Lower Bounds for Circuits of Bounded Negation Width

Stasys Jukna[1] [*] Andrzej Lingas[2] [†]

[1] *Institute of Computer Science, Goethe University Frankfurt, Germany*

[2] *Department of Computer Science, Lund University, Box 118, 22100 Lund, Sweden*

Abstract

We consider Boolean circuits over $\{\vee, \wedge, \neg\}$ with negations applied only to input variables. To measure the "amount of negation" in such circuits, we introduce the concept of their "negation width." In particular, a circuit computing a monotone Boolean function $f(x_1, \ldots, x_n)$ has negation width w if no nonzero term produced (purely syntactically) by the circuit contains more than w distinct negated variables. Circuits of negation width $w = 0$ are equivalent to monotone circuits, while those of negation width $w = n$ have no restrictions. Our motivation is that already circuits of moderate negation width $w = n^\epsilon$ for an arbitrarily small constant $\epsilon > 0$ can be even exponentially stronger than monotone circuits.

We show that the *size* of any circuit of negation width w computing f is roughly at least the minimum size of a monotone circuit computing f divided by $K = \min\{w^m, m^w\}$, where m is the maximum length of a prime implicant of f. We also show that the *depth* of any circuit of negation width w computing f is roughly at least the minimum depth of a monotone circuit computing f minus $\log K$. Finally, we show that *formulas* of bounded negation width can be balanced without increasing their negation width.

1 Introduction and results

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 $\{\vee, \wedge\}$ circuits, we can currently prove only depressingly small (linear) lower bounds on the size of $\{\vee, \wedge, \neg\}$ circuits when there are no restrictions on the number or the usage of negation gates.

In this paper, we concentrate on *DeMorgan circuits*, that is, on $\{\vee, \wedge, \neg\}$ circuits with fanin-2 OR and AND gates, and with negation applied only to input variables. In other words, a DeMorgan circuit is a circuit with fanin-2 OR and AND gates, while inputs are variables $x_1, \ldots, x_n$ and their negations $\overline{x}_1, \ldots, \overline{x}_n$; to simplify notation, we will write $\overline{x}_i$ instead of $\neg x_i$. DeMorgan circuits are sometimes called *normalized* circuits [21], *standard* circuits [34, Section 6.13] or circuits *with tight negations* [28]. A circuit is a *formula* if its underlying graph is a tree. A *monotone circuit* is a DeMorgan circuit with no negated input variables at all. By just doubling the circuit size and using DeMorgan rules, any circuit over

[*] Research supported by the DFG grant JU 3105/1-1 (German Research Foundation).
Affiliated with the Institute of Data Science and Digital Technologies, Faculty of Mathematics and Informatics of Vilnius University, Lithuania. Email: stjukna@gmail.com

[†] Research supported in part by VR grant 2017-03750 (Swedish Research Council).
Email: Andrzej.Lingas@cs.lth.se

$\{\vee, \wedge, \neg\}$ of size s can be converted to a DeMorgan circuit computing the same function and having size at most $2s$ (see, for example, [8, Theorem 3.1]).

We use standard terminology regarding Boolean functions (see, for example, [34]). 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 contain a variable and its negation. An *implicant* of a Boolean function $f(x_1, \ldots, x_n)$ 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 of f is a *prime implicant* of f if no proper sub-term of p is an implicant of f. 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 Negation width of circuits

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 trivial fact that every DeMorgan circuit not only computes a particular Boolean function but also *produces* (purely syntactically) some set of terms in a natural way.

Definition 1 (Terms produced by circuits)**.** The set of terms produced at an input gate holding a literal z is a singleton-set $\{z\}$. The set produced at an OR gate is a union of sets produced at its two inputs, while the set produced at the AND gate is obtained by taking the AND of every term produced at one of its inputs with every term produced at the other input.

The set $T(C)$ of terms produced by the entire circuit C is the set of terms produced at the output gate of C. During the production of terms, we use the "shortening" axiom $x \wedge x = x$, but do not use the "annihilation" axiom $x \wedge \overline{x} = 0$. So, $T(C)$ can contain also zero terms, those having a variable and its negation.[1] Easy induction on the circuit size shows that the Boolean function f computed by a circuit C is the function computed by the OR of all terms produced by C.

If the circuit C is *monotone* (has no negated inputs at all), then we clearly have $PI(f) \subseteq T(C)$, 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, Okol'nishnikova [22] exhibited an explicit monotone Boolean function f of n variables which can be computed by a monotone circuit of size $O(n)$, but any monotone circuit C satisfying $T(C) = PI(f)$ must have $2^{\Omega(n^{1/4})}$ gates.

The situation when the computed function f is monotone, but a DeMorgan circuit C is *not* necessarily monotone, is even more subtle: then even the inclusion $PI(f) \subseteq T(C)$ does not need to hold. For example, the function $f = x \vee y$ is computed by a circuit $C = x\overline{y} \vee y$, but $T(C) = \{x\overline{y}, y\}$ whereas $PI(f) = \{x, y\}$.

[1] 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 $T(C)$ 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.

However, we have the following simple and well-known property of (not necessarily monotone) DNFs computing monotone Boolean functions (see, for example, [7, Theorem 1.24 on p. 37]): if D is a (not necessarily monotone) DNF computing a monotone Boolean function f, then the monotone DNF obtained from D by first removing all zero terms, and then removing all occurrences of negated variables from the remaining terms, also computes f.

If C is a DeMorgan circuit computing f, then the OR of terms in $T(C)$ computes f. So, by the aforementioned fact [7, Theorem 1.24 on p. 37], for every prime implicant p of f, the set $T(C)$ must contain either p itself or at least one *extension* of p, that is, a nonzero term of the form $p \cdot r$, where the term $r = \overline{x}_{i_1} \cdots \overline{x}_{i_l}$ consists solely of negated variables. This motivates the following measure of DeMorgan circuits computing monotone Boolean functions.

Definition 2 (Negation width)**.** A DeMorgan circuit computing a monotone Boolean function f has *negation width* w if for every prime implicant p of f, the circuit produces either p itself or some its extension containing at most w negated variables.

There are no other restrictions on the remaining produced terms, except a trivial one that the function computed by the OR of all produced terms must coincide with f. Note that the negation width w of any DeMorgan circuit computing f satisfies $0 \leq w \leq n - m$, where m is the minimum length of a prime implicant of f. Also, minimal circuits of negation width $w = 0$ are monotone circuits: just replace each negated input gate $\overline{x}_i$ by constant 0.

The negation width (without using this term) was already considered by Amano and Maruoka [3, Sect. 4] (we recall their result right before Corollary 2). Examples of sufficient 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 [24], and Guo et al.[12].
- No input-output path has more than[2] $\log w$ AND gates; such circuits computing quadratic forms (multi-output functions) were considered in [21].
- 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 puts no restrictions on the length of produced *zero* terms. So, at intermediate gates, the circuit can produce very long terms, and then cancellate them (turn them into zero terms). 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 input variables by constants 1, and the resulting monotone circuit would still compute f.

We will also consider DeMorgan circuits of bounded *average* negation width. Let C be a DeMorgan circuit computing a monotone Boolean function f.

Definition 3 (Average negation width)**.** The *negation width* of a prime implicant $p \in PI(f)$ in the circuit C is the minimum number w such that $T(C)$ contain an extension of p with at most w negated variables. The *average negation width* of the circuit C is the average, over all prime implicants $p \in PI(f)$, of the negation width of p in C.

Note that a circuit C computing f has negation width w if *every* prime implicant of f has negation width at most w in C. Average negation width relaxes this "every" requirement.

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

1.2 Motivation

Our motivation to consider circuits of bounded negation width w is that allowance of even moderately large negation width $w = n^\epsilon$ for an arbitrarily small constant $\epsilon > 0$ *can* substantially reduce the size of monotone circuits.

Example 1. The *triangle function* has one variable for every edge of the complete graph K_n on $\{1, \ldots, n\}$, and accepts a subgraph G of K_n if and only if G contain a triangle. It is known that this function requires monotone circuits of almost cubic size $n^{3-o(1)}$ [26, 2]. According to Claim 1 in Appendix A, the function can be computed in already *sub-cubic* size $n^{3-\epsilon/4}$ if negation width $w = n^\epsilon$ is allowed; we will show (Corollary 1) that about $n^{3-4\epsilon}$ gates are then still necessary.

Example 2. The threshold-k function Th_k^n accepts a Boolean input of length n if and only if it contains at least k ones. The smallest known monotone circuits for Th_k^n have size of order $n \log k$ (see, for example, [19]). On the other hand, for $k \le n^{1/3}$, the function Th_k^n *can* be computed by a DeMorgan circuit of *linear* size $O(n)$ if negation width $w = k^3$ is allowed (see Claim 3 in Appendix A.3).

Using monotone circuit lower bounds of Razborov [27] and Tardos [32], one can show that, on some monotone Boolean functions, super-polynomial, and even exponential gaps between the size of monotone circuits and circuits of moderate negation width can be achieved; see Examples 4 and 5 in Appendix A.

1.3 Results

Our first result relates (non-monotone) DeMorgan circuits and formulas of bounded negation width to *monotone* circuits and formulas.

Theorem 1. *Let f be a monotone Boolean function with all prime implicants of length at most m. Let s be the minimum size of a monotone circuit computing f, and d the minimum depth of such a circuit. Then any DeMorgan circuit of negation width w computing f must have size at least $s/K-1$ and depth at least $d-\log K$, where $K = 8\min\{m^w, w^m\}\cdot\log|PI(f)|$.*

Our second result concerns circuits of bounded *average* width.

Definition 4. A monotone Boolean function h *K-approximates* a monotone Boolean function f if there is an OR g of at least a $1/K$ portion of prime implicants of f such that $g \le h \le f$ holds.

Theorem 2. *Let f be a monotone Boolean function with all prime implicants of length at most m. Let $w \ge 0$ and $K = 8 \cdot \min\{m^{2w}, (2w)^m\}$. If every monotone circuit K-approximating f requires at least t gates, then every DeMorgan circuit of average negation width w computing f must also have at least t gates.*

Remark 1. Note the difference between Theorems 1 and 2. To apply Theorem 1, one can *directly* use known lower bounds on the monotone circuit complexity of the function f *themselves*. 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 K. 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.

Our third result extends the well-known Spira's depth reduction theorem [30] to DeMorgan formulas of bounded negation width: it shows that such formulas can also be balanced *without* increasing their negation width.

Theorem 3. *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 $3 \cdot \log s$ and the same negation width w.*

The rest of the paper is organized as follows. In Section 2, we shortly recall previous work on $\{\vee, \wedge, \neg\}$ circuits with limited use of negations. In Section 3, a special type of "random subcircuits" is introduced. Sections 4–6 are devoted to the proof of our main results (Theorems 1–3). In Section 7, we give some applications of our general results to specific Boolean functions. Appendix A contains proofs of the upper bounds claimed in our motivating examples (Examples 1–4). Appendix B gives an alternative proof of the lower bound on the depth of DeMorgan circuits of bounded negation width using the communication complexity arguments.

2 Related work

The effect of negations on the size or depth of $\{\vee, \wedge, \neg\}$ circuits was mainly considered by either restricting the total *number* of used negation gates, or by restricting the *usage* of negations.

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. We refer to [16, Chapter 10] and the papers cited therein for this line of research; see also [29, 5, 12] for more recent developments in this direction.

Another line of research (which attracted much less attention, and which we follow in this paper) was to restrict the *usage* of negation gates. One of the first results in this direction was proved by Raz and Wigderson [24] for the s-t connectivity function $f = \mathrm{STCON}(n)$ of n-vertex graphs. This is a monotone Boolean function of $\binom{n}{2}$ 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 accepts the assignment if and only if the specified graph contains a path from vertex $s = 1$ to vertex $t = n$. It was know that monotone circuits for this require depth $\Omega(\log^2 n)$; see Karchmer and Wigderson [17], or Grigni and Sipser [10] for a simpler proof. Raz and Wigderson [24, Theorem 4.1] extend this to non-monotone circuits: if $w \leq n^{2-\epsilon}$ for a constant $\epsilon > 0$, then any DeMorgan circuit for f with at most w negated input variables must have depth $\Theta(\log^2 n)$.

Our Theorem 3 holds also in the special case when w is the total number of allowed negated input variables in DeMorgan formulas. So, the result of Raz and Wigderson implies that if a DeMorgan *formula* computes $\mathrm{STCON}(n)$ and uses only $w \leq n^{2-\epsilon}$ negated inputs for a constant $\epsilon > 0$, then the formula must have size $n^{\Omega(\log n)}$. On the other hand, $\mathrm{STCON}(n)$ can be computed by an even monotone *circuit* of size $O(n^3)$ arising from the well-known Bellman–Ford dynamic programming algorithm. So, this shows a super-polynomial gap between monotone *circuits* and non-monotone DeMorgan *formulas* with bounded number of allowed negated inputs.

Guo et al. [12] have recently used the communication complexity approach of Karchmer and Wigderson [17] to prove 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.

DeMorgan circuits with bounded total number of allowed negated input variables (considered in [24, 12]) have the following *functional* restriction. Recall that a function $g(x_1, \ldots, x_n)$ is monotone in the ith variable x_i if changing the value of x_i from 0 to 1 (while keeping the other variables fixed) cannot change the value of g from 1 to 0; hence, a function is monotone if and only if it is monotone in each of its variables. Now, if C is a DeMorgan circuit with w negated input variables, then the functions computed at all gates of C are monotone in *one and the same* set of at least $n - w$ variables, namely, are monotone in all those variables x_i whose negations are not inputs of C.

Koroth and Sarma [20] relax this restriction, and say that a (not necessarily DeMorgan) circuit C over $\{\vee, \wedge, \neg\}$ has *orientation weight* w if the function computed at each gate of C is monotone in at least $n-w$ variables; for different gates, these sets of variables can be different. Using the communication complexity approach of Karchmer and Wigderson [17], they prove that the depth of any circuit over $\{\vee, \wedge, \neg\}$ of orientation weight w computing a monotone function f is at least the minimum depth of a monotone circuit computing f divided by $4w+1$. They also prove an interesting result showing the limitations of their relaxation: there exists a (non-explicit) monotone Boolean function f which cannot be computed by poly-log depth monotone circuits, but f can be computed by a circuit of poly-log depth if at some two of its gates non-monotone Boolean functions can be computed.

In circuits of nonzero orientation w, negations are allowed to be applied to inner gates (not only to input variables). On the other hand, the (functional) use of such NOT gates is severely restricted: the function computed at each NOT gate in such a circuit cannot depend on more than $2w$ variables. To see this, let $g = \neg h$ be the function computed at some NOT gate, and h the function computed at its input. Let X be the set of variables on which g depends. We know that neither g nor h can be non-monotone in more than w variables. If g is monotone in a variable $x_i \in X$, then h is non-monotone in x_i. So, g cannot be monotone in more than w variables of X. Since, due to the orientation width restriction, the function g itself cannot be non-monotone in more than w variables, the desired upper bound $|X| \leq 2w$ follows.

Remark 2. Our relaxation (the negation width, see Definition 2) is of a more "syntactic" nature than that of the orientation weight in [20], but is also of a similar spirit. Instead of requiring that the produced extensions of prime implicants can only use negated variables from one *fixed* subset of $\leq w$ negated variables (as in [24, 12]), we now allow the extensions to use *different* subsets of $\leq w$ negated variables for different prime implicants. In contrast to [20], we have no restrictions on functions computed at *intermediate* gates: only terms produced at the end do matter. And only *nonzero* terms do matter: produced zero terms do not contribute to the negation width at all.

Remark 3. Examples 1-2 given in the introduction (as well as Examples 4-5 in Appendix A) show that already moderately negation width (n^ϵ for an arbitrary small constant $\epsilon > 0$) *can* substantially reduce the size of monotone circuits (see also Appendix A). We are not aware of any similar separating examples for restrictions on the use of negations considered in [24, 20, 12]: restricted number of allowed negated input variables, or restricted orientation weight.

3 Random subcircuits

Let $f(x_1,\ldots,x_n)$ be a monotone Boolean function, and C be a DeMorgan circuit computing f. For a subset $Y=\{x_i\colon i\in I\}$ of variables, the *monotone Y-subcircuit* of C is obtained as follows.

1. First, set to 0 all variables in Y; so, for every $i\in I$, the input gate x_i is set to 0, while the negated input gate $\overline{x}_i$ is set to 1.
2. Then replace by constant 0 each of the remaining negated input gates $\overline{x}_j$ for $j\notin I$.
3. Finally, eliminate constant input gates through repeated replacements of $0\land u$ by 0, $1\lor u$ by 1, and $0\lor u$, $1\land u$ by u.

Schematically:

$$C(x,y,\overline{x},\overline{y}) \overset{\text{Step 1}}{\mapsto} C(x,0,\overline{x},1) \overset{\text{Step 2}}{\mapsto} C(x,0,0,1) \overset{\text{Step 3}}{\mapsto} C_+(x)\,.$$

Example 3. Consider the DeMorgan formula $C=(x_1\lor x_2\lor\overline{x}_3)(\overline{x}_1\lor\overline{x}_2\lor x_5)(x_3\lor x_4\lor\overline{x}_5)$, and $Y=\{x_1,x_4\}$. After the first step, we obtain the formula $(0\lor x_2\lor\overline{x}_3)(1\lor\overline{x}_2\lor x_5)(x_3\lor 0\lor\overline{x}_5)$. After the second step, we obtain the formula $(0\lor x_2\lor 0)(1\lor 0\lor x_5)(x_3\lor 0\lor 0)$ and, after the elimination of constants, the resulting monotone sub-formula of C is x_2x_3.

The following lemma is just a simple observation.

Lemma 1. *If a DeMorgan circuit C computes a monotone Boolean function f, then the monotone Boolean function h computed by any monotone subcircuit of C satisfies $h\le f$.*

Proof. Take an arbitrary subset $Y=\{x_i\colon i\in I\}$ of variables, and let C_+ be a monotone Y-subcircuit of C. Let h be the monotone Boolean function computed by C_+. We have to show that $h\le f$ holds.

Let g be a monotone Boolean function computed by the circuit C' obtained from C by setting all variables in Y to 0. Since the function f is monotone, we have $g\le f$, and even $PI(g)\subseteq PI(f)$. Now, the circuit C_+ is obtained from C' by replacing by zeroes all remaining (not yet set to constant 1) negated input variables. So, the set $T(C_+)$ of terms produced by C_+ is obtained from $T(C')$ by removing all terms with at least one negated variable (including all zero terms). Since g is the OR of all terms in $T(C')$, and h is the OR of all terms in $T(C_+)$, the inclusion $T(C_+)\subseteq T(C')$ yields $h\le g$. So, $h\le g\le f$, as desired. □

Let $m\ge 3$ and $w\ge 1$ be integers. A *random (m,w)-subcircuit* $\boldsymbol{C}$ of C is a monotone $\boldsymbol{Y}$-subcircuit of C for $\boldsymbol{Y}\subseteq\{x_1,\ldots,x_n\}$ being a random subset of variables with each variable included in $\boldsymbol{Y}$ independently with probability $1-\epsilon$, where

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

The next lemma is just a refinement of [21, Lemma 3].

Lemma 2. *Let C be a DeMorgan circuit computing a monotone Boolean function f, and $\boldsymbol{C}$ be a random (m,w)-subcircuit of C for $m\ge 3$ and $w\ge 1$. If a prime implicant p of f has length at most m, and has negation width at most w in C, then p is produced by $\boldsymbol{C}$ with probability at least $1/K$, where $K\le 4m^w$ for $w=1,2$, and $K\le 4\cdot\min\{m^w,w^m\}$ for $w\ge 3$.*

Proof. Since the negation width of the prime implicant p in the (deterministic) circuit C is at most w, the set $T(C)$ of terms produced by C must contain a nonzero term $p \cdot r$, where term r consists solely of $l \leq w$ negated variables. The probability that all these negated l variables are set to 0 (and hence, that the term r is set to 1) is at least $(1-\epsilon)^l \geq (1-\epsilon)^w$. The probability that none of the $t \leq m$ variables of p is set to 0 is $\epsilon^t \geq \epsilon^m$. So, the prime implicant p is produced by $\boldsymbol{C}$ with probability at least $\alpha := \epsilon^m(1-\epsilon)^w$. So, it remains to show that $\alpha \geq 1/K$.

We will use two simple facts: $(1-1/t)^t \geq 1/4$ holds for all integers $t \geq 2$, and $t^s \geq s^t$ holds for all integers $3 \leq t \leq s$. 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 $(t^2/(t^2-1))^t$ where, by the Bernoulli inequality, the latter term $(1+1/(t^2-1))^t \geq (1+1/t^2)^t$ is at least $1 + t \cdot (1/t^2) = (t+1)/t$. So, $(1-1/t)^t \geq a_2 = 1/4$ holds for all integers $t \geq 2$. To see the second fact, observe that $t^s \geq s^t$ is equivalent to $t^{1/t} \geq s^{1/s}$, and that the sequence $t^{1/t}$ for $t = 3, 4, \ldots$ is non-decreasing. This latter claim can be shown by an easy induction on t. Namely, the inequality $t^{1/t} \geq (t+1)^{1/(t+1)}$ is equivalent to $t^{t+1} \geq (t+1)^t$, or $t \geq (1+1/t)^t$. Since $(1+1/t)^t$ is at most the Euler number $\mathrm{e} < 3$, and since we assumed that $t \geq 3$, the inequality follows.

Now, if $w \geq m$, then $\epsilon = 1/w$, and we obtain $\alpha = (1/w)^m(1-1/w)^w \geq \frac{1}{4}w^{-m} \geq \frac{1}{4}m^{-w}$, where the last inequality holds because $m \geq 3$. If $w < m$, then $\epsilon = 1 - 1/m$, and we obtain $\alpha = (1-1/m)^m(1/m)^w \geq \frac{1}{4}m^{-w} \geq \frac{1}{4}w^{-m}$, where the last inequality holds, as long as $w \geq 3$. In both cases, we have that α is at least $\frac{1}{4} \cdot \max\{m^{-w}, w^{-m}\} \geq 1/K$, as desired. If $w = 1$ or $w = 2$, then $w < m$, and we have $\alpha \geq \frac{1}{4}m^{-w}$. □

4 Proof of Theorem 1

Theorem 1 is a direct consequence of the following lemma.

Lemma 3 (Reduction lemma)**.** *Let f be a monotone Boolean function with all prime implicants of length at most m. If C is a DeMorgan circuit of negation width w computing f, then there exist at most $K = 8 \cdot \min\{m^w, w^m\} \cdot \log|PI(f)|$ monotone sub-circuits of C whose OR also computes f.*

In particular, if C has size s and depth d, then the resulting monotone circuit has size $s_+ \leq (s+1)K$ and depth $d_+ \leq d + \log K$. Hence, the lower bounds $s \geq s_+/K - 1$ and $d \geq d_+ - \log K$ claimed in Theorem 1 follow.

Proof. Let $\boldsymbol{C}$ be a random (m, w)-subcircuit of C, and take K independent copies $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_K$ of $\boldsymbol{C}$. Since the circuit C has negation width w, every prime implicant of f must have negation width at most w in C. By Lemma 2, we have $\Pr\{p \in T(\boldsymbol{C})\} \geq 1/t$ for every prime implicant $p \in PI(f)$ of f, where $t := 4 \cdot \min\{w^m, m^w\}$. Note that $K/t = 2 \cdot \log|PI(f)|$. Hence, for every prime implicant $p \in PI(f)$, we have

$$\Pr\{p \notin T(\boldsymbol{C}_i) \text{ for all } i = 1, \ldots, K\} \leq (1-1/t)^K \leq \mathrm{e}^{-K/t} \leq |PI(f)|^{-2}.$$

By the union bound, the probability that some prime implicant of f is produced by *none* of the circuits $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_K$ is strictly smaller than 1. Consequently, there must be a sequence $C_1, \ldots, C_K$ of realizations of these circuits such that *every* prime implicant of f is produced by at least one of these circuits. Replace all negated input variables in these circuits by constants 0. Let $C_1^+, \dots, C_K^+$ be the resulting monotone circuits, and consider the monotone Boolean function $h = h_1 \vee \cdots \vee h_K$, where h_i is the function computed by C_i^+. By Lemma 1, we have $h \leq f$. On the other hand, the inclusion $PI(f) \subseteq T(C_1^+) \cup \cdots \cup T(C_K^+)$ yields the converse inequality $f \leq h$. So, the OR of the circuits $C_1^+, \dots, C_K^+$ computes $h = f$, as desired. □

5 Proof of Theorem 2

Let f be a monotone Boolean function with all prime implicants of length at most m. Let C be a DeMorgan circuit of average negation width w computing f. Recall that a monotone Boolean function h *K-approximates* a monotone Boolean function f if there is an OR g of at least a $1/K$ portion of prime implicants of f such that $g \leq h \leq f$ holds. Now suppose that every monotone circuit K-approximating f for $K = 8 \cdot \min\{m^{2w}, (2w)^m\}$ requires t gates. Our goal is to show that then the circuit C must have at least t gates.

Since the average negation width of C is w, some set $P \subseteq PI(f)$ of $|P| \geq \frac{1}{2}|PI(f)|$ prime implicants of f have negation width at most $2w$ in C. Let $\boldsymbol{C}$ be a random (m, w)-subcircuit of C. By Lemma 2, we have $\Pr\{p \in T(\boldsymbol{C})\} \geq 2/K$ for every prime implicant $p \in P$. So, the expected number of prime implicants $p \in P$ produced by $\boldsymbol{C}$ is at least $2|P|/K \geq |PI(f)|/K$.

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

6 Proof of Theorem 3

It is long known that DeMorgan formulas can be balanced: every DeMorgan formula of size s can be simulated by a DeMorgan formula of depth at most $c \log s$. This was first proved by Spira [30] with $c < 3.42$, while the best currently known constant $c < 1.73$ is due to Khrapchenko [18].

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 subformulas (not just to input variables), which may result in a much larger negation width.

We therefore will argue a bit differently: we first show that *monotone* formulas can be turned into balanced formulas by preserving the produced by the original formula terms, and then use this additional property to prove Theorem 3 itself.

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.

Lemma 4. *For every monotone formula F of size s, there is an equivalent monotone formula F' of depth at most $3 \log s$ such that $T(F') \supseteq T(F)$.*

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. The key observation (already made by Brent, Kuck and Maruyama [6], and Wegener [33]) is that, due to the monotonicity, $F_1(x) = 0$ implies $F_0(x) = 0$. Thus the formula $H \wedge F_1 \vee F_0$ is equivalent to F.

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 $1 + 3\log(s/2)$ such that

$$T(F_1) \subseteq T(F_1')\,, \quad T(F_0) \subseteq T(F_0') \quad \text{and} \quad T(H) \subseteq T(H')\,. \tag{1}$$

Thus, the resulting entire formula

$$F' = H' \wedge F_1' \vee F_0' \tag{2}$$

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

It remain 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 $\{t_1 \wedge t_2 \colon t_1 \in T_1, t_2 \in T_2\}$. This yields

$$T(F) = [T(H) * Q] \cup R\,, \quad T(F_1) = Q \cup R \quad \text{and} \quad T(F_0) = R\,. \tag{3}$$

So,

$$\begin{aligned} T(F') &\overset{(2)}{=} [T(H') * T(F_1')] \cup T(F_0') \overset{(1)}{\supseteq} [T(H) * T(F_1)] \cup T(F_0) \\ &\overset{(3)}{=} [T(H) * (Q \cup R)] \cup R \supseteq [T(H) * Q] \cup R \overset{(3)}{=} T(F)\,. \end{aligned}$$ □

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 $\mathrm{D}_w(f) \leq 3 \cdot \log s$ holds.

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:

(a) 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 4 to the formula $F(x, y)$. This gives us a 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 (a). 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

For a monotone Boolean function $f(x_1, \dots, x_n)$, $\mathrm{C}_w(f)$ will denote the minimum size of a DeMorgan circuit of negation width w computing f, while $\mathrm{C}_+(f)$ will denote the minimum size of a monotone circuit computing f. In the case of DeMorgan *formulas*, these measures are denoted by $\mathrm{L}_w(f)$ and $\mathrm{L}_+(f)$; in this case, the *size* of a formula is the number of leaves of the underlying tree. Let also $\mathrm{D}_w(f)$ denote the minimum *depth* of a DeMorgan circuit of negation width w computing f, and let $\mathrm{D}_+(f)$ denote the minimum depth of a monotone circuit computing f.

Theorem 1 directly yields the following lower bounds on the size and depth of DeMorgan circuits of bounded negation width. Let $f(x_1, \dots, x_n)$ be a monotone Boolean function with M prime implicants, each of length at most m. Then for any $w \geq 0$, we have

$$\mathrm{C}_w(f) \geq \frac{\mathrm{C}_+(f)}{K} - 1\,, \quad \mathrm{L}_w(f) \geq \frac{\mathrm{L}_+(f)}{K} - 1 \quad \text{and} \quad \mathrm{D}_w(f) \geq \mathrm{D}_+(f) - \log K\,, \tag{4}$$

where

$$K = 8 \cdot \min\{m^w, w^m\} \cdot \log|PI(f)|\,. \tag{5}$$

In Appendix B, we also give an entirely different proof of a lower bound $\mathrm{D}_w(f) \geq \mathrm{D}_+(f) - w \cdot \lceil \log(n+1) \rceil$ on the depth of DeMorgan circuit of negation width w using communication complexity arguments. The argument exposes the meaning of the negation width restriction from the communication point of view.

The *k-clique function* $\mathrm{CLIQUE}(n,k)$ has $\binom{n}{2}$ variables, one for each edge of the complete graph K_n on $[n] = \{1, \dots, n\}$. Every assignment of Boolean values to these variables specifies a subgraph of K_n, and the function accepts the assignment if and only if the specified graph contains a complete graph on k or more vertices; note that we do not require k to be an integer.

Corollary 1 (Small cliques)**.** *There are absolute constants $c_1, c_2 > 0$ such that, if $f = \mathrm{CLIQUE}(n,3)$, and $w \leq n^\epsilon$ for $\epsilon > 0$, then*

$$c_1 n^{3-4\epsilon} \leq \mathrm{C}_w(f) \leq c_2 n^{3-\epsilon/4}\,.$$

Proof. Here we only show the lower bound; the proof of the upper bound $\mathrm{C}_w(f) = O(n^{3-\epsilon/4})$ is given in Appendix A.2 (see Claim 2). As shown by Alon and Boppana [2, Lemma 3.14], $\mathrm{C}_+(f) = \Omega\left(n^3/\log^3 n\right)$ holds. Since f has $M = \binom{n}{3} \leq n^3$ prime implicants, each of length $m = 3$, the parameter K in Eq. (5) is at most a constant times $w^m \cdot \log M \leq 3n^{3\epsilon} \log n$, and Eq. (4) yields the desired lower bound $\mathrm{C}_w(f) \geq \mathrm{C}_+(f)/K - 1 = \Omega(n^{3-4\epsilon})$. □

Amano and Maruoka [3, Theorem 4.2] proved that, for any $3 \leq k \leq n^{2/3}$, DeMorgan circuits of negation width $w = o(\sqrt{k})$ computing $f = \mathrm{CLIQUE}(n,k)$ require $2^{\Omega(\sqrt{k})}$ gates. (In their definition of the negation width [3, Definition 4.1], they use different terminology, but it is not difficult to see that their measure coincides with that given in our Definition 2.) Note, however, that here the allowed negation width $w = o(\sqrt{k})$ is much smaller than the clique size k. When combined with the lower bound of Alon and Boppana [2] for cliques of moderate (logarithmic) size, Eq. (4) directly yields super-polynomial lower bounds also when the allowed negation width is much larger, even *exponential*, in the cliques size.

Corollary 2 (Moderate cliques)**.** *Let $f = \mathrm{CLIQUE}(n,s)$ with $k = \log^{1/3} n$. Then $\mathrm{C}_w(f) = n^{\Omega(k)}$ holds for $w = 2^k$.*

Proof. By [2, Theorem 3.16], we have $\mathrm{C}_+(f) \geq n^k/t$ where $t = (8k^2\mathrm{e}^k \log n)^k$ is at most a constant times $2^{2k} \cdot \log^2 n \leq n$. Since f has $M = \binom{n}{k} \leq n^k$ prime implicants, each of length $m = \binom{k}{2} \leq k^2$, the parameter K in Eq. (5) is at most a constant times $w^m \cdot \log M \leq 2^{k^3} k \log n \leq n \log^2 n$, and Eq. (4) yields that $\mathrm{C}_w(f) \geq \mathrm{C}_+(f)/K - 1$ is at least a constant $c > 0$ times $n^k/n^2 \log^2 n = n^{\Omega(k)}$. □

Corollary 3 (Large cliques)**.** *Let $f = \mathrm{CLIQUE}(n, n/2)$. If $w \leq \epsilon n/\log n$ for a sufficiently small constant $\epsilon > 0$, then $\mathrm{D}_w(f) = \Omega(n)$.*

Proof. Raz and Wigderson [25, Corollary 4.1] have proved that $\mathrm{D}_+(f) = \Omega(n)$. Since f has $M = \binom{n}{n/2} \leq 2^n$ prime implicants, each of length $m = \binom{n/2}{2} \leq n^2$, the logarithm of the parameter K in Eq. (5) is at most a constant times $w \log m + \log \log M = O(w \log n)$. Eq. (4) yields $\mathrm{D}_w(f) \geq \mathrm{D}_+(f) - \log K = \mathrm{D}_+(f) - O(w \log n) = \Omega(n)$, as desired. □

Corollary 4. *If $f = \mathrm{CLIQUE}(n, n/2)$, then $\mathrm{L}_w(f) = 2^{\Omega(n)}$ holds for DeMorgan formulas of negation width $w = o(n/\log n)$.*

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

Corollary 5 (Tardos' function)**.** *There is a monotone Boolean function T_n of n variables such that T_n can be computed by a DeMorgan circuit of polynomial in n size, but $\mathrm{C}_w(T_n) = 2^{\Omega(n^{1/7})}$ holds when the allowed negation width is $w \leq n^{1/7}$.*

Proof. Tardos [32] observed that an efficient algorithm for computing the Lovász theta function, designed by Grötschel, Lovász and Schrijver [11], gives us a monotone Boolean function T_n of $n = \binom{v}{2}$ variables which is computable by DeMorgan circuits of polynomial in n size, and shares common properties with Clique functions sufficient for Alon and Boppana [2] to yield a lower bound $\mathrm{C}_+(T_n) = 2^{\Omega(v/\log v)^{1/3}} = 2^{\Omega(n/\log n)^{1/6}}$. On the other hand, the parameter K in Eq. (5) is exponential in at most a constant times $w \log n \leq n^{1/7} \log n$. So, Eq. (4) immediately yields the claimed lower bound $\mathrm{C}_w(T_n) \geq \mathrm{C}_+(T_n)/K - 1 = 2^{\Omega(n^{1/7})}$ for circuits of negation width $w = n^{1/7}$. □

Remark 4. Note that the total number N of variables in each clique function $\mathrm{CLIQUE}(n, k)$ is $N = \binom{n}{2} = \Theta(n^2)$. The highest known lower bound on the monotone circuit complexity of an explicit Boolean function of N variables was proved by Harnik and Raz [13], and is exponential in $(N/\log N)^{1/3}$ Recently, Pitassi and Robere [23] gave an explicit monotone Boolean function f of N variables such that $\mathrm{D}_+(f) = \Omega(N)$. The lower bound in Eq. (4) 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 $\mathrm{L}_w(f) = 2^{\Omega(N)}$ on the size of DeMorgan *formulas* of negation width $w = \epsilon N$. Note that the ultimative 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.

Finally, let us give an application of our Theorem 2 concerning DeMorgan circuits of bounded *average* negation width. As we already mentioned in Section 1.3, 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 4).

Fortunately, known lower bound arguments for monotone circuits (see, for example, [16, Chapter 9] and the literature cited herein) work also when the monotone circuits are only required to produce a large enough subset of prime implicants (not necessarily *all* prime implicants). Just to give an example, let us show the following simple consequence of [15, Theorem 3.4].

Lemma 5. *Let $3 \leq k \leq \sqrt{n}$, and let f be a monotone Boolean function which rejects all graphs of chromatic number at most $k-1$, and accepts a $1/K$-fraction of all k-cliques. Then $\mathrm{C}_+(f) \geq 2^{\Omega(\sqrt{k})}/K$.*

Proof. 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. Note that the chromatic number of the complement of every G_h does not exceed q; so, for $q := k-1$, the complements of graphs G_h must be rejected by f. An *s-forest* is a forest with s edges.

As shown in [15, 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 $t \cdot (2s)^{2r}$ r-cliques, a family of $t \cdot (2r)^{2s}$ s-forests, and a set E of r^2 edges such that at least one of the following two assertions holds:

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

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

Every r-clique is contained in exactly $\binom{n-r}{k-r}$ k-cliques. So, under the first alternative (1), the size t of the circuit must be at least $\binom{n}{k}/K$ divided by $(2s)^{2r}\binom{n-r}{k-r}$, which is at least $(n/4ks^2)^r/K$. On the other hand, out of all q^n possible q-colorings h of the vertices of K_n, at most q^{n-l} of the graphs G_h can contain a *fixed* forest with l edges. This is directly shown in the proof of Theorem 3.4 in [15], but also follows from the fact that random q-coloring colors two vertices by the same color with probability $1/q$, and these events are independent for edges in a *forest*. So, under the second alternative (2), the size t of the circuit must be at least $q^n - r^2 \cdot q^{n-1} = q^n(1-r^2/q)$ divided by $(2r)^{2s}q^{n-s}$ which, for any $r \leq \sqrt{q/2}$, is at least $\frac{1}{2}(k/4r^2)^s$.

By taking the parameters $r := \lfloor\sqrt{k/16}\lfloor$ and $s := \lfloor\sqrt{n/8k}\rfloor$, the first alternative yields a lower bound $t \geq 2^r/K$, 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 $t \geq 2^r/K \geq 2^{\Omega(k)}/K$ follows. □

Corollary 6. *Let $f = \mathrm{CLIQUE}(n,k)$ for $k \leq \sqrt{n}$. 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 5 implies that, for every $K \geq 1$, every monotone circuit K-approximating f requires at least $t = 2^{\Omega(\sqrt{k})}/K$ gates. The length of prime implicants of f is $m = \binom{k}{2}$. So, by taking $K := 8m^{2w} = 2^{o(\sqrt{k})}$, Theorem 2 yields the desired lower bound on the size t of any DeMorgan circuit of average negation width w computing f. □

Let us mention that the aforementioned result [15, Theorem 3.4] holds also for monotone circuits with *unbounded* fanin AND and OR gates. The reduction lemma (Lemma 3) also holds for DeMorgan circuits with unbounded fanin AND and OR gates. So, Corollary 6 holds for DeMorgan circuits with unbounded fanin gates.

References

[1] L.M. Adleman, K.S. Booth, F.P. Preparata, and W.L. Ruzzo. Improved time and space bounds for Boolean matrix multiplication. *Acta Inf.*, 11:61–77, 1978.

[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] M.D. Atkinson and N. Santoro. A practical algorithm for Boolean matrix multiplication. *Inf. Process. Lett.*, 29:37–38, 1988.

[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] R.P. Brent, D.J. Kuck, and K. Maruyama. The parallel evaluation of arithmetic expressions without divisions,. *IEEE Trans. Computers*, C-22:523–534, 1973.

[7] 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.

[8] 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.

[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] M. Grigni and M. Sipser. Monotone separation of logarithmic space from logarithmic depth. *J. Comput. Syst. Sci.*, 50(3):433–437, 1995.

[11] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[12] 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.

[13] D. Harnik and R. Raz. Higher lower bounds on monotone size. In *Proc. 32nd Ann. ACM Symp. on Theory of Computing*, pages 378–387, 2000.

[14] 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.

[15] S. Jukna. Combinatorics of monotone computations. *Combinatorica*, 19(1):65–85, 1999. Preliminary versions in: ECCC Report Nr. 26, 1996, and in Proc. of 12th Ann. IEEE Conf. on Comput. Complexity. 1997, pp. 223-238.

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

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

[18] 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. 1978. (In Russian).

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

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

[21] A. Lingas. Small normalized boolean circuits for semi-disjoint bilinear forms require logarithmic conjunction-depth. In *Proc. of 33rd Comput. Complexity Conf.*, volume 102 of *LIPIcs*, pages 26:1–26:10, 2018. Extended version in: ECCC Report Nr. 108, 2018.

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

[23] 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.

[24] 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.

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

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

[27] 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.

[28] A.A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990.

[29] B. Rossman. Correlation bounds against monotone NCˆ1. In *Proc. of 30th Comput. Complexity Conf.*, volume 33 of *LIPIcs*, pages 392—411, 2015.

[30] 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.

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

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

[33] I. Wegener. Relating monotone formula size and monotone depth of Boolean functions. *Infrom. Process. Letters*, 16:41–42, 1983.

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

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

A Motivating examples

Examples of monotone Boolean functions $f(x_1,\ldots,x_n)$, for which already "moderate" negation width allows to substantially reduce the size of monotone circuits computing f, can be constructed using two simple observations: (1) negation width is always at most the total number of input variables, and (2) OR gates cannot increase the negation width. In particular, if $f = f_1 \vee \cdots \vee f_r$ where each f_i is a monotone Boolean function, then $\mathrm{C}_w(f) \leq \mathrm{C}_w(f_1) + \cdots + \mathrm{C}_w(f_r) + r - 1$ holds for every $w \geq 0$.

A general idea is the following. Take a monotone Boolean function $h : \{0,1\}^m \to \{0,1\}$, about which we know that it can be computed by small (unrestricted) DeMorgan circuits, but requires large monotone circuits. An *extension* $f(X)$ of h to $|X| = r \cdot m$ variables is obtained as follows. Split X into disjoint blocks $X_1,\ldots,X_r$ of size $|X_i| = m$, and let $f(X) = h(X_1) \vee h(X_2) \vee \cdots \vee h(X_r)$.

Then $\mathrm{C}_+(f) \geq \mathrm{C}_+(h)$ clearly holds: having a monotone circuit for f, just set to 0 the variables in all but one block X_i, and the resulting circuit will compute $h(X_i)$. On the other hand, since each of the functions $h(X_i)$ has only m variables, the negation width of any DeMorgan circuit computing h is trivially at most m. So, $\mathrm{C}_w(h) \leq \mathrm{C}(h)$ holds for $w = m$; here, $\mathrm{C}(h)$ is the *unrestricted* DeMorgan circuit complexity of h. Since OR gates do not increase the negation width, we have the upper bound $\mathrm{C}_w(f) \leq r \cdot \mathrm{C}_w(h) + r \leq r \cdot \mathrm{C}(h) + r$ for circuits of negation width $w = m$. So, the monotone vs. bounded negation width gap $\mathrm{C}_+(f)/\mathrm{C}_w(f)$ on the function f is at least about the monotone vs. non-monotone gap $\mathrm{C}_+(h)/\mathrm{C}(h)$ on the initial function h.

A.1 Super-polynomial gaps

Example 4 (Logical permanent). The *logical permanent* function Per_m is a monotone Boolean function of m^2 variables which takes a Boolean $m \times m$ matrices Y as inputs, and outputs 1 if and only if 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 if and only if $\mathrm{Per}_m(Y) = 1$ 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 [27], 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 [14]. 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 $r^2 \cdot m^{O(1)} = n^{O(1)}$ and negation width $w \le m^2 = n^{2\epsilon}$.

Example 5 (Tardos' function). Let $0 < \epsilon < 1$ be an arbitrarily small constant, and assume for simplicity that both $m = n^\epsilon$ and $r = n^{1-\epsilon}$ are integers. As we already mentioned in the proof of Corollary 5, Tardos [32] exhibited a monotone Boolean function T_m of $m = \binom{v}{2}$ variables which be computed by a DeMorgan circuit of polynomial in m size, but the monotone circuit complexity of T_m is exponential in $(v/\log v)^{1/3} = m^{\Omega(1)}$. Let f_n be an extension of T_m on $n = r \cdot m$; hence, f_n is the OR of r copies of T_m each on a distinct set of m variables. Then the monotone circuit complexity of f_n is also exponential in $m^{\Omega(1)} = n^{\Omega(1)}$, but the function f_n can be computed by a DeMorgan circuit of size $r \cdot n^{O(1)} = n^{O(1)}$ if the negation width $w = m$ $(= n^\epsilon)$ is allowed.

These two examples show that the size of monotone circuits (DeMorgan circuits of negation width $w = 0$) can be substantially (even super-polynomially) reduced by allowing moderate negation width $w = n^\epsilon$. Our next two examples (of the triangle function and threshold functions) show that non-trivial savings are also possible for monotone Boolean functions that *have* small (polynomial) monotone circuits.

A.2 The triangle function

It is known that every monotone circuit computing the triangle function $\Delta_n = \mathrm{CLIQUE}(n, 3)$ must have about $n^3/\log^3 n$ gates [2, Lemma 3.14]. Corollary 1 shows that any DeMorgan circuit of negation width $w = n^\epsilon$ for a constant $\epsilon > 0$ computing Δ_n must have at least $\Omega(n^{3-4\epsilon})$ gates. Our goal is to show that this lower bound is not too far from an optimal one: for every constant $\epsilon > 0$, the function Δ_n *can* be computed by a DeMorgan circuit of negation width $w = n^\epsilon$ using a sub-cubic number $O(n^{3-\epsilon/4})$ of gates.

To this end, we first consider a *multi-output* "cousin" of the triangle function—the *Boolean matrix multiplication* operator $\mathrm{BMM}(n) : \{0,1\}^{2n^2} \to \{0,1\}^{n^2}$. This operator takes two $n \times n$ Boolean matrices $X = (x_{i,j})$ and $Y = (y_{i,j})$ as inputs, and computes n^2 monotone Boolean functions $f_{i,j} = \bigvee_{k=1}^{n} x_{i,k} y_{k,j}$. Note that now, instead of just one output gate, every circuit computing $\mathrm{BMM}(n)$ has n^2 output gates. The negation width of such a (multi-output) circuit is just the maximum negation width of its sub-circuits computing the functions $f_{i,j}$, each of $2n$ variables.

Fast algebraic algorithms for arithmetic matrix multiplication [31, 9, 35] yield circuits over $\{\vee, \wedge, \neg\}$ for the $n \times n$ Boolean matrix product with $O(n^\omega)$ gates, where ω is the so-called matrix multiplication exponent; after the Strassen [31] breakthrough algorithm showed that $\omega < 2.807$, this exponent was pushed down by Vassilevska Williams [35] and Le Gall [9] to $\omega < 2.373$. This can be used to show that the circuit complexity of $\mathrm{BMM}(n)$ remains *sub-cubic* also when the negation width of circuits is lowered from the trivial $w = 2n$ (unrestricted circuits) to $w = n^\epsilon$ for an arbitrarily small constant $\epsilon > 0$.

Claim 1. *For every $0 < \epsilon \le 1$, the operator $\mathrm{BMM}(n)$ can be computed by a DeMorgan circuit of negation width $w = n^{2\epsilon}$ and size $O(n^{3-\epsilon/2})$.*

Proof. We use the standard "divide and conquer" idea to reduce the multiplication of large matrices to the multiplication of matrices of smaller dimension. This idea was used many times to speed up matrix multiplication; see, for example, Adleman et al. [1], or Atkinson and Santoro [4]. The idea was also recently used in [21, Proposition 12] to show that $\mathrm{BMM}(n)$ can be computed by DeMorgan circuits of sub-cubic size when the allowed AND-depth is smaller than $\log n$.

Set $m := \frac{1}{2}n^{\epsilon}$, and assume (for the sake of simplicity) that both m and $r := n/m$ are integers. Partition the given $n \times n$ matrices X and Y into disjoint $m \times m$ submatrices $X_{u,v}$ and $Y_{u,v}$, for $u, v = 1, \ldots, r$. The corresponding $m \times m$ submatrix $Z_{u,v}$ of the product matrix $Z = X \cdot Y$ is then $Z_{u,v} = \bigvee_{w=1}^{r} X_{u,w} \cdot Y_{w,v}$, where the OR is componentwise. Using fast matrix multiplication, we can compute each of the r^3 matrix products $X_{u,w} \cdot Y_{w,v}$ by a DeMorgan circuit of size $O(m^{\omega})$. Since each such circuit has only $2m^2$ input variables (entries of matrices $X_{u,w}$ and $Y_{w,v}$), the negation width of each of these circuits is trivially at most $2m^2 = n^{2\epsilon}$. Using additional rn^2 OR gates, we can then compute all n^2 entries of the product matrix Z. 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 $w = n^{2\epsilon}$. Since $r = n/m$ with $m = \frac{1}{2}n^{\epsilon}$, and since $3 - \omega \geq 1/2$, the size of the resulting circuit is at most a constant times $r^3 m^{\omega} + rn^2 = n^3/m^{3-\omega} + n^3/m \leq 2n^3/\sqrt{m} \leq 3n^{3-\epsilon/2}$, as desired. □

Claim 2. *For every $0 < \epsilon \leq 1$, the triangle function $f = \mathrm{CLIQUE}(n, 3)$ can be computed by a DeMorgan circuit of negation width $w = n^{2\epsilon}$ and size $O(n^{3-\epsilon/2})$.*

Proof. 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 product $Z = Y^2$ of matrix Y with itself. Note that, for every $i \neq j$, $z_{i,j} = 1$ if and only if there is an $l \notin \{i, j\}$ such that $x_{i,l} = x_{l,j} = 1$. So, $f = \bigvee_{i<j} z_{i,j} \cdot x_{i,j}$.

By Claim 1, we know that the entries $z_{i,j}$ of the matrix Z can be simultaneously computed by a DeMorgan circuit of negation width $w = n^{2\epsilon}$ and size $O(n^{3-\epsilon/2})$. So, the triangle function f itself can be computed by a DeMorgan circuit of size $O(n^{3-\epsilon/2} + n^2) = O(n^{3-\epsilon/2})$ and the same negation width w: neither the additional OR gates nor taking the ANDs $z_{i,j} \cdot x_{i,j}$ of the $z_{i,j}$ with (not negated) variables $x_{i,j}$ can increase the negation width. □

A.3 Threshold functions

Recall that the threshold-k function Th_k^n accepts a Boolean input of length n if and only if it contains at least k ones. The smallest known monotone circuits for Th_k^n have size of order $n \log k$ (see, for example, [19]). On the other hand, we will now show that Th_k^n can be computed by a DeMorgan circuit of *linear* size $O(n)$ if negation width $w = k^3$ is allowed.

Claim 3. *If $w = k^3$, then $\mathrm{C}_w(\mathrm{Th}_k^n) = O(n)$.*

Proof. For the sake of simplicity of argumentation, assume that the number of variables n is divisible by the parameter $s \geq k$ (to be chosen latter). Divide the sequence X of $|X| = n$ Boolean variables into $m := n/s$ consecutive segments $X_1, \ldots, X_m$ of length s, and let $Q_l^j = \mathrm{Th}_l^s(X_j)$ be the threshold-l function on the s variables in the jth segment.

It is well known (see, for example, [34, Sect. 3.4]) that all functions $\mathrm{Th}_1^n, \mathrm{Th}_2^n, \ldots, \mathrm{Th}_n^n$ can be simultaneously computed by a (non-monotone) DeMorgan circuit of size $O(n)$. So, for every $j = 1, \dots, m$, all the functions $Q^j_0, Q^j_1, \dots, Q^j_k$ can be simultaneously computed by a DeMorgan circuit of size $O(s)$. It follows that all functions Q^j_l for $j = 1, \dots, m$ and $l = 1, \dots, k$ can be simultaneously computed by a DeMorgan circuit of size at most a constant times $s \cdot (n/s) = n$. We now use a simple dynamic program to compute all the Boolean functions P^j_l such that $P^j_l = 1$ if and only if there are at least l ones in the first j segments.

As basis functions we take $P^j_0 = Q^j_0 = 1$ (constant 1 functions) for all $j = 1, \dots, m$, $P^1_l = Q^1_l$ for all $l = 1, \dots, k$, and construct a DeMorgan circuit C using the recurrences

$$P^j_l = \bigvee_{r=0}^{l} P^{j-1}_{l-r} \wedge Q^j_r \,. \tag{6}$$

It is easy to see that the whole input sequence contains at least k ones iff $P^m_k = 1$. For the jth segment, we account $O(k^2)$ additional gates implementing the recurrences for P^j_l. Hence, the size of the DeMorgan circuit C computing P^m_k is at most a constant times $mk^2 = (n/s)k^2$.

Induction on the number j on segments shows that for every (fixed) j and all $l = 1, \dots, k$, the negation width of the sub-circuits computing the functions P^j_l does not exceed ls; hence, the negation width of the entire circuit is at most ks. But let us give a non-inductive proof explicitly showing the *form* of the terms produced by the entire circuit.

Namely, by expanding the recursion (6), we see that P^m_k is computed as the OR of ANDs

$$Q^1_{r_1}(X_1) \wedge Q^2_{r_2}(X_2) \wedge \cdots \wedge Q^m_{r_m}(X_m) \tag{7}$$

over all sequences $r_1, \dots, r_m$ of nonnegative integers satisfying $r_1 + \cdots + r_m = k$; recall that $Q^j_0 = 1$ for all j. Since at most k of the r_js in each such sequence can be nonzero, at most k of the functions $Q^j_{r_j}$ in Eq. (7) can be not constant 1 functions. So, every term produced by the circuit C is of the form $q = \bigwedge_{j \in J} q_j$ for some subset $J \subseteq [m]$ of size $|J| \le k$, where each q_j is a (not necessarily nonzero) term containing variables or their negations only from the jth segment X_j. So, if q is a nonzero term, then it can have at most $\sum_{j \in J} |X_j| \le ks$ distinct literals and, hence also at most ks distinct negated variables. In particular, this means that all nonzero terms produced by the circuit C including the extensions of prime implicants of the computed by C function P^m_k, have at most ks distinct negated variables.

So, the constructed circuit C for the threshold-k function Th^n_k has negation width $w \le ks$ and size of order $(n/s)k^2$. It remains to take the segment-length $s = k^2$. This gives us a circuit of linear size $O(n)$ and negation width at most k^3, as desired. $\square$

A.4 Cancellations

We will now give and example which *explicitly* demonstrates how cancellations can reduce the number of gates.

Let $g(X)$ be a Boolean function of $n = m^2$ variables arranged into an $m \times m$ matrix $X = (x_{i,j})$. The function accepts X if and only if every row of X has at least one 1 and every column of X has at least one 0. To make the function monotone, just take an OR $f(X) = g(X) \vee \mathrm{Th}^n_{m+1}(X)$ of g with the threshold-$(m+1)$ function. So, $f(X)$ accepts X if and only if either X has $|X| > m$ ones, or $|X| = m$ and $g(X) = 1$ holds. A trivial *monotone* formula for f is $F_+ = (G \wedge H) \vee T$, where

$$G(X) = \bigwedge_{i=1}^{m} \Big(\bigvee_{j=1}^{m} x_{i,j} \Big) , \qquad H(X) = \bigvee_{j_1 < j_2} \Big(\bigvee_{i=1}^{m} x_{i,j_1} \Big) \wedge \Big(\bigvee_{i=1}^{m} x_{i,j_2} \Big) ,$$

and $T(X)$ is a minimal monotone formula computing Th^n_{m+1}. Note that $G(X)=1$ iff every row of X has at least one 1, and $H(X)=1$ iff not all 1s of X are in the same column. On the other hand, the function f can also be computed by a (also trivial but simpler) formula $F=(G\wedge H')\vee T$ of negation width $w=m=\sqrt{n}$, where

$$H'(X)=\bigwedge_{j=1}^{m}\left(\bigvee_{i=1}^{m}\overline{x}_{i,j}\right)=\bigvee_{i=1}^{m}\bigwedge_{j=1}^{m}\overline{x}_{i,j}\,.$$

The formula $H'(X)$ accepts a matrix X if and only if every its column has at least one zero. The size of the monotone formula F_+ is of the order $m^3+s=n^{3/2}+s$, where s is the size of the formula T, whereas the size of the formula F is only of the order $m^2+s=n+s$. The set of terms produced by the formula F consists of all terms produced by the formula T and all terms of the form $x_{1,j_1}x_{2,j_2}\cdots x_{m,j_m}\cdot\overline{x}_{i_1,1}\overline{x}_{i_2,2}\cdots\overline{x}_{i_m,m}$, a lot of which are zero terms.

B Negation-width and communication

Let $f:\{0,1\}^n\to\{0,1\}$ be a monotone Boolean function, and $w\geq 0$ be and integer. Let $d_+=\mathrm{D}_+(f)$ be the minimum depth of a monotone circuit computing f, and $d=\mathrm{D}_w(f)$ be the minimum depth of a DeMorgan circuit of negation width w computing f. Lemma 3 gives an upper bound $d_+\leq d+\log K$, where K is a constant times $\min\{w^m,m^w\}\cdot\log|PI(f)|$, and m is the maximum length of a prime implicant of f.

In this section, we give an entirely different proof of a slightly weaker upper bound $d_+\leq d+w\cdot\lceil\log(n+1)\rceil$ using the Karchmer–Wigderson communication complexity arguments [17].

As shown by Karchmer and Wigderson [17], 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. We call such a position a *separating position* for the pair (a,b). Note that, since f is monotone, such a position always exists.

Now take a DeMorgan circuit C 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 [17], 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 $(a,b)\in f^{-1}(1)\times f^{-1}(0)$.

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 C is bounded by w, there is a term $r=\bigwedge_{i\in I}\overline{x}_i$ with $|I|\leq w$ such that $p\cdot r\in T(C)$ 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 the computed by C function f is monotone, we have $f(b')=0$.
5. The players now replace by zeros all negated input gates $\overline{x}_i$ with $i\notin I$, and consider the resulting circuit C'.

6. Since no negated literal of the term $p \cdot r$ was set to 0, this term belongs also to the set $T(C')$ of terms produced by the new circuit. So, since $p \cdot r(a') = 1$, the circuit C' accepts vector a'. On the other hand, since the original circuit C rejected vector b, and we have only replaced some input gates by zeros, the circuit C' rejects vector b'.
7. So, the players can now run the standard Karchmer–Wigderson protocol on the pair (a', b') using the circuit C' (see [17, Lemma 2.1]). After communicating at most d bits (the depth of C' can only be smaller than d), they will arrive at some input literal z of the circuit C' such that $z(a') = 1$ and $z(b') = 0$. The literal z is either an non-negated variable x_i, or a negated variable $\overline{x}_i$ for some $i \in I$: the circuit C' has no other 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 (non-negated) 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, i is the desired separating position for the input pair (a, b). □