Size-Treewidth Tradeoffs for Circuits Computing the Element Distinctness Function*

Mateus de Oliveira Oliveira

Institute of Mathematics – Czech Academy of Sciences, Zitná 25, 115 67 Praha 1, Czech Republic
mateus.oliveira@math.cas.cz

Abstract

In this work we study the relationship between size and treewidth of circuits computing variants of the element distinctness function. First, we show that for each n, any circuit of treewidth t computing the element distinctness function $\delta_n : \{0,1\}^n \to \{0,1\}$ must have size at least $\Omega(\frac{n^2}{2^{O(t)} \log n})$. This result provides a non-trivial generalization of a super-linear lower bound for the size of Boolean formulas (treewidth 1) due to Nečiporuk. Subsequently, we turn our attention to read-once circuits, which are circuits where each variable labels at most one input vertex. For each n, we show that any read-once circuit of treewidth t and size s computing a variant $\tau_n : \{0,1\}^n \to \{0,1\}$ of the element distinctness function must satisfy the inequality $t \cdot \log s \geq \Omega(\frac{n}{\log n})$. Using this inequality in conjunction with known results in structural graph theory, we show that for each fixed graph H, read-once circuits computing τ_n which exclude H as a minor must have size at least $\Omega(n^2/\log^4 n)$. For certain well studied functions, such as the *triangle-freeness* function, this last lower bound can be improved to $\Omega(n^2/\log^2 n)$.

1998 ACM Subject Classification F.2.3 Tradeoffs Between Complexity Measures

Keywords and phrases non-linear lower bounds, treewidth, element distinctness

Digital Object Identifier 10.4230/LIPIcs.STACS.2016.56

1 Introduction

The problem of explicitly defining a function in NP which requires super-linear circuit size has proven to be notoriously hard. Currently, the best known lower bound for a function in NP is of the order[1] of $3n - o(1)$ for circuits with arbitrary fan-in-2 gates [4, 8], and of the order of $5n - o(1)$ for circuits with gates from the binary De-Morgan basis [18, 21]. In the particular case of boolean formulas, Nečiporuk proved an $\Omega(n^2/\log n)$ lower bound for the size of boolean formulas over the full binary basis computing the n-bit element distinctness function [24]. Intuitively, the element distinctness function $\delta_n : \{0,1\}^n \to \{0,1\}$ takes as input a sequence of m numbers $s_1, s_2, \ldots, s_m \in \{1, \ldots, m^2\}$ encoded as binary strings with $2 \log m$ bits, and returns 1 if and only if all numbers in this sequence are distinct. Remarkably, Nečiporuk's lower bound has resisted improvements during the last four decades, and remains the strongest known lower bound for the size of formulas over the full binary basis. In the restricted setting of formulas over the De-Morgan basis, a size lower bound of $n^{3-o(1)}$ was obtained by Håstad [15] using different techniques.

In this work, we consider the problem of proving circuit size lower bounds for circuits of low treewidth. During the past decade a considerable amount of research has been devoted to the

* This work was supported by the European Research Council, grant number 339691, in the context of the project Feasibility, Logic and Randomness (FEALORA).

[1] Recently, this lower bound was improved to $(3 + 1/86)n - o(n)$ [10].

33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016).
Editors: Nicolas Ollinger and Heribert Vollmer; Article No. 56; pp. 56:1–56:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

study of the computational power and the combinatorial properties of circuits parameterized by treewidth [1, 2, 6, 11, 12, 16, 19]. In our first result we generalize Nečiporuk's lower bound to the context of circuits of low treewidth.

▶ **Theorem 1.** *Let $\mathcal{C}$ be a circuit of treewidth t computing the element distinctness function δ_n. Then $\mathcal{C}$ has size $\Omega(\frac{n^2}{2^{O(t)}\log n})$.*

Here the size of a circuit $\mathcal{C}$ is defined as its wire-complexity, i.e., the total number of edges in $\mathcal{C}$. Therefore, our lower bound holds for circuits containing unbounded fan-in AND and OR gates, and more generally, unbounded fan-in associatively constructible gates, which we will define in Section 2. Theorem 1 generalizes Nečiporuk's non-linear lower bound, from the context of Boolean formulas (that is to say, circuits of treewidth 1) to the context of circuits of low treewidth. In particular, our result implies an $\Omega(n^2/\log n)$ lower bound for the size of circuits whose underlying undirected graph belongs to several interesting classes, such as trees (treewidth at most 1), TTSP series-parallel graphs (treewidth at most 2), outer-planar graphs (treewidth at most 2), Halin graphs (treewidth at most 3), k-outerplanar graphs for fixed k (treewidth at most $O(k)$), etc. Additionally, Theorem 1 implies non-linear lower bounds even for circuits of treewidth $o(\log n)$.

It is worth comparing our result with another prominent restricted family of circuits for which no non-linear lower bound is known, namely, circuits whose underlying graph belongs to the class of Valiant Series-Parallel graphs [31]. We refer to [7] for a clear definition of this class. It can be shown that the class of Valiant-series-parallel graphs strictly contains the class of TTSP-series-parallel graphs (which have treewidth 2). Nevertheless, Valiant-series-parallel graphs are incomparable with graphs of treewidth k, for $k \geq 3$. On the one hand, there are Valiant-series-parallel graphs of treewidth at least k for every $k \in \mathbb{N}$. For instance, the $k \times k$ grid-graph is Valiant-series-parallel but has treewidth k. On the other hand, it is easy to construct graphs of treewidth 3 which are not Valiant-series-parallel. Proving a non-linear lower bound for Valiant-series-parallel circuits remains a major open problem in circuit complexity [25, 28].

Next, we turn our attention to *read-once* circuits, which are circuits where each variable labels at most one input vertex. These circuits have also been known in the VLSI literature as *semilective* circuits [17]. Read-once circuits parameterized by treewidth have been studied by the SAT-solving and proof-complexity communities. Part of the interest in these circuits is due to the fact that the satisfiability problem for read-once circuits size s and treewidth t can be solved in time $2^{O(t)} \cdot s^{O(1)}$ [1, 2, 6, 12]. Questions related to the design of optimal VLSI circuits have motivated the study of the complexity of *planar* read-once circuits computing explicit functions (i.e. functions in NP). Within this line of research, quadratic lower bounds have been obtained for the size of planar read-once circuits computing both multiple-output functions [22] and single-output functions [29]. We contrast these quadratic lower bounds with the fact that for multilective planar circuits, i.e., planar circuits in which variables can label arbitrarily many input gates, the best known lower bounds are of the order of $O(n \log n)$ for single-output functions and of the order of $O(n^{3/2})$ for multiple-output functions [30].

In this work we introduce the *symmetric non-deterministic state complexity* (symmetric-NSC) of a Boolean function, a complexity measure that is lower-bounded by the size of the smallest read-once oblivious branching program computing the function in question. We show that if $\mathcal{C}$ is a read-once circuit of size s and treewidth t computing a function $f_n : \{0,1\}^n \to \{0,1\}$ of symmetric-NSC $snsc(f_n)$, then $t \cdot \log s \geq \Omega(\log snsc(f_n))$. Using this tradeoff in conjunction with known results from structural graph theory, we show that for

each fixed graph H, read-once H-minor-free[2] circuits computing f_n must have size at least $\Omega\left(\frac{\log^2 snsc(f_n)}{\log^2 n}\right)$. Subsequently, we introduce a variant $\tau_n : \{0,1\}^n \to \{0,1\}$ of the element distinctness function and show that its symmetric-NSC is lower bounded by $2^{\Omega(n/\log n)}$. From these results we have that read-once H-minor-free circuits computing τ_n require size $\Omega(n^2/\log^4 n)$. Near-quadratic lower bounds can also be obtained for the size of read-once H-minor-free circuits computing certain well studied functions, such as the *triangle-freeness* function Δ_n, and the *triangle-parity* function $\bigoplus \text{Clique}_{3,n}$. A result from [9] implies that the symmetric-NSC of these functions is lower-bounded by $2^{\Omega(n)}$. Therefore, read-once H-minor-free circuits computing both Δ_n and $\bigoplus \text{Clique}_{3,n}$ require size $\Omega(\frac{n^2}{\log^2 n})$.

2 Preliminaries

Let Σ be a finite set of symbols. A k-ary gate over Σ is a function $g : \Sigma^k \to \Sigma$. For $k \geq 3$, we say that a k-ary gate g is associatively constructible if there exists an associative operation $\oplus : \Sigma \times \Sigma \to \Sigma$ such that $g(x_1, \ldots, x_k) = x_1 \oplus \ldots \oplus x_k$. Alternatively, we say that g is a $\oplus$-gate of fan-in k. Unbounded fan-in AND and OR gates are clearly associatively constructible. An unbounded fan-in MOD_r gate can be simulated by an associatively constructible gate g that computes the sum of its inputs modulo r, together with a unary gate $g' : \Sigma \to \Sigma$ that returns 0 if this sum is congruent to 0 mod r, and which returns 1 otherwise.

An associatively constructible circuit with n inputs is a directed acyclic graph $\mathcal{C} = (V, E, \mathbf{g})$ where V is a set of vertices, E is a set of directed edges, and $\mathbf{g}$ is a function that labels each vertex $v \in V$ with a symbol from Σ, a gate over Σ, or a variable from $\{x_1, \ldots, x_n\}$. The function $\mathbf{g}$ must satisfy the following conditions:

1. If the in-degree of v is 0, then $\mathbf{g}(v)$ is either an element of Σ or a variable in $\{x_1, \ldots, x_n\}$.
2. If the in-degree of v is k, then $\mathbf{g}(v)$ is a k-ary gate over Σ. Additionally, if $k \geq 3$, then $\mathbf{g}(v)$ is associatively constructible.

Vertices of in-degree 0 are called *inputs*. An input is *initialized* if it is labeled with an element of Σ, and *uninitialized* if it is labeled with a variable. A formula is a circuit whose underlying graph is a tree. We say that a circuit $\mathcal{C} = (V, E, \mathbf{g})$ is *read once* if no two input vertices are labeled with the same variable. Since our circuits may contain associatively constructible gates of unbounded fan-in and unbounded fan-out, we define the size $|\mathcal{C}|$ of a circuit $\mathcal{C}$ as the number of edges in $\mathcal{C}$.

Below, we define the notion of *rooted carving decomposition* of a circuit, a variant of the notion of carving decomposition defined in [27]. If T is a tree, we denote by $nodes(T)$ the set of all nodes of T, and by $leaves(T)$ the set of all leaves of T. For each node $u \in nodes(T)$, we let $T[u]$ denote the subtree of T rooted at u.

▶ **Definition 2** (Carving Decomposition). A *rooted carving decomposition* of a circuit $\mathcal{C} = (V, E, \mathbf{g})$ is a pair (T, γ) where T is a binary tree and $\gamma : leaves(T) \to V$ is a bijection mapping each leaf $u \in leaves(\mathcal{T})$ to a single vertex $\gamma(u) \in V$.

Observe that the internal nodes of a carving decomposition $\mathcal{T}$ are unlabeled. Given a node $u \in nodes(T)$, we let $V(u) = \gamma(leaves(T[u])) = \{\gamma(v) \mid v \in leaves(T[u])\}$ be the image of the leaves of $T[u]$ under γ. For two distinct subsets V_1, V_2 of vertices of a circuit $\mathcal{C}$ we let

[2] We say that a circuit $\mathcal{C}$ is H-minor-free if its underlying undirected graph excludes H as a minor.

$E(V_1, V_2)$ denote the set of edges in G with one endpoint in V_1 and another endpoint in V_2. The width $carw(\mathcal{T})$ of the carving decomposition $\mathcal{T}$ is defined as

$$\max\{|E(V(u), V \backslash V(u))| : u \in nodes(T)\}.$$

The carving width $carw(\mathcal{C})$ of a circuit $\mathcal{C}$ is the minimum width of a carving decomposition of $\mathcal{C}$. The following lemma, whose proof is based on a result from [23], relates carving width and treewidth of a circuit.

▶ **Lemma 3** (From Tree-Decompositions to Carving Decompositions). *Let $\mathcal{C} = (V, E, \mathbf{g})$ be an associatively constructible circuit of treewidth t. There is a circuit $\mathcal{C}'$ of size $|\mathcal{C}'| \leq 2 \cdot |\mathcal{C}|$, maximum degree 3, and carving width at most $3t + 3$ such that $\mathcal{C}$ and $\mathcal{C}'$ compute the same function.*

3 Nečiporuk's Method

In this section we briefly describe Nečiporuk's method for proving non-linear lower bounds on the size of Boolean formulas over the complete binary basis. For our purposes, it will be convenient to divide this method into three steps. Our first main result (Theorem 1) follows from a generalization of Step 1 given below. A complete proof of Nečiporuk's theorem can be found in [20].

Step 1: Let $X = \{x_1, \ldots, x_n\}$ be a set of variables, $f : \{0,1\}^X \to \{0,1\}$ be a Boolean function on X, and $Y \subseteq X$ be a subset of variables of X. We denote by $N_f(Y)$ the number of distinct functions that can be obtained by initializing all variables in $X \backslash Y$ with values in $\{0,1\}$. The first step in the proof of Nečiporuk's theorem consists in providing an upper bound for $N_f(Y)$. If f can be computed by a Boolean formula F, such upper bound can be given in terms of the number of inputs of F labeled with variables in Y.

▶ **Proposition 4.** *Let $f : \{0,1\}^X \to \{0,1\}$ be a function computable by a boolean formula F. Let $Y \subseteq X$ be a subset of variables such that at most l inputs of F are labeled with variables in Y. Then $N_f(Y)$ is at most $2^{O(l)}$.*

Note that if $g : \{0,1\}^Y \to \{0,1\}$ is a function obtained from f by initializing all variables in $X \backslash Y$, then g can be represented by a boolean formula F_g with l uninitialized inputs which is obtained from F by initializing all inputs labeled with variables in $X \backslash Y$. The proof of Proposition 4 follows by noting that the Boolean formula F_g can be simplified into a Boolean formula F'_g also computing the function g, in such a way that F'_g has at most l inputs, all of which are uninitialized, and in which all internal nodes have fan-in 2. This implies that F'_g has at most $l - 1$ internal nodes. Since there are 16 possible Boolean functions of fan-in 2, there are at most 16^{l-1} choices for g.

Step 2: The second step consists in exhibiting an explicit Boolean function with many sub-functions. Intuitively, a function $f : \{0,1\}^X \to \{0,1\}$ has many sub-functions if the quantity $N_f(Y)$ is large for some subsets $Y \subseteq X$ of suitable size. Let $X = \{x_1, \ldots, x_n\}$ be a set of $n = 2m \log m$ distinct variables partitioned into m blocks $Y_1, Y_2, \ldots, Y_m$, where each block Y_i has $2 \log m$ variables. The *element distinctness* function $\delta_n : \{0,1\}^X \to \{0,1\}$ is defined as follows for each assignment $s_1, s_2, \ldots, s_m$ of the blocks $Y_1, Y_2, \ldots, Y_m$ respectively.

$$\delta_n(s_1, s_2, \ldots, s_m) = \begin{cases} 1 & \text{if } s_i \neq s_j \text{ for } i \neq j, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The following lemma states that the element distinctness function defined in Equation 1 has many sub-functions.

▶ **Lemma 5** (See [20], Section 6.5). *Let $\delta_n : \{0,1\}^X \to \{0,1\}$ be the element distinctness function defined in Equation 1, where $|X| = n$ and $X = Y_1 \dot\cup Y_2 \dot\cup \ldots \dot\cup Y_m$ with $|Y_i| = 2 \log m$. Then for each $i \in \{1, \ldots, m\}$, $N_{\delta_n}(Y_i) \geq 2^{\Omega(n)}$.*

Step 3: In the third step, we combine Proposition 4 with Lemma 5 to obtain a non-linear lower bound for the size of Boolean formulas computing the element distinctness function $\delta_n : \{0,1\}^X \to \{0,1\}$ defined in Equation 1. Let F be a Boolean formula computing δ_n. Let l_i denote the number of inputs of F labeled with some variable in Y_i. By Proposition 4, we have that $N_{\delta_n}(Y_i) \leq 2^{O(l_i)}$. On the other hand, by Lemma 5, $N_{\delta_n}(Y_i) \geq 2^{\Omega(n)}$. Combining these two inequalities, we have that

$$2^{O(l_i)} \geq N_{\delta_n}(Y_i) \geq 2^{\Omega(n)}. \tag{2}$$

This implies that $l_i \geq \Omega(n)$. In other words, there are $\Omega(n)$ inputs of F labeled with variables from Y_i. Since there are $m = \Omega(\frac{n}{\log n})$ blocks Y_i, we have that the number of inputs of F labeled with variables in X is at least $\Omega(\frac{n^2}{\log n})$. □

3.1 Generalizing Nečiporuk's Theorem

In this section we will generalize Nečiporuk's non-linear lower bound to the context of circuits of low treewidth (Theorem 1). We call attention to the fact that this lower bound concerns circuits in which each variable can label arbitrarily many input vertices. The following lemma, which generalizes Proposition 4, is the main technical result towards the proof of Theorem 1.

▶ **Lemma 6.** *Let $f : \{0,1\}^X \to \{0,1\}$ be a function computable by a boolean circuit $\mathcal{C}$ of treewidth t. Let $Y \subseteq X$ be a subset of variables such that at most l inputs of $\mathcal{C}$ are labeled with variables in Y. Then $N_f(Y)$ is at most $2^{l \cdot 2^{O(t)}}$.*

The next two subsections will be dedicated to the proof of Lemma 6. Before, we show how Lemma 6 can be used to prove Theorem 1.

Proof of Theorem 1. Let $|X| = n = 2m \log m$, and $Y_1, \ldots, Y_m$ be a partition of the variables in X, where for each i, $|Y_i| = 2 \log m$. Let l_i be the number of inputs of $\mathcal{C}$ labeled with a variable from Y_i. By Lemma 5, $N_{\delta_n}(Y_i) \geq 2^{\Omega(n)}$. On the other hand, by Lemma 6, $N_{\delta_n}(Y_i) \leq 2^{l_i \cdot 2^{O(t)}}$. Therefore, by combining these two inequalities, we have $2^{l_i \cdot 2^{O(t)}} \geq N_{\delta_n}(Y_i) \geq 2^{\Omega(n)}$. This implies that $l_i \geq \Omega(n/2^{O(t)})$. Since there are $m = \Omega(\frac{n}{\log n})$ blocks of variables Y_i, we have that the number of inputs of $\mathcal{C}$ is at least $\frac{n^2}{2^{O(t)} \cdot \log n}$. ◀

3.2 Defining Relations via Constraint Satisfaction Problems

In this section we will introduce some terminology and basic results which will be used in the proof of Lemma 6. Let X be a set of variables. An *assignment* of X is a function $a : X \to \{0,1\}$ that associates with each variable $x \in X$ a value $a(x) \in \{0,1\}$. We let $\{0,1\}^X$ denote the set of all assignments of X. A *relation* over X is any subset $R \subseteq \{0,1\}^X$. We say that each variable $x \in X$ is *constrained* by R. In some places we write $var(R)$ to denote the set of variables constrained by R. If a is an assignment of X, and $Y \subseteq X$, then we let $a|_Y$ denote the *restriction* of a to Y. More precisely, for each $x \in Y$, $a|_Y(x) = a(x)$. We say that an assignment $a \in \{0,1\}^X$ *satisfies a relation* R over $Y \subseteq X$ if $a|_Y \in R$. If

$R \subseteq \{0,1\}^X$ is a relation over X, and $Y \subseteq X$, then the restriction of R to Y is the relation $R|_Y = \{a|_Y \mid a \in R\}$. The following immediate observation says the result of restricting a relation $R \subseteq \{0,1\}^X$ to a subset X' and subsequently to a subset $Y \subseteq X'$ is equivalent to restricting R directly to Y.

▶ **Observation 1.** *Let R be a relation over X and let $Y \subseteq X' \subseteq X$. Then $R|_Y = (R|_{X'})|_Y$.*

Below, we define the notion of *constraint satisfaction problem* over X.

▶ **Definition 7.** A constraint satisfaction problem (CSP) over a set of variables X is a set of relations

$$K = \{R_1, R_2, \ldots, R_r\} \tag{3}$$

where for each $i \in \{1, \ldots, r\}$, R_i is a relation over some subset $X_i \subseteq X$ of variables.

We note that two relations R_i and R_j in K in which $var(R_i) \neq var(R_j)$ are considered to be different. A CSP K over a set of variables X can be used to define a relation $R(K)$ over X. Intuitively, the relation $R(K)$ consists of all assignments over X that satisfy each relation in K.

$$R(K) = \{a \in \{0,1\}^X \mid a|_{X_i} \in R_i \text{ for } i \in \{1, \ldots, r\}.\} \tag{4}$$

Let K be a CSP over a set of variables X and let $S \subseteq K$. We denote by $c(S)$ the set of variables that are simultaneously constrained by some relation in S and some relation $K \backslash S$. We say that $c(S)$ is the *cutset* of S with respect to K. Given a CSP K over a set of variables X, and a subset $Y \subseteq X$, we will deal with the problem of obtaining a CSP K' with less relations than K, but with the property that $R(K)|_Y = R(K')|_Y$. The following simple lemma will be crucial for this goal.

▶ **Lemma 8.** *Let $K = \{R_1, \ldots, R_r\}$ be a CSP over a set of variables X, let $Y \subseteq X$, and $S \subseteq K$ be such that $var(S) \cap Y \subseteq c(S)$. Consider the CSP*

$$K' = (K \backslash S) \cup \{R(S)|_{c(S)}\}.$$

Then $R(K)|_Y = R(K')|_Y$.

3.3 Circuits vs CSPs

In this section we prove Lemma 6. The idea behind the proof is the following. Let $\mathcal{C}$ be a circuit of carving width w computing a function $f : \{0,1\}^Y \to \{0,1\}$. As a fist step, we associate with $\mathcal{C}$ a CSP $K(\mathcal{C})$ over a set of variables $Y \cup \{x_e \mid e \in E\}$. This CSP has the property that $R(K(\mathcal{C}))|_Y$ consists precisely of those assignments that cause $\mathcal{C}$ to evaluate to 1. In a second step, we use the fact that $\mathcal{C}$ has carving width w to obtain a new CSP K' such that each relation in K' constrains at most w variables, and such that the number of relations in K' is proportional to the number of uninitialized inputs of $\mathcal{C}$. This new CSP K' has the property that $R(K')|_Y = R(K(\mathcal{C}))|_Y$. Finally, if we are given a function $f : \{0,1\}^X \to \{0,1\}$ and a subset $Y \subseteq X$ of variables, then we will have that there are at most $2^{2^{O(w)}}$ distinct functions arising by restricting all variables in $X \backslash Y$ to values in $\{0,1\}$.

▶ **Definition 9** (CSP Derived from a Circuit)**.** Let $\mathcal{C} = (V, E, \mathbf{g})$ be a circuit whose inputs are labeled with variables from Y. We let $K(\mathcal{C}) = \{R_v \mid v \in V\}$ be the CSP over the variables $Y \cup \{x_e \mid e \in E\}$ which is defined as follows.

1. If v is an input vertex labeled by $\mathbf{g}$ with a variable x, and v is the source of edges $e_1, \ldots, e_k$, then R_v is a relation over the variables $Y_v = \{x, x_{e_1}, \ldots, x_{e_k}\}$, and an assignment $a : Y_v \to \Sigma$ is in R_v if and only if

   $$a(x) = a(x_{e_1}) = \ldots = a(x_{e_k}).$$

2. If v is an internal vertex labeled with a gate $\mathbf{g}(v)$, v is the target of edges $e_1, \ldots e_k$, and v is the source of edges $e'_1, \ldots, e'_{k'}$, then R_v is a relation over the variables $Y_v = \{x_{e_1}, \ldots, x_{e_k}, x_{e'_1}, \ldots, x_{e'_{k'}}\}$, and an assignment $a : Y_v \to \Sigma$ is in R_v if and only if

   $$\mathbf{g}(v)(a(x_{e_1}), \ldots, a(x_{e_k})) = a(x_{e'_1}) = \ldots = a(x_{e'_{k'}}).$$

3. If v is the output vertex of $\mathcal{C}$, and v is the target of edges $e_1, \ldots, e_k$, then R_v is a relation over the variables $Y_v = \{x_{e_1}, \ldots, .x_{e_k}\}$ and $a : \{x_{e_1}, \ldots, x_{e_k}\} \to \Sigma$ is in R_v if and only if

   $$\mathbf{g}(v)(x_{e_1}, \ldots, x_{e_k}) = 1.$$

Intuitively, the variables Y are input variables of the circuit $\mathcal{C}$, while the variables $\{x_e \mid e \in E\}$ are used to keep track of the evaluation of the circuit $\mathcal{C}$ when the variables in Y are initialized. The relation $R(K(\mathcal{C}))$ associated with the CSP $K(\mathcal{C})$ contains all assignments of $Y \cup \{x_e \mid e \in E\}$ which encode an initialization of the input variables together with an evaluation of the gates of the circuits which evaluate to 1. If we restrict the relation $R(K(\mathcal{C}))$ to the variables in Y, then we recover precisely the set of assignments that cause $\mathcal{C}$ to evaluate to 1.

▶ **Observation 2.** *Let $\mathcal{C}$ be a circuit computing a function $f : \{0,1\}^Y \to \{0,1\}$. Let $K(\mathcal{C})$ be the CSP associated with $\mathcal{C}$. Then $R(K(\mathcal{C}))|_Y = \{a \in \{0,1\}^Y \mid f(a) = 1\}$.*

We note that the number of relations in $R(K(\mathcal{C}))$ is precisely the number of gates of $\mathcal{C}$, and therefore there is no a priori correspondence between the number of relations in $R(K(\mathcal{C}))$ and the number of inputs of $\mathcal{C}$ labeled by variables in Y. The following theorem says that if $\mathcal{C}$ is a circuit of carving width w then one can construct a CSP K whose size is proportional to the number of inputs of $\mathcal{C}$ labeled with variables in Y, in such a way that the number of variables constrained by each relation in K is proportional to w, and such that $R(K)|_Y = R(K(\mathcal{C}))|_Y$.

▶ **Theorem 10** (CSP Reduction)**.** *Let $\mathcal{C}$ be a circuit of carving width w computing a function $f : \{0,1\}^Y \to \{0,1\}$. Let $l \geq |Y|$ be the number of inputs of $\mathcal{C}$ labeled with variables in Y. Then there exits a CSP $K = \{R_1, \ldots, R_k\}$ with $k \leq 3 \cdot l$ such that for each $i \in \{1, \ldots, k\}$, R_i constrains at most $2 \cdot w$ variables and such that $R(K)|_Y = \{a \in \{0,1\}^Y \mid f(a) = 1\}$.*

It is worth noting that the CSP K in Theorem 10 is obtained from the CSP $K(\mathcal{C})$ by applying several non-trivial simplification steps. Before proving Theorem 10 we show how this theorem can be used to prove Lemma 6.

Proof of Lemma 6. Let $f : \{0,1\}^X \to \{0,1\}$ be a boolean function which is computable by a circuit $\mathcal{C}$ of treewidth t. By Lemma 3, there exists a circuit $\mathcal{C}'$ of size $|\mathcal{C}'| \leq 2 \cdot |\mathcal{C}|$ such that $\mathcal{C}'$ of maximum degree 3 and carving width at most $w = 3(t+1)$ such that $\mathcal{C}'$ computes f. Now let $Y \subseteq X$. Then each initialization of the variables in $X \backslash Y$, gives rise to a circuit $\mathcal{C}''$ in which all l uninitialized inputs are labeled with variables in Y. By Theorem 10, there is a CSP K with at most $3 \cdot l$ relations, such that each relation R in K constrains at most $2 \cdot w = 6(t+1)$ variables, and such that $R(K)|_Y$ is precisely the set of assignments of the variables in Y which cause the circuit $\mathcal{C}''$ to evaluate to 1. This implies that there are at most $2^{2^{6(t+1)} \cdot 3 \cdot l}$ possible distinct functions arising from the restrictions of variables not in Y. ◀

In the remainder of this section, we prove Theorem 10. Let $\mathcal{C} = (V, E, \mathbf{g})$ be a circuit of carving-width w computing a function $f : \Sigma^Y \to \Sigma$. Let $K = K(\mathcal{C}) = \{R_v \mid v \in V\}$ be the CSP associated with $\mathcal{C}$. We note that $Y \cup \{x_e \mid e \in E\}$ is the set of variables constrained by relations in K. We say that a relation R_v is a Y-relation if R_v constrains some variable in Y. Let (T, γ) be a carving decomposition of (V, E). For a node u of T we let $\mathit{leaves}(T[u], Y)$ denote the set of leaves u' of $T[u]$ such that the relation $R_{\gamma(u')}$ is a Y-relation. We say that a node $u \in \mathit{nodes}(T)$ is a Y-node if u is either a leaf such that $R_{\gamma(u)}$ is a Y-relation, or if u is an internal node $u \in \mathit{nodes}(T)$ such that $\mathit{leaves}(T[u.l], Y) \neq \emptyset$ and $\mathit{leaves}(T[u.r], Y) \neq \emptyset$. If u is a Y-node, then we say that a node $u' \neq u$ is the Y-parent of u if u' is the ancestor of u at minimal distance from u with the property that u' is itself a Y-node. We let $\mathit{nodes}(T, Y)$ denote the set of all Y-nodes of T.

▶ **Lemma 11.** $|\mathit{nodes}(T, Y)| = 2 \cdot |\mathit{leaves}(T, Y)| - 1$.

Intuitively, the idea of the proof of Lemma 11 consists in showing that the set of all Y-nodes of T induces a binary tree. Since a binary tree with $|\mathit{leaves}(T, Y)|$ leaves has $|\mathit{leaves}| - 1$ internal nodes, the total number of Y-nodes is $2 \cdot |\mathit{leaves}(T, Y)| - 1$. Now let $T' = T \backslash \mathit{nodes}(T, Y)$ be the forest which is obtained by deleting from T all of its Y-nodes. We note that the number of connected components in the forest T' is at most $|\mathit{nodes}(T, Y)| = 2|\mathit{leaves}(T, Y)| - 1$. We let $T_1, \ldots, T_k$, for $k \leq |\mathit{nodes}(T, Y)|$ be the connected components of T'. For each $i \in \{1, \ldots, k\}$, let $S_i = \{R_v \mid \exists u \in \mathit{leaves}(T_i),\ \gamma(u) = v\}$ be the sub-CSP of $K(\mathcal{C})$ formed by the relations associated to vertices of $\mathcal{C}$ that label the leaves of the connected component T_i. Let $c(S_i) = \mathit{var}(K(\mathcal{C}) \backslash S_i) \cap \mathit{var}(S_i)$ be the cut-set of S_i with respect to $K(\mathcal{C})$. In other words, $c(S_i)$ is the set of variables that are constrained by some relation in S_i, and another relation in $K \backslash S_i$. Note that $c(S_i) \cap Y = \emptyset$. The fact that (T, γ) is a carving decomposition of $\mathcal{C}$ of width w implies that the number of variables in $c(S_i)$ is at most $2 \cdot w$. Let $R_i = R(S_i)|_{c(S_i)}$. Then we define our CSP as follows.

$$K = \left(K(\mathcal{C}) \backslash \bigcup_{i=1}^{k} S_i \right) \cup \bigcup_{i=1}^{k} \{R_i\} \tag{5}$$

Note that each subset of relations $S_i \subseteq K(\mathcal{C})$ corresponding to the connected component T_i is replaced by a unique relation R_i. By Lemma 11 there are at most $2 \cdot |\mathit{leaves}(T, Y)| - 1$ connected components in T'. Therefore, the number of relations in K is upper-bounded by $|\mathit{leaves}(T, Y)| + 2|\mathit{leaves}(T, Y)| - 1 = 3|\mathit{leaves}(T, Y)|$. We claim that $R(K)|_Y = R(K(\mathcal{C}))|_Y$. To prove this claim, let $K_0, K_1, \ldots, K_k$ be a sequence of CSPs where $K_0 = K(\mathcal{C})$, and for each $i \in \{1, \ldots, k\}$, $K_i = (K_{i-1} \backslash S_i) \cup \{R_i\}$. Then clearly we have that $K = K_k$. We claim that for each $j \in \{0, \ldots, k\}$, $K_j|_Y = K(\mathcal{C})|_Y$. In the base case, $k = 0$, and the claim follows trivially. Now assume that $K_j|_Y = K(\mathcal{C})|_Y$. By Lemma 8, we have that $K_j|_Y = K_{j+1}|_Y$. □

4 Symmetric Non-deterministic State Complexity

Let Σ be a finite set of symbols. In this section we introduce the notion of symmetric non-deterministic state complexity of functions of the form $f : \Sigma^n \to \{0, 1\}$ and of finite languages included in Σ^n. We note that this notion is intimately related with the size of the smallest non-deterministic oblivious, read-once branching program [26] computing f. A non-deterministic finite automaton (NFA) over Σ is a 5-tuple $\mathcal{A} = (Q, \Sigma, \mathfrak{R}, Q_0, F)$ where Q is a set of states, $Q_0 \subseteq Q$ is a set of initial states, $F \subseteq Q$ is a set of final states and $\mathfrak{R} \subseteq Q \times \Sigma \times Q$ is a transition relation. We write $q \xrightarrow{a} q'$ to denote that the triple (q, a, q') belongs to $\mathfrak{R}$. We say that a string $w = w_1 w_2 \ldots w_n \in \Sigma^n$ is accepted by $\mathcal{A}$ if there is a

sequence $q_0 \xrightarrow{w_1} q_1 \xrightarrow{w_2} \ldots \xrightarrow{w_n} q_n$ such that $q_0 \in Q_0$ and $q_n \in Q_F$. We denote by $\mathcal{L}(\mathcal{A})$ the set of all strings accepted by $\mathcal{A}$.

Let $\mathcal{L} \subseteq \Sigma^n$ be a set of length-n strings over Σ. The *non-deterministic* state complexity (NSC) of $\mathcal{L}$, denoted $nsc(\mathcal{L})$, is defined as the minimum number of states of a NFA accepting $\mathcal{L}$. Let $f : \Sigma^n \to \{0,1\}$ be a function. We denote by $\mathcal{L}(f)$ the set of all strings $w \in \Sigma^n$ for which $f(w) = 1$. We define the non-deterministic state complexity of f as $nsc(f) := nsc(\mathcal{L}(f))$. For each positive integer n, we define $[n] = \{1, \ldots, n\}$. We denote by $Perm(n)$ the set of all permutations of the set $[n]$. If $\pi : [n] \to [n]$ is a permutation in $Perm(n)$ and $w \in \Sigma^n$, then we let $\pi(w)$ be the string in Σ^n that is defined by setting $\pi(w)_{\pi(j)} = w_j$ for each $j \in [n]$. Intuitively, the j-th position of w is mapped to the position $\pi(j)$ of $\pi(w)$. If $\mathcal{L} \subseteq \Sigma^n$ is a set of length-n strings over Σ, then we denote by $\pi(\mathcal{L})$ the language obtained from $\mathcal{L}$ by permuting the coordinates of each string in $\mathcal{L}$ according to π. More precisely,

$$\pi(\mathcal{L}) = \{\pi(w) \mid w \in \mathcal{L}\}. \tag{6}$$

The symmetric non-deterministic state complexity (symmetric-NSC) of a language $\mathcal{L} \subseteq \Sigma^n$ is defined as the minimum non-deterministic state complexity of a permuted version of $\mathcal{L}$.

▶ **Definition 12** (Symmetric Nondeterministic State Complexity)**.** Let $\mathcal{L} \subseteq \Sigma^n$. The symmetric non-deterministic state complexity of $\mathcal{L}$ is defined as

$$snsc(\mathcal{L}) = \min_{\pi \in Perm(n)} nsc(\pi(\mathcal{L})). \tag{7}$$

The symmetric-NSC of a function $f : \Sigma^n \to \{0,1\}$ is defined as $snsc(f) = snsc(\mathcal{L}(f))$.

We note that the symmetric-NSC of a function $f : \{0,1\}^n \to \{0,1\}$ is lower-bounded by the size of the smallest non-deterministic oblivious read-once $|\Sigma|$-way branching program computing f. Therefore, functions requiring exponential size branching programs of this particular form have exponential symmetric non-deterministic state complexity. Two examples of such functions are the *triangle-freeness* function $\Delta_n : \{0,1\}^n \to \{0,1\}$, and the *triangle-parity* function $\bigoplus \text{Clique}_{3,n} : \{0,1\}^n \to \{0,1\}$. Both functions take as input an array $x = (x_{ij})_{1 \leq i < j \leq m}$ consisting of $n = \binom{m}{2}$ Boolean variables representing an undirected graph $G(x)$ on m vertices $\{1, \ldots, m\}$. The graph $G(x)$ has an edge connecting vertices i and j, with $i < j$, if and only if $x_{ij} = 1$. The triangle-freeness function Δ_n returns 1 on an input x if and only if the graph $G(x)$ does not contain a triangle. The triangle-parity function $\bigoplus \text{Clique}_{3,n}$ returns 1 if and only if the parity of the number of the triangles in $G(x)$ is odd. In [9] it was shown that read-once non-deterministic branching programs computing the functions Δ_n and $\bigoplus \text{Clique}_{3,n}$ require size $2^{\Omega(n)}$. Therefore, the same lower bound holds for the symmetric-NSC of these functions.

▶ **Theorem 13** ([9])**.** *$snsc(\Delta_n) \geq 2^{\Omega(n)}$ and $snsc(\bigoplus \text{Clique}_{3,n}) \geq 2^{\Omega(n)}$.*

4.1 On a Variant of the Element Distinctness Function

We say that a binary string w is *even* if w has an even number of ones. Analogously, we say that w is *odd* if w has an odd number of ones. For each $r \in \mathbb{N}$, we let $even(r)$ denote the set of all strings of even parity in the set $\{0,1\}^r$. Let $\Sigma(m) = \{1, \ldots, m\}$, and $P(m) \subseteq \Sigma(m)^m$ be the set of all length-m strings over $\Sigma(m) = \{1, \ldots, m\}$ whose entries are pairwise distinct. Let $n = (\lceil \log m \rceil + 1) \cdot m$ and let $b : \{1, \ldots, m\} \to even(\lceil \log m \rceil + 1)$ be an injection that maps each number $j \in \{1, \ldots, m\}$ to an even binary string $b(j)$ of length $\lceil \log m \rceil + 1$. We let

$$B(n) = \{b(w_1)b(w_2)\ldots b(w_m) \in \{0,1\}^n \mid w_1 w_2 \ldots w_m \in P(m)\}$$

be the binary language that is obtained from $P(m)$ by mapping each string in $P(m)$ to its binary representation. We define the *even element distinctness function* $\tau_n : \{0,1\}^n \to \{0,1\}$ as the function that returns 1 on an input $w \in \{0,1\}^n$ if and only if $w \in B(n)$. Note that by definition, the symmetric-NSC of τ_n is the symmetric-NSC of $B(n)$.

▶ **Theorem 14.** *The function $\tau_n : \{0,1\}^n \to \{0,1\}$ has* symmetric-*NSC $2^{\Omega(n/\log n)}$.*

The proof of Theorem 14 will use the following result.

▶ **Theorem 15** (Glaister-Shallit [13])**.** *Let $\mathcal{L} \subseteq \Sigma^n$ be a set of length-n strings over Σ, and suppose that there exists a set $F = \{(x_i, w_i) \mid 1 \leq i \leq k\}$ of pairs of strings such that*
1. *$x_i \cdot w_i \in \mathcal{L}$ for $1 \leq i \leq k$*
2. *$x_i \cdot w_j \notin \mathcal{L}$ for $1 \leq i, j \leq k$ and $i \neq j$*

Then any non-deterministic finite automaton accepting $\mathcal{L}$ has at least k states.

The set F in Theorem 15 is called a fooling set for $\mathcal{L}$. We will prove Theorem 14 by constructing, for each permutation $\pi : [n] \to [n]$, a fooling set F_π of size $2^{\Omega(n/\log n)}$ for the language $\pi(B(n))$. To construct F_π it will be convenient to view strings as Boolean functions over sets of positions. In other words, if S is a set of positive integers, then a string over S is simply a Boolean function $w : S \to \{0,1\}$. We note that we allow S to be any set of positive integers and not necessarily an interval of the form $[n] = \{1, \ldots, n\}$. The parity of w is defined as the parity of the number of positions in which w evaluates to 1: $par(w) = |\{i \in S \mid w(i) = 1\}| \mod 2$. The restriction of w to a subset $T \subseteq S$ is the string $w|_T : T \to \{0,1\}$ which is defined by setting $w|_T(i) = w(i)$ for every $i \in T$. If $S \subseteq [n]$, $w : S \to \{0,1\}$ is a string, and $\pi : [n] \to [n]$ is a permutation, then we let $\pi(w)$ be the string $w' : \pi(S) \to \{0,1\}$ that is defined by setting $w'(\pi(i)) = w(i)$ for each $i \in S$. We let $L = \{1, \ldots, \lfloor n/2 \rfloor\}$ and $R = \{\lfloor n/2 \rfloor + 1, \ldots, n\}$ be respectively the first and the second halves of the set $[n] = \{1, \ldots, n\}$. We say that a permutation π splits a subset $S \subseteq [n]$ if $\pi(S) \cap L \neq \emptyset$ and $\pi(S) \cap R \neq \emptyset$. In other words, π splits S if some elements of S are mapped by π to the first half of $[n]$ and some elements of S are mapped by π to the second half of $[n]$.

If S and S' are subsets of $[n]$ such that $S \cap S' = \emptyset$, and $w : S \to \{0,1\}$ and $w' : S' \to \{0,1\}$ are strings with domain S and S' respectively, then the concatenation of w with w' is simply the function $w \cdot w' : S \cup S' \to \{0,1\}$ which is equal to w when restricted to S and equal to w' when restricted to S'. Let $S = \{j_1, j_2, \ldots, j_k\} \subseteq [n]$ where $j_1 < j_2 < \ldots < j_k$. We let $ord_S(i) = j_i$ denote the i-th element of S. Let S and S' be subsets of $[n]$ of same size. We say that strings $w : S \to \{0,1\}$ and $w' : S' \to \{0,1\}$ are equivalent, which we denote by $w \equiv w'$, if for each $i \in \{1, \ldots, |S|\}$, $w(ord_S(i)) = w(ord_{S'}(i))$. Note that if $S = S'$ then $w \equiv w'$ if and only if $w = w'$. Let $n = (1 + \lceil \log m \rceil) \cdot m$. Let $I_1, \ldots, I_m$ be the sequence of subsets of $[n]$ such that for each $i \in [m]$, $I_i = \{(i-1) \cdot (1 + \lceil \log m \rceil), \ldots, i \cdot (1 + \lceil \log m \rceil)\}$. In other words, $I_1, \ldots, I_m$ is a partition of the set $[n]$ into m consecutive intervals of equal size. We say that $I_1, \ldots, I_m$ is the *uniform interval partition* of $[n]$. Let $I_{i_1}, I_{i_2}, \ldots, I_{i_k}$ be intervals which are split by π. Let $(I^L_{j_1}, I^R_{j_1}), \ldots, (I^L_{j_l}, I^R_{j_l})$ be pairs of intervals such that for each $r \in \{1, \ldots, l\}$, $\pi(I^L_{j_r}) \subseteq L$ and $\pi(I^R_{j_r}) \subseteq R$. Let $a_1, \ldots, a_k$, $b_1, \ldots, b_k$, $c_1, \ldots, c_l$ and $d_1, \ldots, d_l$ be $2k + 2l$ distinct binary strings with domain $\{1, \ldots, \lceil \log m \rceil + 1\}$ such that the following conditions are satisfied for each $r \in \{1, \ldots, k\}$.
1. $a_r|_{\pi(I_{i_r}) \cap L}$ is even and $a_r|_{\pi(I_{i_r}) \cap R}$ is even.
2. $b_r|_{\pi(I_{i_r}) \cap L}$ is odd and $b_r|_{\pi(I_{i_r}) \cap R}$ is odd.

For each $(k+l)$-tuple of bits $x_1, \ldots, x_k, y_1, \ldots, y_l$, select a string $w[x_1, \ldots, x_k, y_1, \ldots, y_l]$ in $\{0,1\}^n$ satisfying the following properties for each $r \in \{1, \ldots, k\}$ and $s \in \{1, \ldots, l\}$.
1. If $x_r = 0$ then $w[x_1, \ldots, x_k, y_1, \ldots, y_l]|_{I_{i_r}} \equiv a_r$

2. If $x_r = 1$ then $w[x_1, \ldots, x_k, y_1, \ldots, y_l]|_{I_{i_r}} \equiv b_r$
3. If $y_s = 0$ then $w[x_1 \ldots, x_k, y_1, \ldots, y_l]|_{I^L_{j_s}} \equiv c_s$ and $w[x_1, \ldots, x_k, y_1, \ldots, y_l]|_{I^R_{j_s}} \equiv d_s$
4. If $y_s = 1$ then $w[x_1 \ldots, x_k, y_1, \ldots, y_l]|_{I^L_{j_s}} \equiv d_s$ and $w[x_1, \ldots, x_k, y_1, \ldots, y_l]|_{I^R_{j_s}} \equiv c_s$

We let $F^{k,l}_\pi$ be the set obtained by splitting each string $w[x_1, \ldots, x_k, y_1, \ldots, y_l]$ into a left part and a right part.

$$F^{k,l}_\pi = \{ \; (\; w[x_1, \ldots, x_k, y_1, \ldots, y_l]|_L, \; w[x_1, \ldots, x_k, y_1, \ldots, y_l]|_R \;) \; \mid \; x_r, y_s \in \{0,1\}\} \quad (8)$$

▶ **Theorem 16.** *The set $F^{k,l}_\pi$ defined in Equation 8 is a fooling set for $\pi(B(n))$ of size 2^{k+l}.*

We note that for each permutation π, there exists an $\alpha \leq m/4$ such that the fooling set $F_\pi = F^{\alpha, m/4-\alpha}_\pi$ is well defined. Therefore, by Theorem 16, we have that

$$|F_\pi| \geq 2^{m/4} = 2^{\Omega(n/\log n)}.$$

5 Non-Linear Lower Bounds for Read-Once Circuits Excluding a Minor

In this section we show that exponential lower bounds for the symmetric non-deterministic complexity of a function $f_n : \{0,1\}^n \to \{0,1\}$ imply super-linear lower bounds for the size of read-once circuits excluding a fixed graph H as a minor. We start by establishing a connection between the symmetric-NSC of a circuit, and its pathwidth. More precisely, we will show that any function that can be computed by a read-once circuit of pathwidth k has symmetric-NSC at most $2^k \cdot |\mathcal{C}|$.

▶ **Definition 17** (Path Decomposition)**.** A path decomposition of a circuit $\mathcal{C} = (V, E, \mathbf{g})$ is a sequence $\mathcal{P} = (B_1, B_2, \ldots, B_m)$ of subsets of vertices of G satisfying the following properties.

(i) $V = \bigcup_{i=1}^n B_i$.
(ii) For each $i, j, k \in \mathbb{N}$ with $i < j < k$, $B_i \cap B_k \subset B_j$.
(iii) For each edge $(u, v) \in E$ there is an j such that $\{u, v\} \subseteq B_j$.

The sets B_i are the bags of the decomposition. The path-width of $\mathcal{P}$ is defined as the size of its largest bag minus one. In other words $\boldsymbol{pw}(G, \mathcal{P}) = \max_i\{|B_i|\} - 1$. The pathwidth of a graph G is defined as $\boldsymbol{pw}(G) = \min_{\mathcal{P}} \boldsymbol{pw}(G, \mathcal{P})$ where $\mathcal{P}$ ranges over all path decompositions of G. Let $\mathcal{C}$ be a read-once circuit and $\mathcal{P} = (B_1, B_2, \ldots, B_m)$ be a path decomposition of $\mathcal{C}$. If v is a vertex of $\mathcal{C}$ then we let $\mathit{first}(v, \mathcal{P})$ denote the smallest i such that $v \in B_i$.

▶ **Theorem 18.** *Let $\mathcal{C}$ be a read-once circuit and $\mathcal{P} = (B_1, B_2, \ldots, B_m)$ be a path decomposition of $\mathcal{C}$ of width w. Let $x_1 x_2 \ldots x_n$ be an ordering of the variables of $\mathcal{C}$ such that $\mathit{first}(x_i, \mathcal{P}) < \mathit{first}(x_{i+1}, \mathcal{P})$ for each $i \in \{1, \ldots, n-1\}$. Then for each $b \in \Sigma$, one can construct a NFA on $|\Sigma|^{O(w)} \cdot m$ states accepting the following language.*

$$\mathcal{L}(\mathcal{C}, b) = \{a_1 a_2 \ldots a_n \in \Sigma^n \mid C(a_1 a_2 \ldots a_n) = b\}.$$

We note that any read-once circuit $\mathcal{C}$ of pathwidth k has a decomposition of width k with $O(|\mathcal{C}|)$ bags[3]. Therefore, as a corollary of Theorem 18 and Theorem 14 we have a trade-off between the size of a circuit and its pathwidth.

[3] Any graph G of pathwidth w has a path decomposition of width w with $|G|$ bags.

▶ **Theorem 19.** *Let $f_n : \{0,1\}^n \to \{0,1\}$ be a function of symmetric-NSC $snsc(f_n)$. Then for any read-once circuit computing f_n, the following inequality is satisfied.*

$$\boldsymbol{pw}(\mathcal{C}) + \log|\mathcal{C}| \geq \log snsc(f_n). \tag{9}$$

It is well known that the pathwidth of any graph is greater than its treewidth by at most a multiplicative logarithmic factor [5]. In other words, the following relation between the pathwidth and treewidth of a circuit (graph) can be verified: $\boldsymbol{pw}(\mathcal{C}) \leq \boldsymbol{tw}(\mathcal{C}) \cdot O(\log|\mathcal{C}|)$. Therefore, stated in terms of treewidth, Equation 9 can be rewritten as follows.

$$\boldsymbol{tw}(\mathcal{C}) \cdot \log|\mathcal{C}| + \log|\mathcal{C}| \geq \Omega(\log snsc(f_n)). \tag{10}$$

▶ **Theorem 20** ([3],[14])**.** *For any fixed graph H, every H-minor-free graph G with s vertices has treewidth at most $O(\sqrt{s})$.*

Therefore, combining Equation 10 with Theorem 20 we have the following theorem. We say that a circuit $\mathcal{C}$ is H-minor-free if its underlying undirected graph is H-minor-free.

▶ **Theorem 21.** *Let $\mathcal{C}$ be an H-minor-free, read-once circuit computing a Boolean function $f : \{0,1\}^n \to \{0,1\}$. Then $|\mathcal{C}| \geq \Omega\left(\left(\frac{\log snsc(f_n)}{\log n}\right)^2\right)$.*

Finally, as a corollary of Theorem 21, Theorem 13 and Theorem 14 we have that the triangle-freeness function Δ_n, the triangle-parity function $\bigoplus \text{Clique}_{3,n}$ and the even element distinctness function τ_n require H-minor-free read-once circuits of near quadratic size.

▶ **Corollary 22.** *Let $\mathcal{C}$, $\mathcal{C}'$ and $\mathcal{C}''$ be H-minor-free, read-once circuits computing the triangle-freeness function Δ_n, the triangle-parity function $\bigoplus \text{Clique}_{3,n}$ and the even element distinctness function τ_n respectively. Then $|\mathcal{C}| \geq \Omega(\frac{n^2}{\log^2 n})$, $|\mathcal{C}'| \geq \Omega(\frac{n^2}{\log^2 n})$, and $|\mathcal{C}''| \geq \Omega(\frac{n^2}{\log^4 n})$.*

Acknowledgements. The author would like to thank Pavel Pudlák for interesting discussions on circuit lower-bounds, Pavel Hrubeš for pointing me out to reference [20], Michal Koucký and Bruno Loff for useful feedback during a seminar presentation of this work, and anonymous referees for valuable comments.

References

1 Michael Alekhnovich and Alexander A Razborov. Satisfiability, branch-width and Tseitin tautologies. In *Proc. of the 43rd Symposium on Foundations of Computer Science*, pages 593–603, 2002.

2 Eric Allender, Shiteng Chen, Tiancheng Lou, Periklis A. Papakonstantinou, and Bangsheng Tang. Width-parametrized SAT: Time–space tradeoffs. *Theory of Computing*, 10(12):297–339, 2014. doi:10.4086/toc.2014.v010a012.

3 Noga Alon, Paul Seymour, and Robin Thomas. A separator theorem for graphs with an excluded minor and its applications. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 293–299. ACM, 1990.

4 Norbert Blum. A boolean function requiring 3n network size. *Theoretical Computer Science*, 28(3):337–345, 1983.

5 Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998.

6 Elizabeth Broering and Satyanarayana V Lokam. Width-based algorithms for SAT and CIRCUIT-SAT. In *Theory and Applications of Satisfiability Testing*, pages 162–171. Springer, 2004.

7 Chris Calabro. A lower bound on the size of series-parallel graphs dense in long paths. *Electronic Colloquium on Computational Complexity (ECCC)*, 15(110), 2008.

8 Evgeny Demenkov and Alexander S Kulikov. An elementary proof of a $3n - o(n)$ lower bound on the circuit complexity of affine dispersers. In *Mathematical Foundations of Computer Science 2011*, pages 256–265. Springer, 2011.

9 Pavol Duris, Juraj Hromkovic, Stasys Jukna, Martin Sauerhoff, and Georg Schnitger. On multi-partition communication complexity. *Information and Computation*, 194(1):49–75, 2004.

10 Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than-3n lower bound for the circuit complexity of an explicit function. *Electronic Colloquium on Computational Complexity (ECCC)*, 22(166), 2015.

11 Anna Gál and Jing-Tang Jang. A generalization of spira's theorem and circuits with small segregators or separators. In *Theory and Practice of Computer Science (SOFSEM 2012)*, pages 264–276. Springer, 2012.

12 Konstantinos Georgiou and Periklis A Papakonstantinou. Complexity and algorithms for well-structured k-sat instances. In *Proc. of the 11th International Conference on Theory and Applications of Satisfiability Testing*, pages 105–118. Springer, 2008.

13 Ian Glaister and Jeffrey Shallit. A lower bound technique for the size of nondeterministic finite automata. *Information Processing Letters*, 59(2):75–77, 1996.

14 Martin Grohe. Local tree-width, excluded minors, and approximation algorithms. *Combinatorica*, 23(4):613–632, 2003.

15 Johan Håstad. The shrinkage exponent of de morgan formulas is 2. *SIAM Journal on Computing*, 27(1):48–64, 1998.

16 Jing He, Hongyu Liang, and Jayalal MN Sarma. Limiting negations in bounded treewidth and upward planar circuits. In *Mathematical Foundations of Computer Science 2010*, pages 417–428. Springer, 2010.

17 Juraj Hromkovič. Communication complexity and lower bounds on multilective computations. *RAIRO-Theoretical Informatics and Applications*, 33(02):193–212, 1999.

18 Kazuo Iwama and Hiroki Morizumi. An explicit lower bound of 5n- o (n) for boolean circuits. In *Mathematical foundations of computer science 2002*, pages 353–364. Springer, 2002.

19 Maurice Jansen and Jayalal Sarma. Balancing bounded treewidth circuits. In *Computer Science – Theory and Applications*, pages 228–239. Springer, 2010.

20 Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

21 Oded Lachish and Ran Raz. Explicit lower bound of 4.5 n-o (n) for boolena circuits. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 399–408. ACM, 2001.

22 Richard J Lipton and Robert Endre Tarjan. Applications of a planar separator theorem. *SIAM journal on computing*, 9(3):615–627, 1980.

23 Igor L Markov and Yaoyun Shi. Constant-degree graph expansions that preserve treewidth. *Algorithmica*, 59(4):461–470, 2011.

24 Nečiporuk. On a Boolean function. *Soviet Math. Dokl.*, 7(4):999–1000, 1966.

25 Ramamohan Paturi and Pavel Pudlák. Circuit lower bounds and linear codes. *Journal of Mathematical Sciences*, 134(5):2425–2434, 2006.

26 Pavel Pudlák. The hierarchy of boolean circuits. *Computers and artificial intelligence*, 6(5):449–468, 1987.

27 Neil Robertson and Paul D Seymour. Graph minors. xiii. the disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

28 Rahul Santhanam and Srikanth Srinivasan. On the limits of sparsification. In *Automata, Languages, and Programming*, pages 774–785. Springer, 2012.

29 John E. Savage. Planar circuit complexity and the performance of VLSI algorithms. In *INRIA Report* 77 (1981). *Also in VLSI Systems and Computations*, pages 61–67. Computer Science Press Rockwille MD, 1981.

30 György Turán. On the complexity of planar boolean circuits. *Computational Complexity*, 5(1):24–42, 1995.

31 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *6th Symposium on Mathematical Foundations of Computer Science*, pages 162–176, 1977.