-monotone) DeMorgan circuit of negation width $w = \epsilon n$ for a sufficiently small constant $\epsilon > 0$ must have linear depth $\Omega(n)$. Together with Theorem 3, this result implies a truly exponential lower bound $L_w(f) = 2^{\Omega(n)}$ on the size of DeMorgan *formulas* of negation width $w = \epsilon n$. Note that the ultimate goal is to prove lower bounds for DeMorgan circuits of negation width $w = n$ (or only $w = n - m$, where m is the minimum length of a prime implicant): these bounds then would hold for *unrestricted* circuits.

7.3. Average negation width

We will now give an application of our Theorem 2 concerning DeMorgan circuits of bounded *average* negation width. As we already mentioned in Section 2, in order to apply this theorem, we need lower bounds on the size of monotone circuits that only *approximate* a given monotone Boolean function (see Definition 3).

Fortunately, known lower bound arguments for monotone circuits work also when the monotone circuits are only required to produce a large enough subset of prime implicants (not necessarily *all* prime implicants). In particular, these arguments yield the following lower bound on the size of monotone circuits approximating clique functions. A *k-clique* is a subgraph of K_n consisting of a complete graph on some k vertices, and $n - k$ isolated vertices.

Lemma 4. *Let* $\tau \geq 1$, $3 \leq k \leq \sqrt{n}$, *and let* f *be a monotone Boolean function which rejects all graphs of chromatic number at most* $k - 1$, *and accepts a* $1/\tau$*-fraction of all* k*-cliques. Then* $C_+(f) \geq 2^{\Omega(\sqrt{k})}/\tau$.

Proof. Set $q := k - 1$. Every q-coloring $h : [n] \to [q]$ of the vertices of K_n defines the graph G_h whose edges are pairs of vertices receiving the *same* color. Different colorings may lead to the same graph, but we treat these graphs as distinct, for the simplicity of counting. Note that the chromatic number of the complement of every G_h does not exceed $q = k - 1$; so, the complements of graphs G_h must be rejected by f. An *l-forest* is a forest with l edges. We will need the following simple fact.

Claim 2. *For any integer* $1 \leq s < n$, *at most* q^{n-s} *of the graphs* G_h *can contain a fixed* s*-forest.*

Proof. Fix an s-forest F, and let $T_1, \ldots, T_t$ be all its connected components (trees). Hence, F touches $m = |F| + t = s + t$ vertices. All vertices in each of these trees must receive the same color. There are only q^t possibilities to assign colors to the trees, and only $q^{n-m} = q^{n-s-t}$ possibilities to color the vertices outside the forest F. So, the number of graphs G_h containing the forest F does not exceed $q^t q^{n-m} = q^{n-s}$. □

As shown in [14, Theorem 3.4], if f can be computed by a monotone circuit of size t, then for any integer parameters $1 \leq r, s \leq n - 1$ there exist a family of at most $t \cdot (2s)^{2r}$ r-cliques, a family of at most $t \cdot (2r)^{2s}$ s-forests, and a set E of at most r^2 edges such that at least one of the following two assertions holds:

(i) every k-clique accepted by f contains at least one of the given r-cliques;

(ii) for every q-coloring h, the graph G_h either intersects E or contains at least one of the given s-forests.

Now take a monotone circuit F_+ which accepts a $1/\tau$-fraction of all k-cliques, and rejects all graphs of chromatic number at most $k-1$.

Every r-clique is contained in $\binom{n-r}{k-r}$ k-cliques. So, under the first alternative (i), the size t of the circuit F_+ must be at least $\binom{n}{k}/\tau$ divided by $(2s)^{2r}\binom{n-r}{k-r}$. Since $\binom{n}{k}/\binom{n-r}{k-r} \geq (n/k)^r$, this is at least $(n/4ks^2)^r/\tau$.

On the other hand, the number of q-colorings h for which $G_h \cap E = \emptyset$ holds is at least $q^n - r^2 \cdot q^{n-1} = q^n(1-r^2/q)$, which is at least $q^n/2$ as long as $r \leq \sqrt{q}$. We have $t \cdot (2r)^{2s}$ s-forests and, by Claim 2, for at most q^{n-s} q-colorings h the graphs G_h can contain any fixed s-forest. So, under the second alternative (ii), the size t of the circuit F_+ must be at least $(q^s/2)/(2r)^{2s}q^{n-s} = \frac{1}{2}(q/4r^2)^s$ which, for any $r \leq \sqrt{(q-1)/2} = \sqrt{(k-2)/2}$ is at least $\frac{1}{2}(k/4r^2)^s$.

By taking the parameters $r := \lfloor\sqrt{k/16}\rfloor$ and $s := \lfloor\sqrt{n/8k}\rfloor$, the first alternative yields a lower bound $t \geq 2^r/\tau$, while the second one yields $t \geq \frac{1}{2}4^s \geq 2^s$. Since our assumption $k \leq \sqrt{n}$ yields $s \geq r$, the desired lower bound $t \geq 2^r/\tau \geq 2^{\Omega(\sqrt{k})}/\tau$ follows. □

Corollary 6. *Let $f = \mathrm{Clique}(n,k)$ for $3 \leq k \leq \sqrt{n}$. Then every DeMorgan circuit of average negation width $w = o(\sqrt{k}/\log k)$ computing f must have $2^{\Omega(\sqrt{k})}$ gates.*

Proof. Lemma 4 implies that, for every $\tau \geq 1$, every monotone circuit τ-approximating f requires at least $t = 2^{\Omega(\sqrt{k})}/\tau$ gates. The length of prime implicants of f is $m = \binom{k}{2}$. So, by taking $\tau := 8m^{2w} = 2^{o(\sqrt{k})}$, Theorem 2 yields the desired lower bound on the size of any DeMorgan circuit of average negation width w computing f. □

Appendix A. Reducing negation width via decompositions

An *m-decomposition* of a monotone Boolean function f is its representation in the form $f = \varphi(f_1,\ldots,f_l)$, where $f_1(X_1),\ldots,f_l(X_l)$ and $\varphi(y_1,\ldots,y_l)$ are monotone Boolean functions, and $|X_i| \leq m$ for all i. Such a decomposition is *semidisjoint* if the outer function φ fulfills the following condition:

$$\text{if both variables } y_i \text{ and } y_j \text{ belong to the same prime implicant of } \varphi\text{, then } X_i \cap X_j = \emptyset. \tag{A.1}$$

This condition ensures that the functions $f_i(X_i)$ substituted to the variables y_i of any prime implicant of $\varphi(y_1,\ldots,y_l)$ depend on *disjoint* sets of variables. We say that a DeMorgan circuit F simultaneously computing all Boolean functions $f_1(X_1),\ldots,f_l(X_l)$ is *local* if, for every i, no term produced at a gate of F computing the function $f_i(X_i)$ contains a literal x_j or $\overline{x}_j$ for $x_j \notin X_i$. Thus, the negation width of the subcircuit F_i computing $f_i(X_i)$ is automatically at most $|X_i|$. Let $C(f_1,\ldots,f_l)$ denote the minimum size of a local DeMorgan circuit simultaneously computing all these functions.

Lemma 5. *Let $f = \varphi(f_1,\ldots,f_l)$ be an semidisjoint m-decomposition of a monotone Boolean function f, and let k be the maximum length of a prime implicant of φ. Then for the negation width $w = km$, we have*

$$C_w(f) \leq C_+(\varphi) + C(f_1,\ldots,f_l)\,.$$

Proof. Let $F(X)$ be a local DeMorgan circuit of size $C(f_1,\ldots,f_l)$ simultaneously computing all functions $f_1(X_1),\ldots,f_l(X_l)$ of the m-decomposition of $f(X)$; hence, $|X_i| \leq m$. Let $Y = \{y_1,\ldots,y_l\}$ be the variables of φ, and let $H(Y)$ be a monotone circuit of size $C_+(\varphi)$ computing φ. The circuit

$F'(X) = H(F(X))$ computes our function $f(X)$. The size of this circuit is the size $C(f_1, \dots, f_l)$ of the circuit F plus the size $C_+(\varphi)$ of the circuit H. So, it remains to show that the negation width of F' does not exceed km.

To show this, take an arbitrary prime implicant p of f. This implicant is of the form $p = \bigwedge_{i \in S} p_i$ for some prime implicant $t = \bigwedge_{i \in S} y_i$ of φ and some prime implicants p_i of the corresponding functions $f_i(X_i)$ with $i \in S$. Since the decomposition $f = \varphi(f_1, \dots, f_l)$ is semidisjoint, all sets X_i of variables with $i \in S$ are disjoint. By Fact 1, for every i, the circuit F must produce some extension $q_i = p_i r_i$ of the prime implicant p_i. Since the circuit F is local, this extension cannot contain any literal $z = x_j^\sigma$ with $x_j \notin X_i$. Thus, the sets of variables of these extensions are disjoint. This ensures that the extension $q = \bigwedge_{i \in S} q_i$ of the prime implicant p of f is nonzero, i.e. q does not contain a variable together with its negation. Since the circuit $H(Y)$ is monotone, the prime implicant $\bigwedge_{i \in S} y_i$ of φ is produced by this circuit. So, the entire extension $\bigwedge_{i \in S} q_i$ of p with at most $|S|m \le km$ negated variables is produced by the entire circuit F'. □

For example, if $f(X) = f_1(X_1) \vee f_2(X_2) \vee \cdots \vee f_l(X_l)$ for some w-element subsets $X_i \subset X$ and functions $f_i : \{0,1\}^{X_i} \to \{0,1\}$, then $C_w(f) \le C(f_1, \dots, f_l) + l - 1$. That is, then the minimum size of a circuit of negation width w for the function f is essentially the same as the minimum size of an unrestricted circuit computing $f_1, \dots, f_l$. This follows from Lemma 5 by taking OR as the outer function φ. The following upper bound for threshold functions uses a less trivial decomposition.

Appendix B. Upper bound for threshold functions

Recall that the threshold-k function Th_k^n accepts a Boolean input vector of length n iff this vector contains at least k ones. The smallest known monotone circuits for Th_k^n have size $\mathcal{O}(n \log k)$; as shown by Kochol [20], the circuits of this size can be constructed using the sorting network of Ajtai, Komlós and Szemerédi [1]. On the other hand, we will now show that Th_k^n can be computed by a DeMorgan circuit of *linear* size $\mathcal{O}(n)$ if negation width $w = k^3$ is allowed.

Lemma 6. *If $k \le n^{1/3}$ and $w = k^3$, then $C_w(\mathrm{Th}_k^n) = \mathcal{O}(n)$.*

Proof. For the sake of simplicity of argumentation, assume that the number of variables n is divisible by an integer parameter $m \ge k$ (to be chosen latter). Divide the sequence X of $|X| = n$ Boolean variables into $\ell := n/m$ consecutive blocks $X_1, \dots, X_\ell$, each of length m. For $i \in [\ell]$ and $r \in \{0, 1, \dots, k\}$, let $\mathrm{Th}_r^m(X_i)$ be the threshold-r functions on the m variables in the ith block X_i.

It is well known (see, for example, [38, Sect. 3.4]) that all the threshold functions $\mathrm{Th}_1^m, \mathrm{Th}_2^m, \dots, \mathrm{Th}_m^m$ on the same set of m variables can be simultaneously computed by a (non-monotone) DeMorgan circuit of size $\mathcal{O}(m)$. So, for every $i = 1, \dots, \ell$, all threshold functions $\mathrm{Th}_0^m(X_i), \mathrm{Th}_1^m(X_i), \dots, \mathrm{Th}_k^m(X_i)$ can be simultaneously computed by a DeMorgan circuit $F_i(X_i)$ of size $\mathcal{O}(m)$. If we put all circuits $F_1(X_1), \dots, F_\ell(X_\ell)$ side by side, then the obtained circuit $F(X)$ of size $\mathcal{O}(\ell m) = \mathcal{O}(n)$ is local (circuits F_i and F_j for $i \ne j$ have no common input gates) and simultaneously computes all functions $\mathrm{Th}_r^m(X_i)$ for $i = 1, \dots, \ell$ and $r = 0, 1, \dots, k$. On the other hand, we have

$$\mathrm{Th}_k^n(X) = \text{ OR of all functions } \bigwedge_{i \in S} \mathrm{Th}_{r_i}^m(X_i) \text{ for } S \subseteq [\ell] \text{ such that } \sum_{i \in S} r_i = k\,. \tag{B.1}$$

This suggests to consider the following monotone Boolean function $\varphi(Y)$ of the set of variables $y_{i,r}$ for $i = 1, \dots, \ell$ and $r = 0, 1, \dots, k$:

$$\varphi(Y) = \text{ OR of all terms } \bigwedge_{i \in S} y_{i,r_i} \text{ for } S \subseteq [\ell] \text{ such that } \sum_{i \in S} r_i = k\,. \tag{B.2}$$

By Eq. (B.1), if we substitute the functions $\mathrm{Th}^m_r(X_i)$ for variables $y_{i,r}$ in $\varphi(Y)$, then the obtained function on the variables in X computes the threshold-k function $\mathrm{Th}^n_k(X)$. So, we have an m-decomposition $\mathrm{Th}^n_k(X) = \varphi(\mathrm{Th}^m_r(X_i) \colon i = 1, \dots, \ell,\ r = 0, 1, \dots, k)$ of $\mathrm{Th}^n_k(X)$, where all the functions Th^m_r can be simultaneously computed by a local DeMorgan circuit of size $\mathcal{O}(\ell m) = \mathcal{O}(n)$. By the definition, no prime implicant of $\varphi(Y)$ contains two variables $y_{i,r}$ and $y_{j,s}$ with $i = j$. So, the sets X_i of variables of functions $\mathrm{Th}^m_r(X_i)$ corresponding to variables $y_{i,r}$ in any one single prime implicant of φ are disjoint, meaning that the decomposition is semidisjoint.

Since prime implicants of the function h have length at most k, Lemma 5 implies that Th^n_k can be computed by a DeMorgan circuit of negation width $w \leq km$ and size at most $C_+(\varphi) + \mathcal{O}(n)$. So, it remains to show that the function φ can be computed by a monotone circuit of size $\mathcal{O}(n)$. This can be done by an easy dynamic programming. Equation (B.2) suggests the following subproblems:

$$P^j_r = \text{ OR of all terms } \bigwedge_{i\in S} y_{i,r_i} \text{ for } S \subseteq [j] \text{ such that } \sum_{i\in S} r_i = r\,.$$

By letting $P^0_r = y_{0,r} \equiv 1$ for all $r = 0, 1, \dots, k$, these functions can be computed by the recursion

$$P^{j+1}_r = \bigvee_{u=0}^{r} P^j_{r-u} \wedge y_{j+1,u} \quad \text{for } r = 0, 1, \dots, k\,.$$

It is easy to see that P^ℓ_k is our target function φ. To compute P^{i+1}_r for all $r = 0, 1, \dots, k$ from already known values, we need only $\mathcal{O}(k^2)$ new gates. So, the entire monotone circuit computing $\varphi = P^\ell_k$ has only $\mathcal{O}(\ell k^2) = \mathcal{O}((n/m)k^2)$ gates. If we take $m := k^2$, this is $\mathcal{O}(n)$ gates, as desired. $\square$

Appendix C. Upper bound for the triangle function

Let $f = \mathrm{Clique}(n, 3)$ be the triangle function, and let $w = n^\epsilon$ for $\epsilon > 0$. Our goal is to prove that this function can be computed by a DeMorgan circuit of negation width w and size $\mathcal{O}(n^{3-\epsilon/4})$, as claimed in Corollary 2.

Recall that the triangle function f has $\binom{n}{2}$ variables $x_{i,j}$, one for each edge $\{i, j\}$ of K_n, and is the OR of all $\binom{n}{3}$ terms $x_{i,l}x_{l,j}x_{i,j}$ for $i < l < j$. Let $Y = (y_{i,j})$ be the $n \times n$ matrix with $y_{i,i} = 0$ and $y_{i,j} = y_{j,i} = x_{i,j}$ for $i \neq j$. Let $Z = (z_{i,j})$ be the Boolean product $Z = Y^2$ of matrix Y with itself. Note that, for every $i \neq j$, $z_{i,j} = 1$ iff there is an $l \notin \{i, j\}$ such that $x_{i,l} = x_{l,j} = 1$. So, $f = \bigvee_{i<j} z_{i,j} \cdot x_{i,j}$. It therefore remains to show that the entries $z_{i,j}$ of the matrix Z can be simultaneously computed by a DeMorgan circuit of negation width $w = n^\epsilon$ and size $\mathcal{O}(n^{3-\epsilon/4})$. We will apply Lemma 5 with a trivial decomposition where the outer function φ is the OR function, and all blocks of variables are disjoint.

Set $m := \frac{1}{2}n^{\epsilon/2}$, and assume for the sake of simplicity that both m and $r := n/m$ are integers. Partition the $n \times n$ matrix Y into disjoint $m \times m$ submatrices $Y_{i,j}$, for $i, j = 1, \dots, r$. The corresponding $m \times m$ submatrix of the product matrix $Z = Y^2$ is then $Z_{i,j} = \bigvee_{k=1}^r Y_{i,k} \cdot Y_{k,j}$, where the OR is componentwise. Using fast matrix multiplication [35, 9, 39], we can compute each of the r matrix products $Y_{i,k} \cdot Y_{k,j}$ by a DeMorgan circuit of size $M = \mathcal{O}(m^\omega)$, where $\omega < 2.373$. Since each such circuit has only $2m^2$ input variables, the negation width of each of these circuits is trivially at most $2m^2 = n^\epsilon$. Using additional rm^2 OR gates, we can then compute all m^2 entries of the matrix $Z_{i,j}$. Since the negation width can only increase at AND gates, the negation width of the resulting circuit remains the same, that is, remains at most $w = n^\epsilon$. Since we only have r^2 submatrices $Z_{i,j}$ of the product matrix Z, and since $M \geq m^2$, all entries of Z can be computed by a circuit of size at most $r^2(rM + rm^2) \leq 2r^3M$. Since $r = n/m$ with $m = \frac{1}{2}n^{\epsilon/2}$, the size of the resulting circuit is at most a constant times $(n/m)^3 m^\omega = n^3/m^{3-\omega} \leq n^{3-\epsilon/4}$, as desired. $\square$

Appendix D. Negation width and communication

Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone Boolean function, and $w \geq 0$ be an integer. Let $d_+ = D_+(f)$ be the minimum depth of a monotone circuit computing f, and $d = D_w(f)$ be the minimum depth of a DeMorgan circuit of negation width w computing f. Corollary 1 gives an upper bound $d_+ \leq d + \log \tau$, where $\tau = 4 \cdot \min\{w^m, m^w\} \cdot \log |PI(f)|$, and m is the maximum length of a prime implicant of f. In this section, we give a *direct* proof of an upper bound $d_+ \leq d + w \cdot \lceil \log(n+1) \rceil$ using the Karchmer–Wigderson communication arguments [18].

As shown by Karchmer and Wigderson [18], d_+ is exactly the maximum, over all inputs $(a,b) \in f^{-1}(1) \times f^{-1}(0)$, of the minimum number of bit of communication required for the players, Alice and Bob, in the following game. When an input pair (a,b) with $f(a) = 1$ and $f(b) = 0$ arrives, the first vector a is given to Alice, and the second vector b to Bob. Their goal is to find a position $i \in [n]$ such that $a_i = 1$ and $b_i = 0$; since f is monotone, such a position always exists.

Now take a DeMorgan circuit F of negation width w computing f, and whose depth is d. In order to show the inequality $d_+ \leq d + w \cdot \lceil \log(n+1) \rceil$ it is enough, by the aforementioned result of Karchmer and Wigderson [18], to design a communication protocol for the game on f which uses at most $d + w \cdot \lceil \log(n+1) \rceil$ bits of communication on all input pairs $(a,b) \in f^{-1}(1) \times f^{-1}(0)$. So, suppose such an input pair (a,b) arrives. If we directly run the Karchmer–Wigderson protocol on the (nonmonotone) circuit F, then only d bits will be communicated. But at the end, the players will only obtain an input literal $z = x_i^\sigma$ such that $z(a) \neq z(b)$. We, however, want to end up with an input literal with $\sigma = 1$ (unnegated variable). To achieve this, we will allow to communicate more bits, and use the fact that the negation width of our circuit F is bounded.

1. Alice takes a vector $a' \leq a$ with a minimal number of 1s which still satisfies $f(a') = 1$. Then $p(a') = 1$ holds for some prime implicant $p = \bigwedge_{i \in S} x_i$ of f; note that $a'_i = 1$ if and only if $i \in S$.
2. Since the negation width of the circuit F is bounded by w, there is a term $r = \bigwedge_{i \in I} \overline{x}_i$ with $|I| \leq w$ such that $p \cdot r \in T(F)$ and $p \cdot r(a') = 1$; hence, $f(a') = 1$ and $S \cap I = \emptyset$.
3. Alice uses $|I| \cdot \lceil \log(n+1) \rceil \leq w \cdot \lceil \log(n+1) \rceil$ bits to send Bob the entire set I of positions of negated variables in her chosen term $p \cdot r$.
4. Since Bob knows Alice's strategy, he knows that Alice's current input a' must have solely zeros in all positions $i \in I$. So, he replaces his original input vector b by the vector $b' \leq b$ with $b'_i = 0$ for $i \in I$, and $b'_i = b_i$ for $i \notin I$. Since $f(b) = 0$ and function f is monotone, we have $f(b') = 0$.
5. The players now replace by constant 0 each negated input gate $\overline{x}_i$ in the circuit F with $i \notin I$, and consider the resulting (not necessarily monotone) circuit F'.
6. Since no negated literal of the term $p \cdot r$ was set to 0, this term belongs also to the set $T(F')$ of terms produced by the new circuit. So, since $p \cdot r(a') = 1$, the circuit F' accepts vector a'. On the other hand, since the original circuit F rejected vector b, and we have only replaced some input gates by zeros, the circuit F' rejects vector b'.
7. So, the players can now run the standard Karchmer–Wigderson protocol ([18, Lemma 2.1]) on the pair (a', b') using the (not necessarily monotone) circuit F' . After communicating at most d bits (the depth of F' can only be smaller than d), they will arrive at some input literal z of the circuit F' such that $z(a') = 1$ and $z(b') = 0$. The literal z is either an unnegated variable x_i, or a negated variable $\overline{x}_i$ for some $i \in I$: the circuit F' has no other non-constant input literals.
8. Since vectors a' and b' coincide in all positions $i \in I$ (both have zeros here), $z = x_i$ must hold for some (unnegated) variable x_i, implying that the found input literal z gives the position i with $a'_i = 1$ and $b'_i = 0$.

Now, $a'_i = 1$ and $a' \leq a$ imply $a_i = 1$. On the other hand, since the position i lies outside I, and since vector b' coincides with b on all such positions, $b'_i = 0$ also implies $b_i = 0$. So, the found position i satisfies $a_i = 1$ and $b_i = 0$, as desired. □

References

[1] M. Ajtai, J. Kolós, and E. Szemerédi. Sorting in $c \log n$ parallel steps. *Combinatorica*, 3:1–9, 1983.

[2] N. Alon and R. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.

[3] K. Amano and A. Maruoka. The potential of the approximation method. *SIAM J. Comput.*, 33(2):433–447, 2004.

[4] K. Amano and A. Maruoka. A superpolynomial lower bound for a circuit computing the clique function with at most $(1/6) \log \log n$ negation gates. *SIAM J. Comput.*, 35(1):201–216, 2005.

[5] E. Blais, C. L. Canonne, I. C. Oliveira, R. A. Servedio, and L.Y. Tan. Learning circuits with few negations. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 40 of *LIPIcs*, pages 512–527, 2015.

[6] Y. Crama and P. L. Hammer, editors. *Boolean Functions: Theory, Algorithms, and Applications*, volume 142 of *Encyclopedia of Mathematics and Its Applications.* Cambridge University Pess, 2011.

[7] P. E. Dunne. Relationship between monotone and non-monotone network complexity. In M.S. Paterson, editor, *Boolean Function Complexity*, volume 169 of *London Math. Soc. Lect. Note Series*, pages 1–24. Cambridge University Press, 1992.

[8] M. J. Fischer. Lectures on network complexity. Technical Report TR-1104, Department of Computer Science, Yale University, 1996.

[9] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proc. of 39th Int. Symp. on Symbolic and Algebraic Computation*, pages 296–303, 2014.

[10] S. Guo and I. Komargodski. Negation-limited formulas. *Theor. Comput. Sci.*, 660:75–85, 2017.

[11] S. Guo, T. Malkin, I. C. Oliveira, and A. Rosen. The power of negations in cryptography. In *Proc. of 12th Theory of Cryptography Conference, TCC*, volume 9014 of *Lect. Notes in Comput. Sci.*, pages 36–65. Springer, 2015.

[12] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *SIAM J. Comput.*, 2:225–231, 1973.

[13] S. Jukna. Two lower bounds for circuits over the basis $\{\wedge, \vee, \neg\}$. In *Math. Found. of Comput. Sci.,MFCS'88*, volume 324 of *Lect. Notes in Comput. Sci.*, pages 371–380. Springer, 1988.

[14] S. Jukna. Combinatorics of monotone computations. *Combinatorica*, 19(1):65–85, 1999.

[15] S. Jukna. On the minimum number of negations leading to super-polynomial savings. *Inf. Process. Lett.*, 89(2):71–74, 2004.

[16] S. Jukna. *Boolean Function Complexity: Advances and Frontiers.* Springer-Verlag, 2012.

[17] S. Jukna and A. Lingas. Lower bounds for demorgan circuits of bounded negation width. In *36th Int. Symp. Theoretical Aspects of Comput. Sci., STACS*, volume 126 of *LIPIcs*, pages 41:1–41:17, 2019.

[18] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3:255–265, 1990.

[19] V. M. Khrapchenko. On a relation between the complexity and the depth of formulas. In *Methods of Discrete Analysis in Synthesis of Control Systems*, volume 32, pages 76–94. Institute of Mathematics. Novosibirsk, 1978. (In Russian).

[20] M. Kochol. Efficient monotone circuits for threshold functions. *Inf. Process. Lett.*, 32:121–122, 1989.

[21] S. Koroth and J. Sarma. Depth lower bounds against circuits with sparse orientation. *Fundam. Inform.*, 152(2):123–144, 2017.

[22] A. Lingas. Small normalized circuits for semi-disjoint bilinear forms require logarithmic and-depth. *Theor. Comput. Sci.*, 820:17–25, 2020. Preliminary version in: proc. of 33rd Comput. Complexity Conf., 2018. Extended version of the conference version towards the journal version in: ECCC Report Nr. 108, 2018.

[23] A. A. Markov. On the inversion complexity of systems of boolean functions,. *Dokl. Akad. Nauk SSSR*, 116:917–919, 1957. English transl. in: J. ACM 5(4), 331–334.

[24] H. Morizumi. Limiting negations in formulas. In *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, pages 701–712, 2009.

[25] H. Morizumi. Limiting negations in non-deterministic circuits. *Theor. Comput. Sci.*, 410(38-40):3988–3994, 2009.

[26] H. Morizumi. Limiting negations in probabilistic circuits. New Trends in Algorithms and Theory of Computation, Departmental Bulletin Paper 1799, pages 81–83, Kyoto University Research Information Repository, Juni 2012.

[27] E. A. Okolnishnikova. On the influence of one type of restrictions to the complexity of combinational circuits. *Diskrete Analysis*, 36:46–58, 1981. (In Russian).

[28] T. Pitassi and R. Robere. Strongly exponential lower bounds for monotone computation. In *Proc. 49th Ann. ACM Symp. on Theory of Computing, STOC*, pages 1246–1255, 2017.

[29] R. Raz and Wigderson. Probabilistic communication complexity of boolean relations. In *Proc. of 30th Ann. Symp. on Foundations of Computer Sci., FOCS*, pages 562–567, 1989.

[30] R. Raz and A. Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992.

[31] A. A. Razborov. Lower bounds for the monotone complexity of some boolean functions. *Soviet Math. Dokl.*, 31:354–357, 1985.

[32] A. A. Razborov. Lower bounds on monotone complexity of the logical permanent. *Math. Notes of the Acad. of Sci. of the USSR*, 37(6):485–493, 1985.

[33] B. Rossman. Correlation bounds against monotone NC^1. In *Proc. of 30th Comput. Complexity Conf.*, volume 33 of *LIPIcs*, pages 392—411, 2015.

[34] P. M. Spira. On time–hardware complexity tradeoffs for boolean functions. In *Proc. of 4th Hawaii Symp. on System Sciences*, pages 525–527. Western Periodicals Company, North Hollywood, 1971.

[35] V. Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13:354–356, 1969.

[36] É. Tardos. The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 7(4):141–142, 1987.

[37] I. Wegener. Relating monotone formula size and monotone depth of Boolean functions. *Inf. Process. Lett.*, 16:41–42, 1983.

[38] I. Wegener. *The complexity of Boolean functions.* Wiley-Teubner, 1987.

[39] V. Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *Proc. of 44th Symp. on Theory of Comput., STOC*, pages 887–898, 2012.