Nondeterminism and an abstract formulation of Nečiporuk's lower bound method

Paul Beame * Nathan Grosshans † Pierre McKenzie ‡ Luc Segoufin §

Tuesday 27th September, 2016

Abstract

A formulation of *Nečiporuk's lower bound method* slightly more inclusive than the usual complexity-measure-specific formulation is presented. Using this general formulation, limitations to lower bounds achievable by the method are obtained for several computation models, such as branching programs and Boolean formulas having access to a sublinear number of nondeterministic bits. In particular, it is shown that any lower bound achievable by the method of Nečiporuk for the size of nondeterministic and parity branching programs is at most $O(n^{3/2}/\log n)$.

1 Introduction

Relatively few methods exist to prove complexity lower bounds in general non-uniform models of computation. Fifty years ago, Nečiporuk wrote his famous two-page note entitled "A boolean function" [15]. That note contained the first super-linear lower bounds on the size of Boolean formulas over arbitrary bases and the size of contact schemes needed to compute some explicit Boolean function.

Nečiporuk's [15] method still yields the best lower bounds known today for explicit functions in a number of complexity measures. In particular, there are explicit functions for which Nečiporuk's method yields lower bounds of $\Omega(n^2/\log n)$ on formula size over an arbitrary basis, $\Omega(n^2/\log^2 n)$ on deterministic branching program size and on contact scheme size, and $\Omega(n^{3/2}/\log n)$ on nondeterministic branching program size, switching-and-rectifier network size, parity branching program size and span program size. All of these are the best known lower bounds for these complexity measures for any explicit function. The first two of these lower bounds are contained in Nečiporuk's original paper [15]. Pudlak [17] points out that Nečiporuk's method yields the third lower bound for *nondeterministic* branching program size, as well as for switching-and-rectifier network size program size [18]; Karchmer and Wigderson [9] point out that the Pudlak's observation extends to parity branching program size and hence also applies to span program size.

Two simple explicit functions that yield the lower bounds mentioned above are the Element Distinctness function and the Indirect Storage Access function.

*University of Washington, beame@cs.washington.edu. Research supported by NSF grants CCF-1217099 and CCF-1524246.

†Université de Montréal and ENS Cachan, Université Paris-Saclay, nathan.grosshans@lsv.ens-cachan.fr. This work was supported by grants from Digiteo France.

‡Université de Montréal and ENS Cachan, Université Paris-Saclay, mckenzie@iro.umontreal.ca. Research supported by NSERC of Canada and by the "Chaire DIGITEO, ENS Cachan - École Polytechnique".

§INRIA and ENS Cachan, Université Paris-Saclay, luc.segoufin@inria.fr.

Nečiporuk's method relies on counting subfunctions induced on blocks in a partition of the input variables. It is natural to try to optimize the use of the method, both in terms of how the bound depends on the numbers of subfunctions for each block in the partition and whether there are functions other than Element Distinctness and Indirect Storage Access for which one can prove stronger lower bounds.

For formula size over arbitrary bases, it is well known (see, e.g., [20, 19]) that $\Theta(n^2/\log n)$ is indeed the best lower bound obtainable by Nečiporuk's method. Savage [19] also cites Paterson (unpublished) as improving the constant factor in the bound. Similarly, $\Theta(n^2/\log^2 n)$ is the best lower bound obtainable by Nečiporuk's method for deterministic branching program size, as noted by Wegener [20, p. 422], who states the claim with a hint at its proof. Moreover, Alon and Zwick [1] derived the optimal multiplicative constant in this lower bound as a function of the number of subfunctions of f in each block.

Since the first two bounds using Nečiporuk's method are asymptotically the best possible, it is natural to ask whether the third lower bound also uses Nečiporuk's method in an optimal way. Jukna seems to be the only one who has explicitly addressed this question. In his discussion [7, p. 207] of the $\Omega(n^{3/2}/\log n)$ lower bound on span program size due to Karchmer and Wigderson [9] and based on Nečiporuk's method, he states that the method "cannot lead to much larger lower bounds" but does not give more details.

In this paper we give a more precise result, namely that the best bound on nondeterministic and parity branching program size obtainable by Nečiporuk's method is indeed $\Theta(n^{3/2}/\log n)$. This automatically applies to span program size and switching-and-rectifier network size since these measures are upper-bounded by parity and nondeterministic branching program size, respectively.

In deriving lower bounds using Nečiporuk's method as discussed in the literature, the major difference between measures is the estimate of the number of functions of low complexity with respect to each measure. In most presentations of Nečiporuk's method, such as those in [20, 2, 21], this bound is determined *syntactically*, for example, via a count of the number of syntactically distinct formulas of a given size or of syntactically distinct branching programs with a given number of nodes. However, this is an overcount of the number of different functions since, for example, many syntactically distinct branching programs may compute the same function. If only *semantically* distinct objects are counted, one may, in principle, obtain stronger lower bounds using Nečiporuk's method. In the case of deterministic branching programs, Alon and Zwick [1] considered the stronger semantic version of Nečiporuk's bound and showed nonetheless that both semantic and syntactic versions reach the same asymptotic limit. Our formulation of the method will subsume such considerations.

We also use Nečiporuk's method to derive lower bounds on models with limited nondeterminism, including branching programs and formulas with limited nondeterminism as well as to prove limitations on the application of Nečiporuk's method. In fact, we show that the Indirect Storage Access function with suitable parameters yields asymptotically best-possible lower bounds in each of these models. To do this, though, we first need to define precisely *what* the Nečiporuk method actually is, independently from any specific complexity measure, so as to provide a unifying framework in which it makes sense to speak about *the* Nečiporuk method. Indeed, although Nečiporuk published his original result 50 years ago and his "technique" has been treated in several classical references (see [19, 20, 21, 8]), to the best of our knowledge, there did not exist any abstract, measure-independent, unifying definition of the "method" that would encompass all previous applications of the "method" and allow new ones to be carried out easily, or at least in a clear way. The definition we suggest is in fact an abstract version of the general definition that was considered by Alon and Zwick in [1] for the case of deterministic branching programs. In this abstract framework we can then show, in a generic way, that for

any complexity measure, an upper bound on the complexity, for this measure, of computing the Indirect Storage Access function with suitable parameters yields an upper bound on the best lower bound obtainable using Nečiporuk's method (defined that way). We then deduce some well-known lower bounds and limitations results, as well as new ones, in this framework.

Summary of results

- We prove that Nečiporuk's method as usually interpreted in the literature yields no better nondeterministic or parity branching program size lower bounds than $\Theta\left(\frac{n^{3/2}}{\log_2 n}\right)$.
- We provide a formulation of Nečiporuk's lower bound method upstream from any specific complexity measure and link the limitations of this method (in terms of the best lower bound obtainable) to upper bounds on the complexity for computing one specific family of functions (the Indirect Storage Access functions).
- We apply the method to two classical concrete complexity measures and some variants: the size of (deterministic) branching programs (BPs) and their nondeterministic (NBPs) and limited nondeterministic (LNBPs) and parity ($\oplus$BPs) variants, as well as the size of binary formulas (BFs) and their limited nondeterministic variant (LNBFs). See Subsection 2.3 for the formal definitions of these models.

Complexity measure	**Best lower bound obtainable**
Size of NBPs	$\Theta\left(\frac{n^{3/2}}{\log_2 n}\right)$
Size of $\oplus$BPs	$\Theta\left(\frac{n^{3/2}}{\log_2 n}\right)$
Size of LNBPs using $\Delta(n)$ nondeterministic bits	$\Theta\left(\frac{n^2}{2^{\Delta(n)}(\log_2(n)-\Delta(n))\log_2 n}\right)$ $(\star)$
Size of BPs	$\Theta\left(\frac{n^2}{\log_2^2 n}\right)$
Size of LNBFs using $\Delta(n)$ nondeterministic bits	$\Theta\left(\frac{n^2}{2^{\Delta(n)}\log_2 n}\right)$ $(\star)$
Size of BFs	$\Theta\left(\frac{n^2}{\log_2 n}\right)$

Table 1: Bounds for the Indirect Storage Access function, which this paper shows to be the best lower bounds obtainable by Nečiporuk's method for any function. The star indicates that the true function is more complicated; however, this current formulation holds for all Δ not too big with respect to $n \mapsto n$.

2 Preliminaries

For $n \in \mathbb{N}$, we denote by $[n]$ the set of integers from 1 to n, using the convention that $[0] = \emptyset$. For $k \in \mathbb{N}_{>0}$ and $\vec{a} \in \{0,1\}^k$ we denote by $\mathrm{bin}_k(\vec{a})$ its associated natural number with big-endian representation, i.e. $\sum_{i=1}^{k} a_i 2^{k-i}$. Throughout the paper, the binary representation of a natural number will refer to its big-endian representation.

Let V be a subset of $\mathbb{N}$. We view $\vec{a} \in \{0,1\}^V$ as a tuple of bits of length $|V|$ and for $i \in V$ we denote by a_i the corresponding bit of $\vec{a}$.

2.1 Boolean functions and subfunctions

For $n \in \mathbb{N}$, an n-ary *Boolean function over* V is a function $f\colon \{0,1\}^V \to \{0,1\}$, where $V \subseteq \mathbb{N}$ and $|V| = n$. When V is not specified we assume that $V = [n]$. A *family of Boolean functions* is

an indexed family $F = \{f_i\}_{i \in I}$ where $I \subseteq \mathbb{N}$ and such that for all $i \in I$, f_i is a Boolean function of arity i.

Subfunctions will play a key role in the lower bound method studied in this paper, the intuition being that the more different subfunctions a given Boolean function has, the more difficult it is to compute it.

Let f be an n-ary Boolean function. For any $V \subseteq [n]$ and any $\rho \in \{0,1\}^{[n]\setminus V}$, we will denote by $f|_\rho$ the *subfunction of f on V induced by the partial assignment ρ on $[n] \setminus V$*, that is the function $f|_\rho \colon \{0,1\}^V \to \{0,1\}$ such that for all $\vec{y} \in \{0,1\}^V$, we have $f|_\rho(\vec{y}) = f(\vec{x})$ where $x_i = y_i$ for all $i \in V$ and $x_i = \rho(i)$ for all $i \in [n] \setminus V$. We will also denote by $r_V(f)$ the *total number of subfunctions of f on V*, i.e. the cardinality of the set $s_V(f) = \{f|_\rho \mid \rho \in \{0,1\}^{[n]\setminus V}\}$.

The following easy lemma gives an upper bound on the total number of subfunctions of a given Boolean function on a certain subset of input variable indices.

Lemma 2.1. *Let $f \colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. For any $V \subseteq [n]$, $r_V(f) \leq \min\{2^{2^{|V|}}, 2^{n-|V|}\}$.*

Proof. Since $r_V(f)$ counts the total number of subfunctions $f|_\rho \colon \{0,1\}^V \to \{0,1\}$ of f on V induced by a partial assignment $\rho \in \{0,1\}^{[n]\setminus V}$, it is at most the total number of Boolean functions on $|V|$ variables (i.e. $2^{2^{|V|}}$) and the total number of assignments to $n - |V|$ variables (i.e. $2^{n-|V|}$). □

Let $f \colon \{0,1\}^V \to \{0,1\}$ be a Boolean function and $i \in V$. We say that *f depends on its i^{th} variable* if there exist $\vec{a}, \vec{a}' \in \{0,1\}^V$ that differ only in the bit corresponding to i such that $f(\vec{a}) \neq f(\vec{a}')$ (definition based on [8]). In particular, if f does not depend on a set W of variables and $\vec{a}, \vec{a}' \in \{0,1\}^V$ differ only on bits whose positions are in W, then $f(\vec{a}) = f(\vec{a}')$. The following proposition shows that variables on which f does not depend do not affect its number of subfunctions.

Proposition 2.2. *Let $f \colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. Let $V \subseteq [n]$ and $W \subseteq V$ such that for all $i \in W$, f does not depend on x_i. Then for $V' = V \setminus W$, $r_V(f) = r_{V'}(f)$.*

Proof. We give a bijection from $s_V(f)$ to $s_{V'}(f)$. For $g \in s_V(f)$, say $g = f|_\rho$ for some $\rho \in \{0,1\}^{[n]\setminus V}$, define $\psi(g) \in s_{V'}(f)$ by $\psi(g) = g|_\zeta = f|_{\rho\zeta}$ where $\zeta = 0^W$ assigns 0 to all elements of W. By assumption, for all $i \in W$, f does not depend on x_i so for all $\zeta' \in \{0,1\}^W$, $f|_{\rho\zeta'} = f|_{\rho\zeta}$; moreover, it is also easy to see that $f|_{\rho\zeta} = f|_{\rho'\zeta}$ for any $\rho' \in \{0,1\}^{[n]\setminus V}$ such that $g = f|_{\rho'}$. Now let $h' \in s_{V'}(f)$. By definition, there is some $\rho' \in \{0,1\}^{[n]\setminus V}$ and $\zeta' \in \{0,1\}^W$ such that $h' = f|_{\rho'\zeta'}$ and by assumption, the latter equals $f|_{\rho'\zeta} = \psi(g')$ for $g' = f|_{\rho'}$ and hence ψ is surjective.

Similarly, for $g, g' \in s_V(f)$ such that $g \neq g'$, we have that there exists $\vec{a} \in \{0,1\}^V$ verifying $g(\vec{a}) \neq g'(\vec{a})$. Let $\rho, \rho' \in \{0,1\}^V$ such that $g = f|_\rho$ and $g' = f|_{\rho'}$, and let $\zeta' \in \{0,1\}^W$ such that $\zeta'(i) = a_i$ for all $i \in W$. Then $\psi(g) = f|_{\rho\zeta} = f|_{\rho\zeta'} \neq f|_{\rho'\zeta'} = f|_{\rho'\zeta} = \psi(g')$, so ψ is 1-1 and hence $r_V(f) = |s_V(f)| = |s_{V'}(f)| = r_{V'}(f)$. □

It will often also be useful to enlarge the size of the domain of a Boolean function by adding additional input variables on which the function does not depend in order to obtain complete families of Boolean functions even if we cannot build a specific Boolean function for each possible input size.

Lemma 2.3. *Let $f \colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. Let $n' \in \mathbb{N}$ such that $n' > n$ and let $f' \colon \{0,1\}^{n'} \to \{0,1\}$ be the Boolean function defined by $f'(\vec{a}) = f(a_1, \ldots, a_n)$ for all $\vec{a} \in \{0,1\}^{n'}$. Then, for any $V \subseteq [n]$, $r_V(f') = r_V(f)$.*

Proof. It is immediate by definition that for $V \subseteq [n]$, $s_V(f') = s_V(f)$. □

2.2 Hard functions: Indirect Storage Access functions and Element Distinctness

In this section we define two natural families of functions for which Nečiporuk's method is known to produce asymptotically optimal lower bounds for some complexity measures. The first is the Element Distinctness function.

Definition 2.1. The *Element Distinctness* function $\mathrm{ED}_{N,m}$ for $m \geq N$ is the Boolean function that takes as input $n = N \cdot \lceil \log_2 m \rceil$ bits representing N integers in $[m]$ (outputting 0 on illegal inputs) and outputs 1 iff all the N integers have distinct values. When $n = 2k \cdot 2^k$, we write ED_n for the function ED_{N,N^2} where $N = 2^k$.

The second is the family of Indirect Storage Access functions. These will turn out to be useful in a broader range of applications than the Element Distinctness function and we will see that we can characterize Nečiporuk's method in terms of the bounds it achieves for these functions.

Indirect Storage Access functions seem to have been originally defined by Paul in [16] to give an example of a family of Boolean functions for which we have a trade-off between the minimum sizes of Boolean binary formulas computing them and the minimum sizes of Boolean binary circuits computing them.

Definition 2.2. The *Indirect Storage Access* function for $k, \ell \in \mathbb{N}_{>0}$, denoted

$$\mathrm{ISA}_{k,\ell} \colon \{0,1\}^{k+2^k\ell+2^\ell} \to \{0,1\}$$

is such that for all $\vec{a} \in \{0,1\}^{k+2^k\ell+2^\ell}$, $\mathrm{ISA}_{k,\ell}(\vec{a}) = a_{\gamma(\vec{a})}$ where γ is computed from $\vec{a}$ as follows:

Let $\alpha(\vec{a})$ be the number represented in binary by the first k bits of $\vec{a}$. Let $\beta(\vec{a})$ be the number represented in binary by the sequence of ℓ bits of $\vec{a}$ starting at position $k+1+\ell\alpha(\vec{a})$. Then $\gamma(\vec{a})$ is the bit of $\vec{a}$ at position $k+1+\ell 2^k+\beta(\vec{a})$. Informally speaking, $\mathrm{ISA}_{k,\ell}$ is just a function reading a bit using two levels of addressing: a k-bit pointer selects an ℓ-bit pointer (among 2^k such pointers) that picks one bit from a 2^ℓ-bit data string.

It is known that both these families of Boolean functions yield the asymptotically strongest lower bounds obtainable using Nečiporuk's method for Boolean formula size over arbitrary binary bases, and deterministic branching program complexity [20, 2, 21]. The essence of the argument in each case is the existence of a good partition with a large count of the number of subfunctions on the variables of the partition.

Lemma 2.4. *Let $n = 2k \cdot 2^k > 0$ for $k \in \mathbb{N}$. There is a partition of $[n]$ into blocks $V_1, \ldots, V_N$ for $N = 2^k$ such that for all $i \in [N]$, $|V_i| = 2k$ and $r_{V_i}(\mathrm{ED}_n) = \binom{N^2}{N-1} + 1 \geq N^{N-1} = 2^{k(2^k-1)}$.*

Proof. Each block in the partition $V_1, \ldots, V_N$ corresponds to the bits of one of the N numbers for the ED_{N,N^2} function. Observe that for each assignment of distinct values to the $N-1$ other blocks, the subfunction induced on the i-th block must be different, since precisely those $N-1$ values must be avoided for the function to have value 1. There are $\binom{N^2}{N-1}$ possible choices of those $N-1$ distinct values; for other assignments, we get the constant 0 function. □

We now see that for different choices of k and ℓ, the function family $\mathrm{ISA}_{k,\ell}$ provides similar bounds but a more flexible range of parameters to obtain partitions of different sizes.

Definition 2.3. In the definition of $\mathrm{ISA}_{k,\ell}$, we will refer to $\alpha(\vec{a})$ and $\beta(\vec{a})$ as to the *primary* and *secondary* pointers of the $\mathrm{ISA}_{k,\ell}$ instance $\vec{a}$. The bits of the secondary pointer will be denoted $\mathrm{sec}_1, \ldots, \mathrm{sec}_\ell$, and more generally the bits of the p-th secondary pointer among the 2^k

such pointers in the instance at hand will be denoted $\mathrm{sec}[p]_1, \ldots, \mathrm{sec}[p]_\ell$ for $p \in [2^k]$. The 2^ℓ data bits will be referred to as *Data* and $Data[b_1, \ldots, b_\ell]$ will stand for the data bit at position $\mathrm{bin}_\ell(b_1, \ldots, b_\ell) + 1$. When the context is clear, bits of $\vec{a}$ will also be viewed as input variable indices.

We now see that we can partition the set of input variables of $\mathrm{ISA}_{k,\ell}$ in such a way that the number of induced subfunctions is identical and maximal for all elements of the partition but one: this is formalized in the following lemma.

Lemma 2.5. *For every $k, \ell \in \mathbb{N}_{>0}$, there exists a partition $V_1, \ldots, V_{2^k}, U$ of $[k + 2^k\ell + 2^\ell]$ such that $|V_i| = \ell$ and $r_{V_i}(\mathrm{ISA}_{k,\ell}) = 2^{2^\ell}$ for all $i \in [2^k]$.*

Proof. Let $k, \ell \in \mathbb{N}_{>0}$. Consider the partition $[k + 2^k\ell + 2^\ell] = V_1 \uplus \cdots \uplus V_{2^k} \uplus U$ where V_i is the set $\{\mathrm{sec}[i]_1, \ldots, \mathrm{sec}[i]_\ell\}$ of indices of the ℓ variables forming the i^{th} secondary pointer in the $\mathrm{ISA}_{k,\ell}$ instance $\vec{a}$. Then for each setting of the first k variables $a_1, \ldots, a_k$ of $\vec{a}$, i.e, for each value $i = \mathrm{bin}_k(a_1, \ldots, a_k)$ of the primary pointer, every possible fixing of the 2^ℓ-bit data string induces a different subfunction on V_{i+1}, hence $r_{V_{i+1}}(\mathrm{ISA}_{k,\ell}) = 2^{2^\ell}$. □

Let

$$\begin{aligned} h_{\mathrm{ISA}} \colon \mathbb{N}_{>0} &\to \mathbb{N}_{>0} \\ m &\mapsto m + 2^m(m + \lceil \log_2 m \rceil) + 2^{m + \lceil \log_2 m \rceil} \ . \end{aligned}$$

From $\mathrm{ISA}_{k,\ell}$ we define the Indirect Storage Access functions family $\mathrm{ISA} = \{\mathrm{ISA}_n\}_{n \in \mathbb{N}}$, such that for all $n \in \mathbb{N}$

- if $n < 5$, $\mathrm{ISA}_n(\vec{a}) = 0$ for all $\vec{a} \in \{0,1\}^n$;
- if there exists $k \in \mathbb{N}_{>0}$ such that $n = h_{\mathrm{ISA}}(k)$, then $\mathrm{ISA}_n = \mathrm{ISA}_{k,k+\lceil \log_2 k \rceil}$;
- otherwise, $\mathrm{ISA}_n(\vec{a}) = \mathrm{ISA}_{k',k'+\lceil \log_2 k' \rceil}(a_1, \ldots, a_{n'})$ for all $\vec{a} \in \{0,1\}^n$ where $k' = \max\{k \in \mathbb{N}_{>0} \mid h_{\mathrm{ISA}}(k) < n\}$ and $n' = h_{\mathrm{ISA}}(k')$.

ISA will be used to give, for each complexity measure we study in this paper (this notion will be precisely defined in the next subsection), an actual family of Boolean functions that achieves the best lower bound obtainable using Nečiporuk's lower bound method (to be defined later). The setting of k and ℓ in its definition is crucial, because if we would for example set $\mathrm{ISA}_n = \mathrm{ISA}_{k,k}$ for all $n \in \mathbb{N}$ such that there exists $k \in \mathbb{N}_{>0}$ verifying $n = k + 2^k k + 2^k$, we would not reach the desired bounds.

The next lemma is a simple useful adaptation of Lemma 2.5.

Lemma 2.6. *For all $n \in \mathbb{N}, n \geq 5$, there exist $p, q \in \mathbb{N}_{>0}$ verifying $p \geq \frac{1}{32} \cdot \frac{n}{\log_2 n}$ and $q \geq \frac{n}{16}$ such that there exists a partition $V_1, \ldots, V_p, U$ of $[n]$ such that $r_{V_i}(\mathrm{ISA}_n) = 2^q$ for all $i \in [p]$.*

Proof. Let $n \in \mathbb{N}, n \geq 5$. Let $k \in \mathbb{N}_{>0}$ be the unique positive integer verifying $h_{\mathrm{ISA}}(k) \leq n < h_{\mathrm{ISA}}(k+1)$. Set $n' = h_{\mathrm{ISA}}(k)$. By definition we have $\mathrm{ISA}_{n'} = \mathrm{ISA}_{k,k+\lceil \log_2 k \rceil}$. Let $V_1, \ldots, V_{2^k}, U$ be a partition of $[n']$ such that $r_{V_i}(\mathrm{ISA}_{n'}) = 2^{2^{k+\lceil \log_2 k \rceil}}$ for all $i \in [2^k]$ as given by Lemma 2.5. Moreover, by definition of ISA_n and by Lemma 2.3 (for the case in which $n' < n$), we have that $r_{V_i}(\mathrm{ISA}_n) = r_{V_i}(\mathrm{ISA}_{n'}) = 2^{2^{k+\lceil \log_2 k \rceil}}$ for all $i \in [2^k]$.

Set $q = 2^{k+\lceil \log_2 k \rceil}$ and $p = 2^k$.

A bit of elementary algebra shows:

$$n \leq h_{\mathrm{ISA}}(k+1) \leq 16 \cdot 2^{k+\lceil \log_2 k \rceil} = 16q \leq 16 \cdot 2^{k+\log_2 k+1} \leq 32kp \ .$$

Hence $q \geq \frac{n}{16}$ and $p \geq \frac{1}{32} \cdot \frac{n}{k} \geq \frac{1}{32} \cdot \frac{n}{\log_2 n}$ (as $\log_2 n \geq \log_2 2^k = k$). □

2.3 Computational models

In this subsection we define the three concrete models of computation considered extensively in this paper. But first, in view of defining a model-independent notion of Nečiporuk's method, we define a complexity measure merely as a function that associates a non-negative integer to each Boolean function, as follows.

Definition 2.4. A *complexity measure on Boolean functions* is a function

$$\mathbf{M}\colon \bigcup_{n\in\mathbb{N}} \{0,1\}^{\{0,1\}^n} \to \mathbb{N}\,. \tag{1}$$

Note that the models of computation we consider here are *non-uniform* in the sense that each computing device only processes inputs of a fixed length. These models are the following:

- the nondeterministic branching program (NBP),
- the parity branching program ($\oplus$BP),
- the δ-limited nondeterministic branching program (δ-LNBP) and
- the δ-limited nondeterministic Boolean formula (δ-LNBF).

The nondeterministic branching program is well known to capture nondeterministic logspace NL when restricted to polynomial size [17]; similarly, when restricted to polynomial size parity branching programs capture $\oplus\mathsf{L}$. The two other models are motivated by the well-known observation that unrestricted nondeterministic Boolean formulas capture NP (see [4]) and further by Klauck's analysis of restricted nondeterministic fomulas [10]. Both branching program models extend, albeit in different ways, the deterministic branching program model known to capture deterministic logspace [3, 13].

Definition 2.5. A *(nondeterministic) branching program (NBP)* on $\{0,1\}^V$, for a set V of variables, is a tuple $P = (X, s, t_0, t_1, A_0, A_1, var)$ where

- X is a finite set of vertices (or states);
- $s \in X$ is the start (or source) vertex;
- $t_0, t_1 \in X$, $t_0 \neq t_1$ are two distinct sink vertices;
- $A_0 \subseteq X \setminus \{t_0, t_1\} \times X \setminus \{s\}$ is the set of arcs labelled 0;
- $A_1 \subseteq X \setminus \{t_0, t_1\} \times X \setminus \{s\}$ is the set of arcs labelled 1;
- $var\colon X \setminus \{t_0, t_1\} \to V$ labels each non-sink vertex.

Definition 2.6. For a nondeterministic branching program $P = (X, s, t_0, t_1, A_0, A_1, var)$ on $\{0,1\}^V$, each assignment $\vec{a} \in \{0,1\}^V$ defines a set of arcs $A[\vec{a}] = \{(u,v) \in A_0 \mid a_{var(u)} = 0\} \cup \{(u,v) \in A_1 \mid a_{var(u)} = 1\}$ and thus a graph $P[\vec{a}] = (X, A[\vec{a}])$. P computes a Boolean function $f\colon \{0,1\}^V \to \{0,1\}$ given by $f(\vec{a}) = 1$ if and only if there exists a path (*computation*) in $P[\vec{a}]$ from state s to state t_1.

A branching program P defined as above can also be interpreted as a *parity branching program* that computes a Boolean function $f^{\oplus}\colon \{0,1\}^V \to \{0,1\}$ where $f^{\oplus}(\vec{a}) = 1$ if and only if the total number of paths in $P[\vec{a}]$ from state s to state t_1 is odd.

Definition 2.7. Branching program P is *deterministic* if and only if $(X, A_0 \cup A_1)$ is acyclic and A_0 and A_1 each contain precisely one out-arc from each non-sink vertex of X. P is a *δ-limited nondeterministic branching program* for $f\colon \{0,1\}^V \to \{0,1\}$ if and only if P is a deterministic branching program computing a function $f'\colon \{0,1\}^{V'} \to \{0,1\}$ with $V \subseteq V'$ and $|V' \setminus V| = \delta$ such that $f(\vec{a}) = \bigvee_{\vec{b} \in \{0,1\}^{V' \setminus V}} f'(\vec{a}, \vec{b})$.

Definition 2.8. The complexity measure $\mathbf{M}(f)$ is denoted $\mathbf{NBP}(f)$, $\oplus\mathbf{BP}(f)$, $\mathbf{LNBP}_\delta(f)$ and $\mathbf{BP}(f)$ for nondeterministic, parity, δ-limited nondeterministic and deterministic branching programs respectively. $\mathbf{M}(f)$ is defined in each case as the minimum, over every BP of the appropriate type computing f, of the number of non-sink states in (a.k.a. *the size of*) the BP.

Modulo cosmetic details, the above are standard definitions of deterministic, nondeterministic and parity Boolean BPs [21]. The definition of δ-limited nondeterministic BPs, which does not appear to have been studied previously, is inspired by notions of limited nondeterminism for other models [4, 6, 10].

Remark 2.1. The limited nondeterminism of the δ-LNBP model is formulated in a framework of verification of explicitly represented guesses that is typical for time-bounded nondeterminism. In contrast, the NBP model only represents nondeterministic guesses implicitly, which allows them to be used without being stored, as is typical for space-bounded computation. In particular, even if δ is unbounded (say $\delta = \infty$), the smallest ∞-LNBP could be somewhat larger than that of an equivalent NBP and vice-versa. It is not difficult to see that an NBP of size s can be simulated by such an ∞-LNBP of size at most $2s^2$. Indeed, simulating the k-way branch at a given state in this NBP in an ∞-LNBP can be made by accessing $\lceil \log_2 k \rceil$ fresh nondeterministic bits in a decision tree of size at most k; so since each of the s states of our original NBP branches to at most s different states for each of the possible values 0 or 1, we get that we can simulate it by an ∞-LNBP of size at most $2s^2$. Conversely, however, it is unclear by how much the size would increase when simulating an ∞-LNBP by an NBP, but it is widely conjectured to grow exponentially, since one can prove that polynomial size ∞-LNBPs capture (non-uniform) NP, while polynomial size NBPs capture (non-uniform) NL. Hence, the reader should keep in mind that LNBPs are not a restricted variant of NBPs.

Remark 2.2. Two other models comparable to the NBP are *switching networks* (also called *contact schemes*), and the more general *switching-and-rectifier (RS) networks* (see [18, 8]). The graph of an RS network is almost the same as that of an NBP except that its vertices remain unlabelled and each of its arcs is labelled by some literal or remains unlabelled, with the acceptance condition that of the NBP. (Switching networks are a special case of RS networks whose graphs are undirected.) The size of an RS network is the number of its labelled arcs. Note that Jukna [8] calls RS networks "nondeterministic branching programs", but we observe, as did Razborov [18], that the size measure of RS networks and the size measure of NBPs used in this paper are the same to within a constant factor. Indeed, one can simulate NBPs by RS networks of at most twice the size – each NBP vertex becomes an RS vertex with two new successors (one reached by an arc labelled 0, the other reached by an arc labelled 1) which have unlabelled arcs pointing to the corresponding successor vertices in the NBP. Conversely, RS networks can be simulated by NBPs of the same size. It can first be assumed w.l.o.g. that any vertex in the RS network only has unlabelled arcs and arcs labelled by literals from the same variable (any vertex having arcs labelled by literals from several different variables can be replaced at no extra cost by a set of vertices connected by unlabelled arcs having the property). Then, for each vertex u in the RS network that has out-arcs labelled x_i or $\neg x_i$, we have a vertex u' in the NBP labelled i and for each arc e from u to v in the RS network,

- if e is unlabelled, we have two arcs (u', w') in the NBP, one labelled 0, the other labelled 1, for each vertex w in the RS network that has labelled out-arcs and is reachable from u

via an unlabelled path starting with e (that uses unlabelled arcs and whose intermediate vertices do not have labelled out-arcs);

- if e is labelled ℓ
  - and v has labelled out-arcs, we have an arc (u', v') in the NBP labelled accordingly (0 if $\ell = \neg x_i$ and 1 if $\ell = x_i$),
  - otherwise v only has unlabelled out-arcs, and we have an arc (u', w') in the NBP labelled accordingly (0 if $\ell = \neg x_i$ and 1 if $\ell = x_i$) for each vertex w in the RS network that has labelled out-arcs and is reachable from v via an unlabelled path.

Span programs [9] can be simulated by parity branching programs of at most twice the size – their size is also at most polynomial in parity branching program size.

Definition 2.9 (Deterministic and δ-limited nondeterministic formulas, following [10] and [8])**.** A (deterministic) *Boolean binary formula (BF)* φ on $\{0,1\}^n$ ($n \in \mathbb{N}$) is a binary tree with

- a single root,
- every internal node of in-degree 2,
- every internal node labelled by a function $g\colon \{0,1\}^2 \to \{0,1\}$,
- every leaf labelled by one of $0, 1, x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n$.

φ computes a Boolean function f_φ on $\{0,1\}^n$ in the natural way by function composition. Let $\delta \in \mathbb{N}$. A *δ-limited nondeterministic binary formula (δ-LNBF)* on $\{0,1\}^n$ φ is a deterministic binary formula φ' on $\{0,1\}^{n+\delta}$. It computes a Boolean function f_φ such that for $\vec{a} \in \{0,1\}^n$, $f_\varphi(\vec{a}) = \bigvee_{\vec{b}\in\{0,1\}^\delta} f_{\varphi'}(\vec{a}, \vec{b})$.

Definition 2.10. The complexity measure $\mathbf{M}(f)$ for deterministic and δ-limited nondeterministic formulas is denoted $\mathbf{L}(f)$ and $\mathbf{LL}_\delta(f)$ respectively. In each case $\mathbf{M}(f)$ is defined as the minimum, over every formula ϕ of the appropriate type computing f, of the number $|\phi|$ of non-constant leaves in (a.k.a. *the size of*) ϕ.

Lemma 2.7. *Let $\delta \in \mathbb{N}$ and let $g\colon \{0,1\}^{n+\delta} \to \{0,1\}$ and let $f\colon \{0,1\}^n \to \{0,1\}$ be given by $f(\vec{a}) = \bigvee_{\vec{b}\in\{0,1\}^\delta} g(\vec{a}, \vec{b})$. Then, for all $V \subseteq [n]$, we have $r_V(f) \leq B(r_V(g), 2^\delta) - 1 < r_V(g)^{2^\delta}$ where $B(m, r) = \sum_{i=0}^{r} \binom{m}{i}$ is the volume of the Hamming ball of radius r in $\{0,1\}^m$.*

Proof. Let $V \subseteq [n]$. For $\rho \in \{0,1\}^{[n]\setminus V}$, by definition, $f|_\rho = \bigvee_{\vec{b}\in\{0,1\}^\delta} g|_{\rho\vec{b}}$. Since $\rho\vec{b}$ assigns all variables in $[n+\delta] \setminus V$, each function $g|_{\rho\vec{b}}$ is in $s_V(g)$. Therefore, each $f|_\rho \in s_V(f)$ is the $\bigvee$ of 2^δ functions in $s_V(g)$ (not necessarily distinct). Therefore over all choices of ρ, the function $f|_\rho$ only depends on the set of between 1 and 2^δ among these subfunctions of g that are distinct (and not what values $\vec{b}$ with which each such subfunction is associated). Therefore there are at most $B(r_V(g), 2^\delta) - 1$ possible distinct subfunctions $f|_\rho$ in $s_V(f)$. □

3 Nondeterministic Branching Program Lower Bounds via Shannon Bounds

In this section we describe the simplest form of Nečiporuk's technique and its applications in order to give some intuition about the technique. Readers may prefer to skip to the generalised abstract definition of Nečiporuk's method in Section 4. The simple version here is based on the

so-called "Shannon bounds" for a complexity measure. The Shannon function for a complexity measure maps n to the maximum complexity of any Boolean function over $\{0,1\}^n$ in that measure. Lower bounds on the Shannon function typically follow by a simple enumeration of the number of distinct functions of bounded measure.

For all $n, s \in \mathbb{N}$, and $\mathbf{M}$ a complexity measure let us denote by $\mathbf{M}_{sem}(n,s)$ the number of distinct n-ary Boolean functions of complexity measure at most s. In particular, define N_{sem} to be the function $\mathbf{M}_{sem}$ for NBPs and $\oplus_{sem}$ be that for $\oplus$BPs. The next lemma is the core of the simple version of Nečiporuk's technique.

Lemma 3.1. *For any $n \in \mathbb{N}$, for any n-ary Boolean function f on V that depends on all its inputs and any partition $V_1, \ldots, V_p$ of V we have*

$$\mathbf{NBP}(f) \geq \sum_{i=1}^{p} \max\big\{|V_i|, \min\{s \in \mathbb{N} \mid N_{sem}(|V_i|, s) \geq r_{V_i}(f)\}\big\}.$$

$$\oplus\mathbf{BP}(f) \geq \sum_{i=1}^{p} \max\big\{|V_i|, \min\{s \in \mathbb{N} \mid \oplus_{sem}(|V_i|, s) \geq r_{V_i}(f)\}\big\}.$$

Proof. Let $n \in \mathbb{N}$, f be an n-ary Boolean function on V depending on all its inputs and $V_1, \ldots, V_p$ a partition of V. Let P be a Boolean NBP computing f. For all $i \in [p]$ we will denote by $s_i \in \mathbb{N}$ the number of vertices in P labelled by elements in V_i. It is clear that P is of size at least $\sum_{i=1}^{p} s_i$.

Let $i \in [p]$. Observe that for every subfunction g of f on V_i, there is by definition a partial assignment ρ on $V \setminus V_i$ such that $f|_\rho = g$, so it is not too difficult to see that g is computed by the Boolean NBP of size s_i obtained from P by:

1. removing all non sink vertices labelled by elements not in V_i;
2. defining the new start vertex as one of the vertices whose label is in V_i and connected to the start vertex of P by a path of nodes labelled by a variable outside of V_i and arcs labelled consistently with ρ and adding both an arc labelled 0 and an arc labelled 1 from this new start vertex to each other such reachable vertex (except for the extreme case of a constant function, in which we just set the start vertex as the appropriate sink vertex);
3. connecting a vertex u to a vertex v by an arc labelled by $a \in \{0,1\}$ if and only if there exists a path from u to v in P verifying that any intermediate vertex of the path is labelled by a variable outside of V_i, the first arc is labelled by a and each arc (but the first one) is labelled consistently with ρ.

Thus, $r_{V_i}(f)$ is necessarily upper-bounded by the number of semantically distinct such NBPs we can build from P that way, which is in turn at most $N_{sem}(|V_i|, s_i)$. Moreover, since, by construction, f depends on all variables whose indices are in V_i, we have that for each element $\ell \in V_i$, P contains at least one vertex labelled by ℓ, so $s_i \geq |V_i|$. Hence, for each i, $s_i \geq \max\big\{|V_i|, \min\{s \in \mathbb{N} \mid N_{sem}(|V_i|, s) \geq r_{V_i}(f)\}\big\}$. Since the NBP has size at least $\sum_{i=1}^{p} s_i$ and the NBP is arbitrary, the bound of the lemma follows.

The same argument also applies directly to yield the bound for $\oplus$BPs, with $\oplus_{sem}(|V_i|, s)$ replacing $N_{sem}(|V_i|, s)$. □

Proposition 3.2. *Let $s \geq n$. Then $N_{sem}(n,s), \oplus_{sem}(n,s) < 2^{2(s+1)^2}$.*

Proof. We simply count the number of distinct branching programs. Subject to renaming and reorganising, any n-ary Boolean function computable by an NBP or $\oplus$BP of size at most s, can be computed by one of size exactly s, having $\{v_j\}_{j=1}^{s+2}$ as vertices, v_1 as start vertex, v_{s+1} as 0-vertex and v_{s+2} as 1-vertex, where no arc goes to the 0-vertex (that is, since what matters for the computation of both an NBP and a $\oplus$BP is the number of paths from the start vertex to the 1-vertex, arcs ending in the 0-vertex are unnecessary). The out-arcs at each node v_i can be described by the subset of vertices v_j, $j \neq i$ and $j \neq v_{s+1}$, reached on each of values 0 and 1. There are $(s-1)!$ different ways of reordering the names of vertices $v_2, \ldots, v_s$ that keep identical connectivity of the branching program and hence the function it computes, both as an NBP and a $\oplus$BP. Hence, it directly follows that $N_{sem}(n,s), \oplus_{sem}(n,s) \leq (2^{2s}n)^s/(s-1)! \leq 2^{2s^2}s^s/(s-1)!$, since $s \geq n$, therefore, since $s! \geq (s/e)^s$, $N_{sem}(n,s), \oplus_{sem}(n,s) \leq 2^{2s^2}se^s < 2^{2(s+1)^2}$. □

Definition 3.1. For the complexity measures $\mathbf{M} = \mathbf{NBP}, \oplus\mathbf{BP}$, the *simple Nečiporuk lower bound method* consists of the following.

1. Giving explicitly a non-decreasing function $b\colon \mathbb{N}_{>0} \to \mathbb{N}$ such that for any $n \in \mathbb{N}$, for any n-ary Boolean function f on V that depends on all its inputs and any partition $V_1, \ldots, V_p$ of V, we have $\sum_{i=1}^{p} \max\{|V_i|, \min\{s \in \mathbb{N} \mid \mathbf{M}_{sem}(|V_i|, s) \geq r_{V_i}(f)\}\} \geq \sum_{i=1}^{p} b(r_{V_i}(f))$.
2. For a given n-ary Boolean function g on V that depends on all the variables whose indices are in V, explicitly choosing a partition $V_1, \ldots, V_p$ of V, computing $r_{V_i}(g)$ for all $i \in [p]$ and concluding that $\mathbf{M}(g) \geq \sum_{i=1}^{p} b(r_{V_i}(g))$.

A function b satisfying the condition of Step 1 in the definition above is called a *simple Nečiporuk function for* $\mathbf{M}$.

We now give an explicit simple Nečiporuk function for **NBP** and $\oplus$**BP**.

Proposition 3.3. *The function on* $\mathbb{N}_{>0} \to \mathbb{N}$ *given by* $m \mapsto \left\lceil \sqrt{\frac{1}{2}\log_2 m} - 1 \right\rceil$ *is a simple Nečiporuk function for* **NBP** *and for* $\oplus$**BP**.

Proof. We start by observing that the function on $\mathbb{N}_{>0} \to \mathbb{N}$ given by $m \mapsto \left\lceil \sqrt{\frac{1}{2}\log_2 m} - 1 \right\rceil$ is obviously non-decreasing. Let $n \in \mathbb{N}$, f be an n-ary Boolean function f on V depending on all its variables and $V_1, \ldots, V_p$ be a partition of V. Let $i \in [p]$. Let $s_i = \max\{|V_i|, \min\{s \in \mathbb{N} \mid N_{sem}(|V_i|, s) \geq r_{V_i}(f)\}\}$ for all $i \in [p]$. We claim that $s_i \geq \left\lceil \sqrt{\frac{1}{2}\log_2 r_{V_i}(f)} - 1 \right\rceil$ for all $i \in [p]$.

By definition, $N_{sem}(|V_i|, s_i) \geq r_{V_i}(f)$, and since $s_i \geq |V_i|$, Proposition 3.2 implies that $N_{sem}(|V_i|, s_i) < 2^{2(s_i+1)^2}$ and hence $2^{2(s_i+1)^2} > r_{V_i}(f)$, that is to say, $s_i > \sqrt{\frac{1}{2}\log_2 r_{V_i}(f)} - 1$. Since s_i is integral, we deduce that $s_i \geq \left\lceil \sqrt{\frac{1}{2}\log_2 r_{V_i}(f)} - 1 \right\rceil$. The lemma follows for **NBP**; the argument for $\oplus$**BP** is identical replacing N_{sem} by $\oplus_{sem}$. □

This directly gives us the following lower bounds.

Proposition 3.4 ([17, 9])**.** $\mathbf{NBP}(\mathrm{ED}_n), \mathbf{NBP}(\mathrm{ISA}_n), \oplus\mathbf{BP}(\mathrm{ED}_n), \oplus\mathbf{BP}(\mathrm{ISA}_n) \in \Omega\left(\frac{n^{3/2}}{\log_2 n}\right)$.

Proof. We first consider ED_n for $n = 2k2^k$ and $k \geq 2$. By Lemma 2.4 there is a partition $V_1, \ldots, V_N$ of $[n]$ for $N = 2^k$ such that $r_{V_i}(\mathrm{ED}_n) \geq N^{N-1}$ and $|V_i| = 2k$ for all $i \in [N]$. Applying Proposition 3.3, since ED_n depends on all its variables, we have

$$\begin{aligned}\mathbf{NBP}(\mathrm{ED}_n), \oplus\mathbf{BP}(\mathrm{ED}_n) &\geq \sum_{i=1}^{N} \left\lceil \sqrt{\frac{1}{2}\log_2 r_{V_i}(\mathrm{ED}_n)} - 1 \right\rceil \geq N \cdot \left\lceil \sqrt{\frac{1}{2}\log_2 N^{N-1}} \right\rceil - N \\ &\geq N \cdot \sqrt{\frac{N-1}{2}\log_2 N} - N\end{aligned}$$

which is in $\Omega(N^{3/2}(\log_2 N)^{1/2})$ and hence $\Omega\Big(\frac{n^{3/2}}{\log_2 n}\Big)$ since n is $O(N \log_2 N)$.

We now consider ISA. Let $n \in \mathbb{N}, n \geq 32$. Let $V_1, \ldots, V_p, U$ be a partition of $[n]$ such that $r_{V_i}(\mathrm{ISA}_n) = 2^q$ for all $i \in [p]$ where $p, q \in \mathbb{N}_{>0}$ verify $p \geq \frac{1}{32} \cdot \frac{n}{\log_2 n}$ and $q \geq \frac{n}{16}$ as given by Lemma 2.6. Applying Proposition 3.3, since ISA_n depends on all its variables, we have

$$\begin{aligned}
\mathbf{NBP}(\mathrm{ISA}_n), \oplus\mathbf{BP}(\mathrm{ISA}_n) &\geq \sum_{i=1}^{p} \left\lceil \sqrt{\frac{1}{2} \log_2 r_{V_i}(\mathrm{ISA}_n)} - 1 \right\rceil \\
&\geq \sum_{i=1}^{p} \Big(\sqrt{\frac{1}{2} \log_2(2^q)} - 1 \Big) = p \cdot \Big(\sqrt{\frac{q}{2}} - 1 \Big) \\
&\geq \frac{1}{32} \cdot \frac{n}{\log_2 n} \cdot \Big(\sqrt{\frac{n}{32}} - 1 \Big) .
\end{aligned}$$

So $\mathbf{NBP}(\mathrm{ISA}_n), \oplus\mathbf{BP}(\mathrm{ISA}_n) \in \Omega\Big(\frac{n^{3/2}}{\log_2 n}\Big)$. □

To understand the best lower bounds we can prove with the simple Nečiporuk lower bound method, we first give a lower bound on $N_{sem}(n, s)$ and $\oplus_{sem}(n, s)$ (valid for suitable values of $n, s \in \mathbb{N}$) that will allow us to give an upper bound on all simple Nečiporuk functions for **NBP**, $\oplus$**BP**. We do this by giving an easy upper bound on the size needed by NBPs and $\oplus$BPs to compute any n-ary Boolean function[1]; i.e., simple upper bounds on the Shannon function for **NBP** and $\oplus$**BP**.

Lemma 3.5. *For any n-ary Boolean function f on $\{0,1\}^n$ ($n \in \mathbb{N}$), $\mathbf{NBP}(f), \oplus\mathbf{BP}(f) \leq 3 \cdot 2^{\lceil \frac{n}{2} \rceil}$.*

Proof. Assume that $n = 2t$ is even. The constructed NBPs will have only one nondeterministic level, will be the same for all functions for the other levels 1 to $t-1$ and $t+1$ to $2t$, and every vertex at each level i will query variable x_i.

The first $t-1$ levels form a complete decision tree of height $t-1$ on variables $x_1, \ldots, x_{t-1}$ with a vertex at level t for each assignment $a_1 \ldots a_{t-1}$ to these variables. The last t levels of the NBP consist of a complete fan-in tree of height t on variables $x_{t+1}, \ldots, x_{2t}$ as follows: there is a vertex at level $t' > t$ for every assignment $a_{t'} \ldots a_{2t}$ to $x_{t'}, \ldots, x_t$ and there is an out-arc labelled $a_{t'}$ from this vertex to the vertex at level $t'+1$ corresponding to $a_{t'+1} \ldots a_{2t}$. The 1-output vertex has two in-arcs, one labelled a_{2t} from each vertex a_{2t} at level $2t$.

Finally, we define the nondeterministic level t of the NBP for function f. For each assignment $a_1 \ldots a_{2t}$ on which f evaluates to 1, there is an out-arc labelled a_t from the vertex corresponding to $a_1 \ldots a_{t-1}$ at level t (which queries x_t) to the vertex corresponding to $a_{t+1} \ldots a_{2t}$ at level $t+1$.

The constructed NBP has at most $3 \cdot 2^t = 3 \cdot 2^{\frac{n}{2}}$ vertices. By observing that there is precisely one accepting path on any accepted input, we see that it is also a $\oplus$BP. □

Corollary 3.6. *For all $n, s \in \mathbb{N}$, $n \geq 2\left\lfloor \log_2(\frac{s}{3}) \right\rfloor$, $N_{sem}(n, s), \oplus_{sem}(n, s) > 2^{\frac{s^2}{36}}$.*

Proof. Clearly $N_{sem}(n, s)$ is non-decreasing in n, so it suffices to prove the corollary for $n = 2\left\lfloor \log_2(\frac{s}{3}) \right\rfloor$. Then $3 \cdot 2^{\frac{n}{2}} \leq s < 6 \cdot 2^{\frac{n}{2}}$. There are precisely $2^{2^n} > 2^{\frac{s^2}{36}}$ different Boolean functions on n inputs and, by Lemma 3.5, each may be computed by an NBP or $\oplus$BP of size at most s. □

[1]Note that there are somewhat tighter but more complicated upper bounds of $2^{n/2+1}$ for **NBP** due to Lupanov [11] and a tight asymptotic upper bound of $2^{(n+1)/2}$ for $\oplus$**BP** due to Nečiporuk [14], respectively; see [8].

Theorem 3.7. *Let $F = \{f_n\}_{n\in\mathbb{N}}$ be a family of Boolean functions. Let $L\colon \mathbb{N} \to \mathbb{N}$ be such that for each $n \in \mathbb{N}$, the lower bound $L(n)$ for $\mathbf{NBP}(f_n)$ or $\oplus\mathbf{BP}(f_n)$ has been obtained using the simple Nečiporuk lower bound method. Then, $L(n) \in \mathrm{O}\Big(\frac{n^{3/2}}{\log_2 n}\Big)$.*

Proof. Let $F = \{f_n\}_{n\in\mathbb{N}}$ be a family of Boolean functions, where for each $n \in \mathbb{N}$, f_n depends on all the variables in $D_n \subseteq [n]$. Let $L\colon \mathbb{N} \to \mathbb{N}$ be such that

$$L(n) = \max\Big\{\sum_{i=1}^{p} \max\big\{|V_i|, \min\{s \in \mathbb{N} \mid N_{sem}(|V_i|, s) \geq r_{V_i}(f_n)\}\big\} \;\Big|\; V_1, \ldots, V_p \text{ partition } D_n\Big\}$$

for all $n \in \mathbb{N}$.

Let $n \in \mathbb{N}$ and $V_1, \ldots, V_p$ be a partition of D_n. Let $i \in [p]$ and set

$$s_i = \max\big\{|V_i|, \min\{s \in \mathbb{N} \mid N_{sem}(|V_i|, s) \geq r_{V_i}(f_n)\}\big\}.$$

Suppose that $s_i = \min\{s \in \mathbb{N} \mid N_{sem}(|V_i|, s) \geq r_{V_i}(f_n)\} > |V_i|$. We now have two cases depending on $|V_i|$. If $|V_i| < \log_2\log_2 r_{V_i}(f) + 3$ then, by Lemma 3.5, since NBPs of size $3 \cdot 2^{\left\lceil \frac{|V_i|}{2} \right\rceil}$ suffice to compute all functions on V_i, which include those counted in $r_{V_i}(f_n)$,

$$s_i \leq 3 \cdot 2^{\left\lceil \frac{|V_i|}{2} \right\rceil} \leq 3 \cdot 2^{(\log_2\log_2 r_{V_i}(f))/2+2} = 12\sqrt{\log_2 r_{V_i}(f)}.$$

On the other hand, if $|V_i| \geq \log_2\log_2 r_{V_i}(f) + 3$ then, setting $s = \Big\lceil 6\sqrt{\log_2 r_{V_i}(f)} \Big\rceil$, we have

$$2\log_2\Big(\frac{s}{3}\Big) \leq 2\log_2\Big(\frac{6\sqrt{\log_2 r_{V_i}(f)}+1}{3}\Big) \leq 2\log_2\Big(\frac{7}{3}\sqrt{\log_2 r_{V_i}(f)}\Big) \leq \log_2\log_2 r_{V_i}(f) + 3 \leq |V_i|$$

so, by Corollary 3.6, we have that $N_{sem}(|V_i|, s) > 2^{\frac{s^2}{36}} \geq r_{V_i}(f)$, which means that $s_i \leq \Big\lceil 6\sqrt{\log_2 r_{V_i}(f)} \Big\rceil$. Therefore, for all $n \in \mathbb{N}$,

$$L(n) \leq \max\left\{\sum_{i=1}^{p} \max\Big\{|V_i|, 12\sqrt{\log_2 r_{V_i}(f)}\Big\} \;\middle|\; V_1, \ldots, V_p \text{ partition } D_n\right\}.$$

Let $n \in \mathbb{N}, n \geq 4$. By Lemma 2.1, it follows that

$$\begin{aligned}
L(n) &\leq \max\left\{\sum_{i=1}^{p} \max\Big\{|V_i|, 12\sqrt{\log_2(\min\{2^{2^{|V_i|}}, 2^{n-|V_i|}\})}\Big\} \;\middle|\; V_1, \ldots, V_p \text{ partition } D_n\right\} \\
&= \max\left\{\sum_{i=1}^{p} \max\Big\{v_i, 12\sqrt{\log_2(\min\{2^{2^{v_i}}, 2^{n-v_i}\})}\Big\} \;\middle|\; \sum_{i=1}^{p} v_i \leq n \text{ and } \forall i \in [p], v_i > 0\right\} \\
&\leq \max\left\{\sum_{i=1}^{p} \max\Big\{v_i, 12\sqrt{\min\{2^{v_i}, n-v_i\}}\Big\} \;\middle|\; \sum_{i=1}^{p} v_i = n \text{ and } \forall i \in [p], v_i > 0\right\} \\
&= \max\left\{\sum_{i=1}^{p} h(v_i) \;\middle|\; \sum_{i=1}^{p} v_i = n \text{ and } \forall i \in [p], v_i > 0\right\} \qquad (\star\star)
\end{aligned}$$

where $h(v) = \max\big\{v, 12\min\{2^{v/2}, \sqrt{n-v}\}\big\}$ for all $v \in \mathbb{N}$.

Let now $v_1, \ldots, v_p$ realise the maximum in $(\star\star)$. Clearly, $h(v) = 12 \cdot 2^{v/2}$ for all $v \in \mathbb{N}, v \leq \log_2 n - 1$ and hence $h(v) + h(v') \leq h(v + v')$ if $v + v' \leq \log_2 n - 1$. It follows that without loss

of generality we can assume that there exists at most one $j \in [p]$ such that v_j is smaller than $\frac{\log_2 n-1}{2}$. Such a small v_j has $h(v_j) = 12 \cdot 2^{v_j/2} < 12 \cdot n^{1/4}$. Let now $I \subseteq [p]$ such that $i \in I$ if and only if $h(v_i) = v_i$. We have that $\sum_{i \in I} h(v_i) \leq n$ by definition of $v_1, \ldots, v_p$. Moreover, in $[p] \setminus I$, there are at most $\frac{2n}{\log_2 n-1}$ elements, since for all $i \in I \setminus \{j\}$, $v_i \geq \frac{\log_2 n-1}{2}$, and each such i verifies $h(v_i) \leq 12\sqrt{n - v_i} \leq 12\sqrt{n}$. Hence,

$$L(n) \leq \sum_{i=1}^{p} h(v_i) \leq 24 \cdot \frac{n^{3/2}}{\log_2 n - 1} + 12 \cdot n^{1/4} + n \leq 74 \cdot \frac{n^{3/2}}{\log_2 n}$$

which completes the proof, as the case of $\oplus$BPs is treated identically. □

Limitations of this Formulation

Simply using some adaptation of Definition 3.1 would not allow us to recover the well-known $\Omega\big(\frac{n^2}{\log n}\big)$ lower bound for size of binary formulas contained in Nečiporuk's original article [15]. Indeed, for all $n, s \in \mathbb{N}$, let us denote by $F_{sem}(n, s)$ the number of n-ary Boolean functions on some fixed V computable by BFs of size at most s. We can prove a Lemma analogous to Lemma 3.1 where **NBP** is replaced by **L** and N_{sem} by F_{sem}. Similarly, we can define the *simple Nečiporuk lower bound method* for **L** as in Definition 3.1, as well as *simple Nečiporuk functions for* **L** accordingly. However, Lupanov showed ([12], see [8, p.32]) that for all $n \in \mathbb{N}$, any n-ary Boolean function on some V can be computed by a BF of size at most $\alpha \cdot \frac{2^n}{\log_2 n}$ for some constant $\alpha \in \mathbb{R}_{>0}$ (a result which is analogous to Lemma 3.5). Following the same strategy as for the proofs of Corollary 3.6 and Theorem 3.7, we can show that this implies there exists a constant $\beta \in \mathbb{R}_{>0}$ such that any simple Nečiporuk function $b\colon \mathbb{N}_{>0} \to \mathbb{N}$ for **L** verifies $b(m) \leq \beta \cdot \frac{\log_2 m}{\log_2 \log_2 \log_2 m}$ for any sufficiently large $m \in \mathbb{N}_{>0}$. This means that this does not allow us to recover the well-known Nečiporuk bounding function of $m \mapsto \frac{1}{4} \log_2 m$ (see e.g. [8, Theorem 6.16]), and therefore also not Nečiporuk's original lower bound.

Even if we managed to adapt Definition 3.1 to the case of binary formulas, we cannot really do it in a clean way for all complexity measures we would like to study. If we were to try to adapt Lemma 3.1 to the case of the size of limited nondeterministic branching programs (LNBPs), we would define, as usual, for all $n, s, \delta \in \mathbb{N}$, the number $LN_{sem}(n, s, \delta)$ of n-ary Boolean functions on some fixed V computable by LNBPs of size at most s and using δ nondeterministic bits. But then, it would be false to say that for any $\delta, n \in \mathbb{N}$, for any n-ary Boolean function f depending on all of V and any partition $V_1, \ldots, V_p$ of V we have $\mathbf{LNBP}_\delta(f) \geq \sum_{i=1}^{p} \max\big\{|V_i|, \min\{s \in \mathbb{N} \mid LN_{sem}(|V_i|, s, \delta) \geq r_{V_i}(f)\}\big\}$ (this would induce an overcount, as we would most certainly count vertices corresponding to nondeterministic variables several times).

These considerations led us to the more general formulation of the Nečiporuk method described in the next section.

4 An abstract formulation of Nečiporuk's method

In this section we present an abstract version of Nečiporuk's lower bound method and provide some model-independent meta-results on the limitations of this method.

The main idea of the general version of the method is, for a given Boolean function, to partition its set of input variables and to lower bound its complexity by a sum over each element of the partition of a partial cost that depends only on the number of subfunctions of the function on the variables in this element. More formally, we state the method in the following way.

Definition 4.1. For a given complexity measure $\mathbf{M}$ on Boolean functions, *Nečiporuk's lower bound method* consists of the following.

1. Giving explicitly a non-decreasing function $b\colon \mathbb{N}_{>0} \to \mathbb{N}$ such that for any $n \in \mathbb{N}$, for any n-ary Boolean function f and any partition $V_1, \ldots, V_p$ of $[n]$, we have $\mathbf{M}(f) \geq \sum_{i=1}^{p} b(r_{V_i}(f))$.
2. For a given n-ary Boolean function g, explicitly choosing a partition $V_1, \ldots, V_p$ of $[n]$, computing $r_{V_i}(g)$ for all $i \in [p]$ and concluding that $\mathbf{M}(g) \geq \sum_{i=1}^{p} b(r_{V_i}(g))$.

A function b satisfying the condition of Step 1 in Nečiporuk's method is called a *Nečiporuk function for* $\mathbf{M}$ and we denote by $\mathcal{N}_{\mathbf{M}}$ the *set of all Nečiporuk functions for* $\mathbf{M}$.

The first step of Definition 4.1 is usually not included in the Nečiporuk method. For instance in [20, 8], an explicit Nečiporuk function b is given for a complexity measure $\mathbf{M}$ and therefore the result concerning the limitation of the method is relative to this function b. In the case of deterministic branching programs, the best possible b was given by Alon and Zwick [1], who use a similar definition but we are not aware of any result of this kind for other complexity measures.

It follows from Definition 4.1 that the best lower bound achievable by the Nečiporuk method for a family $F = \{f_n\}_{n\in\mathbb{N}}$ of Boolean functions and a complexity measure $\mathbf{M}$ is the function $\mathrm{N}_F^{\mathbf{M}}$:

$$n \mapsto \max\Big\{\sum_{i=1}^{p} b(r_{V_i}(f_n)) \;\Big|\; b \in \mathcal{N}_{\mathbf{M}} \wedge V_1, \ldots, V_p \text{ partition of } [n]\Big\} \tag{2}$$

4.1 Meta-results on Nečiporuk's method

We now give two results concerning Nečiporuk's method depending on hypotheses on the complexity measure $\mathbf{M}$. We will apply those results in the next section with the appropriate constants and functions for each of the concrete computational models we consider in this paper.

The first meta-result is that an upper bound on the complexity of the functions $\mathrm{ISA}_{k,k}$ implies an upper bound on every $b \in \mathcal{N}_{\mathbf{M}}$. Intuitively this is possible because by definition, b entails a lower bound on $\mathbf{M}(f)$ for every function f.

Lemma 4.1. *Let* $\mathbf{M}$ *be a given complexity measure on Boolean functions and assume that we have a non-decreasing function* $g_{\mathbf{M}}\colon [1, +\infty[\to \mathbb{R}_{\geq 0}$ *such that* $\mathbf{M}(\mathrm{ISA}_{k,k}) \leq g_{\mathbf{M}}(k)$ *for all* $k \in \mathbb{N}_{>0}$ *and there exists a constant* $\alpha \in \mathbb{R}_{>0}$ *such that* $\frac{g_{\mathbf{M}}(k+1)}{g_{\mathbf{M}}(k)} \leq \alpha$ *for all* $k \in \mathbb{N}_{>0}$*. Then, any* $b \in \mathcal{N}_{\mathbf{M}}$ *is such that*

$$b(m) \leq \alpha \cdot \frac{g_{\mathbf{M}}(\log_2 \log_2 m)}{\log_2 m}$$

for all $m \in \mathbb{N}, m \geq 4$.

Proof. Let $b \in \mathcal{N}_{\mathbf{M}}$. Let $m \in \mathbb{N}, m \geq 4$ and $k \in \mathbb{N}_{>0}$ be such that $2^{2^k} \leq m \leq 2^{2^{k+1}}$. Hence $2^k \leq \log_2 m \leq 2^{k+1}$ and of course $k \leq \log_2 \log_2 m \leq k+1$. Consider now $\mathrm{ISA}_{k+1,k+1}$. By Lemma 2.5 we have a partition $V_1, \ldots, V_{2^{k+1}}, U$ of the set of indices $[(k+1)+2^{k+1}(k+1)+2^{k+1}]$ of the input variables of $\mathrm{ISA}_{k+1,k+1}$ such that $r_{V_i}(\mathrm{ISA}_{k+1,k+1}) = 2^{2^{k+1}}$ for all $i \in [2^{k+1}]$. By

hypothesis, it therefore follows that:

$$\begin{aligned} g_{\mathbf{M}}(k+1) &\geq \mathbf{M}(\mathrm{ISA}_{k+1,k+1}) \\ &\geq \sum_{i=1}^{2^{k+1}} b(r_{V_i}(\mathrm{ISA}_{k+1,k+1})) + b(r_U(\mathrm{ISA}_{k+1,k+1})) \\ &\geq \sum_{i=1}^{2^{k+1}} b\big(2^{2^{k+1}}\big) \\ &= 2^{k+1} b\big(2^{2^{k+1}}\big) \ , \end{aligned}$$

therefore $b(2^{2^{k+1}}) \leq \frac{g_{\mathbf{M}}(k+1)}{2^{k+1}}$. But since $\frac{g_{\mathbf{M}}(k+1)}{g_{\mathbf{M}}(k)} \leq \alpha$, $g_{\mathbf{M}}(k) \leq g_{\mathbf{M}}(\log_2 \log_2 m)$ (because $g_{\mathbf{M}}$ is non-decreasing and $1 \leq k \leq \log_2 \log_2 m$), b is non-decreasing and $m \leq 2^{2^{k+1}}$ we have:

$$b(m) \leq b\big(2^{2^{k+1}}\big) \leq \frac{g_{\mathbf{M}}(k+1)}{2^{k+1}} \leq \alpha \cdot \frac{g_{\mathbf{M}}(\log_2 \log_2 m)}{\log_2 m} \ .$$

The lemma follows. □

Assuming an upper bound on every $b \in \mathcal{N}_{\mathbf{M}}$, as given for instance by the previous lemma, we can derive an upper bound on $\mathrm{N}_F^{\mathbf{M}}$ independently of the family of Boolean functions F. That is to say that we can give an overall (asymptotic) upper bound on the best complexity lower bounds we may obtain using Nečiporuk's lower bound method for the complexity measure $\mathbf{M}$, exhibiting the limitation of the method.

Lemma 4.2. *Let $\mathbf{M}$ be a given complexity measure on Boolean functions and assume that we have a function $h_{\mathbf{M}} \colon [4, +\infty[\to \mathbb{R}_{\geq 0}$ such that there exist $x_0 \in \big[2^8, +\infty\big[$ and a constant $\alpha \in \mathbb{R}_{>0}$ verifying that:*

(i) $h_{\mathbf{M}}$ is non-decreasing on $[x_0, +\infty[$;

(ii) $h_{\mathbf{M}}(2^x) \geq \log_2 x$ for all $x \in [\log_2 x_0, +\infty[$;

(iii) $h_{\mathbf{M}}(2^{2^v}) + h_{\mathbf{M}}(2^{2^{v'}}) \leq h_{\mathbf{M}}(2^{2^{v+v'}})$ for all $v, v' \in \mathbb{N}$ verifying $2^{2^v} \geq x_0$ and $2^{2^{v'}} \geq x_0$;

(iv) for all $b \in \mathcal{N}_{\mathbf{M}}$ and $m \in \mathbb{N}, m \geq 4$, we have $b(m) \leq \alpha \cdot h_{\mathbf{M}}(m)$.

Then, for any family of Boolean functions $F = \{f_n\}_{n \in \mathbb{N}}$, we have

$$\mathrm{N}_F^{\mathbf{M}}(n) \leq \alpha \cdot \Big(4 + h_{\mathbf{M}}(\lfloor x_0 \rfloor)\Big) \cdot \frac{n}{\log_2 n} \cdot h_{\mathbf{M}}(2^n)$$

for all $n \in \mathbb{N}, n \geq \log_2 x_0$.

Proof. The condition $n \geq \log_2 x_0$ ensures that $h_{\mathbf{M}}(2^n)$ is always well defined and satisfies (ii). Let $F = \{f_n\}_{n \in \mathbb{N}}$ be a family of Boolean functions. For all $n \in \mathbb{N}_{>0}$, let $h'_n \colon [n] \to \mathbb{R}$ be the function defined on $[n]$ by

$$h'_n(v) = \begin{cases} \alpha \cdot \min\{h_{\mathbf{M}}(2^{2^v}), h_{\mathbf{M}}(2^{n-v})\} & \text{if } \min\{2^{2^v}, 2^{n-v}\} \geq x_0 \\ \alpha \cdot h_{\mathbf{M}}(\lfloor x_0 \rfloor) & \text{otherwise.} \end{cases}$$

Claim 4.3. If $n \in \mathbb{N}_{>0}$ and $v \in [n]$ are such that $v \leq \log_2 n - 1$ and $2^{2^v} \geq x_0$ then $h'_n(v) = \alpha \cdot h_{\mathbf{M}}(2^{2^v})$.

Proof. With the hypothesis of the claim we have:

$$2^{2^v} \leq 2^{2^{\log_2 n-1}} = 2^{\frac{n}{2}} \leq 2^{n-\log_2 n+1} \leq 2^{n-v} \ ,$$

the middle inequality being a consequence of $n \geq \log_2 x_0 \geq 8$. Hence in this case $\min\{2^{2^v}, 2^{n-v}\}$ $= 2^{2^v}$ which is greater than x_0. As by (i) $h_{\mathbf{M}}$ is non-decreasing we have $h'_n(v) = \alpha \cdot h_{\mathbf{M}}(2^{2^v})$. □

Let $n \in \mathbb{N}_{>0}$, $b \in \mathcal{N}_{\mathbf{M}}$ and $V \subseteq [n], V \neq \emptyset$. According to (iv), we have $b(m) \leq \alpha \cdot h_{\mathbf{M}}(m)$ for all $m \in \mathbb{N}, m \geq 4$. Moreover, by Lemma 2.1, we have $r_V(f_n) \leq \min\{2^{2^{|V|}}, 2^{n-|V|}\}$, so since b is non-decreasing, it follows that $b(r_V(f_n)) \leq b(\min\{2^{2^{|V|}}, 2^{n-|V|}\})$. Now, if $\min\{2^{2^{|V|}}, 2^{n-|V|}\} \geq x_0$, we get that

$$\begin{aligned} b(\min\{2^{2^{|V|}}, 2^{n-|V|}\}) &\leq \alpha \cdot h_{\mathbf{M}}(\min\{2^{2^{|V|}}, 2^{n-|V|}\}) && \text{by (iv)} \\ &= \alpha \cdot \min\{h_{\mathbf{M}}(2^{2^{|V|}}), h_{\mathbf{M}}(2^{n-|V|})\} && \text{by (i)} \\ &= h'_n(|V|) \ ; \end{aligned}$$

otherwise (i.e. $\min\{2^{2^{|V|}}, 2^{n-|V|}\} < x_0$), we get that

$$b(\min\{2^{2^{|V|}}, 2^{n-|V|}\}) \leq b(\lfloor x_0 \rfloor) \leq \alpha \cdot h_{\mathbf{M}}(\lfloor x_0 \rfloor) = h'_n(|V|)$$

since b is non-decreasing and $\lfloor x_0 \rfloor \geq 4$. Hence, $b(r_V(f_n)) \leq h'_n(|V|)$ for all $n \in \mathbb{N}_{>0}$, $b \in \mathcal{N}_{\mathbf{M}}$ and $V \subseteq [n], V \neq \emptyset$. Therefore, by definition, it follows that for all $n \in \mathbb{N}_{>0}$, we have

$$\begin{aligned} \mathrm{N}_F^{\mathbf{M}}(n) &= \max\Big\{\sum_{i=1}^{p} b(r_{V_i}(f_n)) \ \Big| \ b \in \mathcal{N}_{\mathbf{M}} \text{ and } V_1, \ldots, V_p \text{ partition of } [n]\Big\} \\ &\leq \max\Big\{\sum_{i=1}^{p} h'_n(|V_i|) \ \Big| \ V_1, \ldots, V_p \text{ partition of } [n]\Big\} \\ &= \max\Big\{\sum_{i=1}^{p} h'_n(v_i) \ \Big| \ \sum_{i=1}^{p} v_i = n \text{ and } \forall i \in [p], v_i > 0\Big\} \ . \qquad (\star) \end{aligned}$$

Let $n \in \mathbb{N}, n \geq \log_2 x_0$ and $v_1, \ldots, v_p \in \mathbb{N}_{>0}$ such that $\sum_{i=1}^{p} v_i = n$ that realizes the maximum $(\star)$. We first show that without loss of generality we can assume that there exists at most one $j \in [p]$ such that $\min\{2^{2^{v_j}}, 2^{n-v_j}\} \geq x_0$ and $v_j \leq \frac{\log_2 n-1}{2}$.

If this is not the case then we have $v, v' \in [n]$ such that $\min\{2^{2^v}, 2^{n-v}, 2^{2^{v'}}, 2^{n-v'}\} \geq x_0$ and $v + v' \leq \log_2 n - 1$. It follows from (iii) and Claim 4.3 that

$$h'_n(v) + h'_n(v') = \alpha \cdot (h_{\mathbf{M}}(2^{2^v}) + h_{\mathbf{M}}(2^{2^{v'}})) \leq \alpha \cdot h_{\mathbf{M}}(2^{2^{v+v'}}).$$

But as $v \leq v + v' \leq \log_2 n - 1$, we have

$$\min\{2^{2^{v+v'}}, 2^{n-(v+v')}\} = 2^{2^{v+v'}} \geq 2^{2^v} = \min\{2^{2^v}, 2^{n-v}\} \geq x_0 \ ;$$

hence by Claim 4.3, $\alpha \cdot h_{\mathbf{M}}(2^{2^{v+v'}}) = h'_n(v+v')$ and $h'_n(v) + h'_n(v') \leq h'_n(v+v')$ and the partition that unifies the corresponding sets would yield a bound at least as big in $(\star)$.

If it exists this j is such that

$$h'_n(v_j) = \alpha \cdot h_{\mathbf{M}}(2^{2^{v_j}}) \leq \alpha \cdot h_{\mathbf{M}}\Big(2^{2^{\frac{\log_2 n-1}{2}}}\Big) \leq \alpha \cdot h_{\mathbf{M}}(2^n).$$

Consider now the remaining elements of the partition, i.e. those $i \in [p] \setminus \{j\}$. If moreover we have $\min\{2^{2^{v_i}}, 2^{n-v_i}\} \geq x_0$ then by definition of h'_n we have

$$h'_n(v_i) \leq \alpha \cdot h_{\mathbf{M}}(2^{n-v_i}) \leq \alpha \cdot h_{\mathbf{M}}(2^n).$$

As for this case we have $v_i > \frac{\log_2 n - 1}{2}$ there are at most $\frac{2n}{\log_2 n - 1}$ such i. Notice that $\frac{n}{\log_2 n - 1} \leq \frac{3}{2} \cdot \frac{n}{\log_2 n}$ for $n \geq 8$.

If otherwise $\min\{2^{2^{v_i}}, 2^{n-v_i}\} < x_0$, then we have

$$h'_n(v_i) = \alpha \cdot h_{\mathbf{M}}(\lfloor x_0 \rfloor)$$

and there are at most n such i.

Putting all together, we get that

$$\begin{aligned}
\mathrm{N}_F^{\mathbf{M}}(n) &\leq \max\Big\{ \sum_{i=1}^{p} h'_n(v_i) \;\Big|\; \sum_{i=1}^{p} v_i = n \text{ and } \forall i \in [p], v_i > 0 \Big\} \\
&\leq \alpha \cdot h_{\mathbf{M}}(2^n) + \frac{3n}{\log_2 n} \cdot \alpha \cdot h_{\mathbf{M}}(2^n) + n \cdot \alpha \cdot h_{\mathbf{M}}(\lfloor x_0 \rfloor) \\
&\leq \alpha \cdot h_{\mathbf{M}}(2^n) + \frac{3n}{\log_2 n} \cdot \alpha \cdot h_{\mathbf{M}}(2^n) + \frac{h_{\mathbf{M}}(2^n)}{\log_2 n} \cdot n \cdot \alpha \cdot h_{\mathbf{M}}(\lfloor x_0 \rfloor) \qquad \text{by (ii)} \\
&\leq \alpha \cdot \Big(4 + h_{\mathbf{M}}(\lfloor x_0 \rfloor)\Big) \cdot \frac{n}{\log_2 n} \cdot h_{\mathbf{M}}(2^n).
\end{aligned}$$

□

5 Upper Bounds for the Computation of $\mathrm{ISA}_{k,\ell}$

The $\mathrm{ISA}_{k,\ell}$ functions play a critical role in our approach to studying Nečiporuk's method. This section collects size upper bounds for computing $\mathrm{ISA}_{k,\ell}$ on every model considered in this paper. These bounds will be required when limits to the Nečiporuk method for these models are investigated.

Theorem 5.1. *Let $\delta \in \mathbb{N}$. For all $k, \ell \in \mathbb{N}_{>0}$,*

$$\mathbf{NBP}(\mathrm{ISA}_{k,\ell}), \oplus\mathbf{BP}(\mathrm{ISA}_{k,\ell}) \leq 3 \cdot 2^{k+\frac{\ell}{2}} + 2^{\ell} \tag{3}$$

$$\mathbf{LNBP}_{\delta}(\mathrm{ISA}_{k,\ell}) \leq \begin{cases} 12 \cdot 2^k \max\{\frac{2^{\ell-\delta}}{\ell-\delta}, \ell\} + \frac{2^{2\ell-\delta}}{\ell-\delta} & \text{if } \ell > \delta \\ 2^k(3\ell+1) + 2 \cdot 2^{\ell} & \text{if } \ell \leq \delta \end{cases} \tag{4}$$

$$\mathbf{BP}(\mathrm{ISA}_{k,\ell}) \leq 9 \cdot \frac{2^{k+\ell}}{\ell} + \frac{2^{2\ell}}{\ell} \tag{5}$$

$$\mathbf{LL}_{\delta}(\mathrm{ISA}_{k,\ell}) \leq 12 \cdot 2^k \cdot \max\{2^{\ell-\delta}, \ell\} + 3 \cdot 2^{\ell} \tag{6}$$

$$\mathbf{L}(\mathrm{ISA}_{k,\ell}) \leq 7 \cdot 2^k \cdot 2^{\ell}. \tag{7}$$

Proof. Recall the notation used to refer to the bits of an $\mathrm{ISA}_{k,\ell}$ instance $\vec{a}$. Here we further use $a_1, \ldots, a_k$ for the bits of the primary pointer p and (when relevant) $x_{n+1}, \ldots, x_{n+\delta}$ for the nondeterministic variables.

We begin with simple constructions:

Lemma 5.2. *Let $v_1, \ldots, v_k, y_1, \ldots, y_k, z_1, \ldots, z_{2^k}$ be Boolean variables, $k \geq 1$.*

1. *A size $2^k - 1$ deterministic branching program can "read" $v_1, \ldots, v_k$ and route the 2^k possible outcomes to 2^k distinct arcs;*

2. *A size $3k$ deterministic branching program can test whether $v_i = y_i$ holds for every $i \in [k]$;*

3. *A size $2^{k+1} - 2$ deterministic branching program with 2^k distinguished states s_w for $w \in \{0,1\}^k$ can ascertain that $(v_1, \ldots, v_k) = w$, i.e., has the property that for each w, a computation started at s_w accepts iff $(v_1, \ldots, v_k) = w$;*

4. *A size $4k$ formula can test whether $v_i = y_i$ holds for every $i \in [k]$;*

5. *A formula with leaves $z_1, \ldots, z_{2^k}$ and, for every $i \in [k]$, with 2^i leaves v_i or $\neg v_i$ can compute $z_{\mathrm{bin}_k(v_1,\ldots,v_k)+1}$.*

Proof. For (1), a full binary tree suffices. For (2), a size-3 program can test whether $v_i = y_i$ for a fixed i, so a cascade of k such programs can check equality for every i. For (3), an inverted binary tree first queries v_1 at each of 2^k leaves s_w, $w \in \{0,1\}^k$; each answer $a \in \{0,1\}$ branches from s_w to the unique state $s_{w'}$, among 2^{k-1} states at the next level, for which $w = aw'$; each state at this next level queries v_2 and branches to one of 2^{k-2} states at the next level, and so on, down to level k with two states querying v_k, for a total of $\Sigma_{1 \le i \le k} 2^i$ states; every missing arc in the above description rejects.

For (4), the formula $\wedge_{1 \le i \le k}[(v_i \wedge y_i) \vee (\neg v_i \wedge \neg y_i)]$ expanded into a binary tree has $4k$ leaves. For (5), we note that $(\neg v_1 \wedge z_1) \vee (v_1 \wedge z_2)$ computes $z_{\mathrm{bin}_1(v_1)+1}$ and use induction, having computed $z_{\mathrm{bin}_k(0,v_2,\ldots,v_k)+1}$ from the leaves $z_1, \ldots, z_{2^{k-1}}$ and the 2^i leaves v_{i+1} or $\neg v_{i+1}$ for $i \in [k-1]$, and having computed $z_{\mathrm{bin}_k(1,v_2,\ldots,v_k)+1}$ similarly from the leaves $z_{2^{k-1}+1}, \ldots, z_{2^k}$ and 2^i further leaves v_{i+1} or $\neg v_{i+1}$ for $i \in [k-1]$. □

The **NBP** *case.* If $\ell = 1$ then, by Lemma 5.2.1, a (deterministic) BP of size $2^k - 1 + 2^k + 2 < 3 \cdot 2^k + 2^\ell$ computes $\mathrm{ISA}_{k,\ell}$. So let $\ell > 1$. For every $w \in \{0,1\}^{\lceil \ell/2 \rceil}$, $w' \in \{0,1\}^{\lfloor \ell/2 \rfloor}$ and $p' \in [2^k]$, the NBP will have states $s_{(w,w')}$, $(p', s_{w'})$ and p'. Together with further states, the NBP implements the following:

- Read bits $a_1, \ldots, a_k, \mathrm{sec}_1, \ldots, \mathrm{sec}_{\lceil \ell/2 \rceil}$.

- Guess $w' \in \{0,1\}^{\lfloor \ell/2 \rfloor}$ and branch to $s_{(\mathrm{sec}_1,\ldots,\mathrm{sec}_{\lceil \ell/2 \rceil},w')}$, forgetting $a_1, \ldots, a_k$.
  (For every $w' \in \{0,1\}^{\lfloor \ell/2 \rfloor}$ and for every $a \in \{0,1\}$, every state querying $\mathrm{sec}_{\lceil \ell/2 \rceil}$, i.e., every bottom node in the binary tree formed by the first stage, is connected to the state $s_{(\mathrm{sec}_1,\ldots,\mathrm{sec}_{\lceil \ell/2 \rceil - 1},a,w')}$ with an arc labelled a.)

- If $\mathit{Data}[\mathrm{sec}_1, \ldots, \mathrm{sec}_{\lceil \ell/2 \rceil}, w'] = 1$ then guess the bits $a'_1, \ldots, a'_k$ of the primary pointer $p \in [2^k]$ and branch to the state $(\mathrm{bin}_k(a'_1, \ldots, a'_k) + 1, s_{w'})$.
  (For every $w \in \{0,1\}^{\lceil \ell/2 \rceil}$ and $w' \in \{0,1\}^{\lfloor \ell/2 \rfloor}$, the state $s_{(w,w')}$ queries $\mathit{Data}[w, w']$ and connects via an arc labelled 1 to every state $(p', s_{w'})$, $p' \in [2^k]$.)

- Ascertain that w' was guessed correctly.
  (For each $p' \in [2^k]$ separately, apply Lemma 5.2.3 to the distinguished states $(p', s_{w'})$, $w' \in \{0,1\}^{\lfloor \ell/2 \rfloor}$, to ascertain that a computation from $(p', s_{w'})$ reaches the state p' iff $(\mathrm{sec}[p']_{\lceil \ell/2 \rceil + 1}, \ldots, \mathrm{sec}[p']_\ell) = w'$.)

- Ascertain that p was guessed correctly.
  (Apply Lemma 5.2.3, to the 2^k distinguished states p', to ascertain that $(a_1, \ldots, a_k) = (a'_1, \ldots, a'_k)$.)

The first stage uses $2^{k+\lceil \ell/2 \rceil} - 1$ states by Lemma 5.2.1. The second stage needs the 2^ℓ states $s_{(w,w')}$. The fourth stage uses 2^k times $2^{\lfloor \ell/2 \rfloor + 1} - 2$ states by Lemma 5.2.3 (and also includes the $2^{k+\lfloor \ell/2 \rfloor}$ states $(p', s_{w'})$). The last stage uses $2^{k+1} - 2$ states by Lemma 5.2.3 for a total $< 2^k(2^{\lceil \ell/2 \rceil} + 2 \cdot 2^{\lfloor \ell/2 \rfloor} - 2) + 2^{k+1} + 2^\ell$, which equals $2^k(3 \cdot 2^{\ell/2}) + 2^\ell$ when ℓ is even and $2^k(\frac{4}{\sqrt{2}} \cdot 2^{\ell/2}) + 2^\ell < 2^k(3 \cdot 2^{\ell/2}) + 2^\ell$ when ℓ is odd.

The $\oplus$**BP** *case.* It is easy to check that the above NBP has a unique accepting path for any input for which $\mathrm{ISA}_{k,\ell}$ is 1 and hence as a $\oplus$BP it also computes $\mathrm{ISA}_{k,\ell}$.

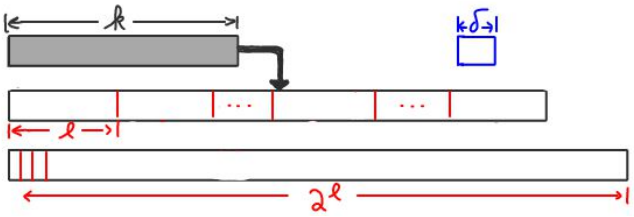
The **LNBP**$_\delta$ *case.* If $\ell \leq \delta$ then the secondary $\mathrm{ISA}_{k,\ell}$ pointer is no wider than δ, i.e., contains no more than δ bits. So a δ-LNBP can "store" the secondary pointer within its first ℓ nondeterministic variables $x_{n+1}, \ldots, x_{n+\ell}$ and solve $\mathrm{ISA}_{k,\ell}$ as follows:

- Read the primary pointer
- Check that $(\mathrm{sec}_1, \ldots, \mathrm{sec}_\ell) = (x_{n+1}, \ldots, x_{n+\ell})$
- Forget everything
- Read $x_{n+1}, \ldots, x_{n+\ell}$
- Check that data$[x_{n+1}, \ldots, x_{n+\ell}]$=1.

The first and second steps use $2^k - 1$ and $2^k 3\ell$ states respectively, appealing to Lemma 5.2.1 and Lemma 5.2.2. Note that across the second step, neither the secondary pointer nor $x_{n+1}, \ldots, x_{n+\ell}$ are remembered. The third step merges every arc that survived the second step and thus requires no state. The fourth and fifth steps require $2^\ell - 1$ and 2^ℓ states, for a total $< 2^k + 2^k 3\ell + 2 \cdot 2^\ell$.

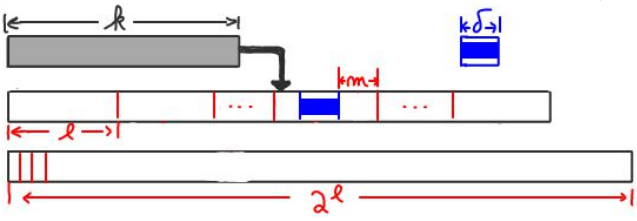
Now suppose that $\ell > \delta$, i.e., the secondary pointer is strictly wider than δ. Let $m \in [\ell - \delta - 1]$, to be set optimally later. A δ-LNBP can implement the following strategy, where grey-shaded regions in the diagrams indicate the portion of the $\mathrm{ISA}_{k,\ell}$ variables that are remembered, at exponential cost in numbers of states, at any given time.

1. Read the primary pointer:



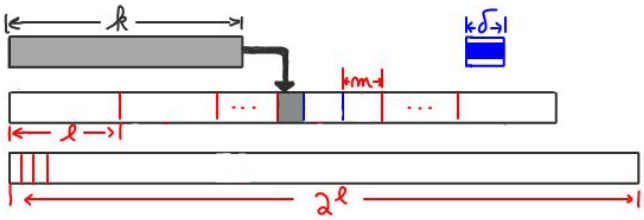
   Uses $2^k - 1$ states as per Lemma 5.2.1.

2. Check δ contiguous secondary pointer bits for equality with $x_{n+1}, \ldots, x_{n+\delta}$:



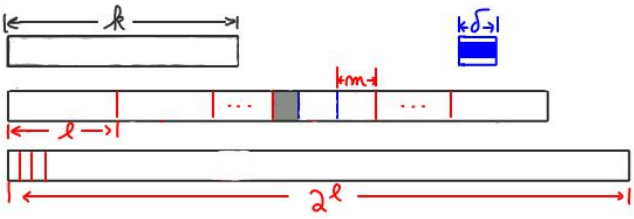
   Uses 2^k times 3δ states, again by Lemma 5.2.2. None of the checked bits are remembered.

3. Read $\ell - m - \delta$ other contiguous bits from the secondary pointer:



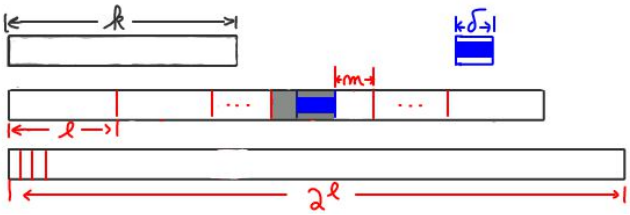
Uses 2^k times $(2^{\ell-m-\delta}-1)$ states.

4. Forget the primary pointer:



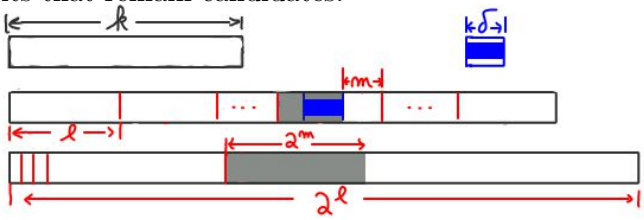
No state required.

5. Read and remember the nondeterministic bits:



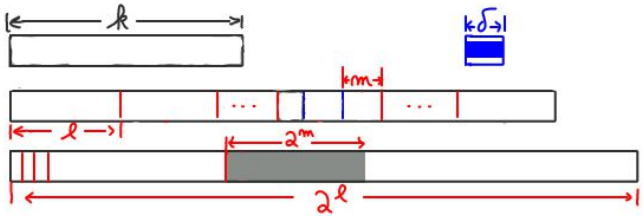
Uses $2^{\ell-m-\delta}(2^\delta-1) < 2^{\ell-m}$ states.

6. Read the data bits that remain candidates:



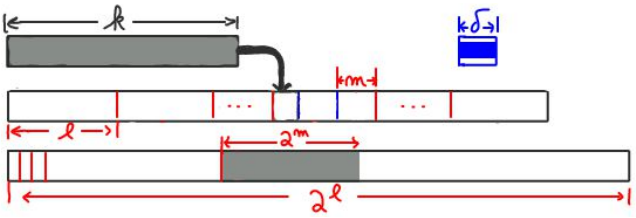
Uses $2^{\ell-m}(2^{2^m}-1) < 2^{\ell-m}2^{2^m}$ states.

7. Forget the part of the secondary pointer that was read:



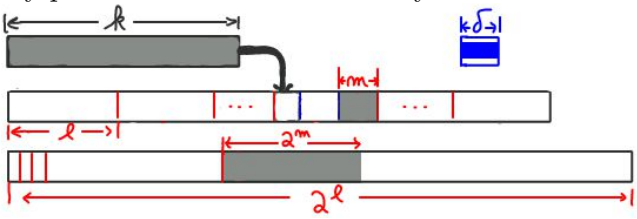
No state required.

8. Read the primary pointer:



Uses $2^{2^m}(2^k-1) < 2^k 2^{2^m}$ states.

9. Read the secondary pointer bits that were never yet accessed:



Uses $2^{2^m} 2^k (2^m - 1) < 2^k 2^m 2^{2^m}$ states.

10. Output the appropriate data bit from memory: no state required.

The resulting δ-LNBP has fewer than

$$2^k + 2^k 3\delta + 2^k 2^{\ell-\delta-m} + 2^{\ell-m} + 2^{\ell-m} 2^{2^m} + 2^k 2^{2^m} + 2^k 2^m 2^{2^m}$$

states, which is less than

$$12 \cdot 2^k \max\left\{ \frac{2^{\ell-\delta}}{\ell-\delta}, \ell \right\} + \frac{2^{2\ell-\delta}}{\ell-\delta}$$

when m is set to $\lfloor \log_2(\ell - \delta - \log_2((\ell-\delta)^2)) \rfloor$ and $\ell - \delta \geq 8$ (and the degenerate case in which $1 \leq \ell - \delta < 8$ is treated separately by using a simpler method to compute $\mathrm{ISA}_{k,\ell}$).

The **BP** *case.* Follows from the $\mathbf{LNBP}_\delta$ case by setting $\delta = 0$. More specifically, stages 2 and 5 in the construction of the δ-NLBP are skipped.

The δ-**LL** *case.* We will not exploit more than ℓ nondeterministic variables amongst $x_{n+1}, \ldots, x_{n+\delta}$ so we suppose that $\delta \leq \ell$. Let $m = \ell - \delta$. The nondeterministic formula $V \wedge D$ solves $\mathrm{ISA}_{k,\ell}$ provided that V and D fulfil

$$V = 1 \text{ iff } (\mathrm{sec}_{m+1}, \ldots, \mathrm{sec}_\ell) = (x_{n+1}, \ldots, x_{n+\delta}),$$
$$D = 1 \text{ iff } \mathit{Data}[F_1, \ldots, F_m, x_{n+1}, \ldots, x_{n+\delta}] = 1,$$

and for $1 \leq j \leq m$, F_j evaluates to sec_j. By Lemma 5.2.5, D exists such that

$$|D| = 2^\ell + \Sigma_{j=1}^m 2^j \cdot |F_j| + \Sigma_{j=m+1}^\ell 2^j < 3 \cdot 2^\ell + \Sigma_{j=1}^m 2^j \cdot |F_j|. \tag{8}$$

By Lemma 5.2.5, each formula F_j, $1 \leq j \leq m$, can be constructed of size

$$|F_j| = 2^k + \Sigma_{j=1}^k 2^j < 3 \cdot 2^k. \tag{9}$$

By Lemma 5.2.4, for every $p \in [2^k]$, a formula V_p of size 4δ can be constructed that evaluates to 1 iff $(\mathrm{sec}[p]_{m+1}, \ldots, \mathrm{sec}[p]_\ell) = (x_{n+1}, \ldots, x_{n+\delta})$. The formula V can then be constructed using Lemma 5.2.5, taking $z_1, \ldots, z_{2^k}$ as $V_1, \ldots, V_{2^k}$. The size of V is then

$$|V| = \Sigma_{j=1}^{2^k} |V_j| \;+\; \Sigma_{j=1}^k 2^j < 2^k \cdot 4\delta + 2^{k+1} \leq 2^k \cdot 6\ell. \tag{10}$$

Substituting (9) into (8) and using (10), the size of $V \wedge D$ is at most

$$2^k \cdot 6\ell + 3 \cdot 2^\ell + 2^{m+1} 3 \cdot 2^k \leq 6 \cdot 2^k (\ell + 2^m) + 3 \cdot 2^\ell \leq 12 \cdot 2^k \cdot \max\{2^m, \ell\} + 3 \cdot 2^\ell.$$

The **L** *case.* Follows from the δ-**LL** case by setting $\delta = 0$. More sharply, V from that construction is not needed, and $|D| = 2^\ell + \Sigma_{j=1}^\ell 2^j |F_j| < 2^\ell + 3 \cdot 2^k \cdot 2^{\ell+1} < 7 \cdot 2^k \cdot 2^\ell$. □

6 Nondeterministic and Parity Branching Programs revisited

We note in this section that, in the case of **NBP** and $\oplus$**BP**, the flexibility added by Definition 4.1 over Definition 3.1 yields no better lower bounds.

We first define the function $b_{\mathbf{NBP},\oplus\mathbf{BP}}\colon \mathbb{N}_{>0} \to \mathbb{N}$ by

$$b_{\mathbf{NBP},\oplus\mathbf{BP}}(m) = \begin{cases} \left\lceil \sqrt{\frac{1}{2}\log_2 m} - 1 \right\rceil & \text{if } m \geq 4 \\ 0 & \text{otherwise} \end{cases}$$

for all $m \in \mathbb{N}_{>0}$. Using the same strategy as in the proofs of Lemma 3.1 and Proposition 3.3, we can prove the following.

Proposition 6.1. *$b_{\mathbf{NBP},\oplus\mathbf{BP}}$ is a Nečiporuk bounding function for the* **NBP** *(respectively,* $\oplus$**BP***) size complexity measure; i.e., $b_{\mathbf{NBP},\oplus\mathbf{BP}} \in \mathcal{N}_{\mathbf{NBP}}, \mathcal{N}_{\oplus\mathbf{BP}}$.*

Combining this with Lemmas 2.4 and 2.6, we can immediately derive asymptotic lower bounds on $\mathbf{NBP}(\mathrm{ED}_n)$ and $\mathbf{NBP}(\mathrm{ISA}_n)$ using Nečiporuk's method and hence on $\mathrm{N}_{\mathrm{ED}}^{\mathbf{NBP}}$, $\mathrm{N}_{\mathrm{ED}}^{\oplus\mathbf{BP}}$, $\mathrm{N}_{\mathrm{ISA}}^{\mathbf{NBP}}$, and $\mathrm{N}_{\mathrm{ISA}}^{\oplus\mathbf{BP}}$.

Proposition 6.2. $\mathrm{N}_{\mathrm{ED}}^{\mathbf{NBP}}(n), \mathrm{N}_{\mathrm{ISA}}^{\mathbf{NBP}}(n), \mathrm{N}_{\mathrm{ED}}^{\oplus\mathbf{BP}}(n), \mathrm{N}_{\mathrm{ISA}}^{\oplus\mathbf{BP}}(n) \in \Omega\left(\frac{n^{3/2}}{\log_2 n}\right)$ *and hence* $\mathbf{NBP}(\mathrm{ED}_n)$, $\mathbf{NBP}(\mathrm{ISA}_n)$, $\oplus\mathbf{BP}(\mathrm{ED}_n)$, $\oplus\mathbf{BP}(\mathrm{ISA}_n)$ *are all* $\Omega\left(\frac{n^{3/2}}{\log_2 n}\right)$.

Then, we can show that $b_{\mathbf{NBP},\oplus\mathbf{BP}}$ is in fact the asymptotically largest function in $\mathcal{N}_{\mathbf{NBP}} \cup \mathcal{N}_{\oplus\mathbf{BP}}$ and that the previous lower bound is in fact also the asymptotically largest we may obtain. To do this, we appeal to our upper bound from Theorem 5.1 on the size of NBPs and $\oplus$BPs computing $\mathrm{ISA}_{k,\ell}$ and apply Lemma 4.1.

Proposition 6.3. *There exists a constant $c \in \mathbb{R}_{>0}$ verifying that any $b \in \mathcal{N}_{\mathbf{NBP}} \cup \mathcal{N}_{\oplus\mathbf{BP}}$ is such that $b(m) \leq c \cdot b_{\mathbf{NBP},\oplus\mathbf{BP}}(m)$, for $m \geq 4$.*

Proof. Let $g\colon [1, +\infty[\to \mathbb{R}_{\geq 0}$ be the non-decreasing function defined by $g(x) = 4 \cdot 2^{\frac{3}{2}x}$ for all $x \in [1, +\infty[$. Theorem 5.1 tells us that for all $k \in \mathbb{N}_{>0}$, we have

$$\mathbf{NBP}(\mathrm{ISA}_{k,k}), \oplus\mathbf{BP}(\mathrm{ISA}_{k,k}) \leq 3 \cdot 2^{\frac{3}{2}k} + 2^k \leq 4 \cdot 2^{\frac{3}{2}k} = g(k)$$

and moreover, $\frac{g(k+1)}{g(k)} = \frac{4 \cdot 2^{\frac{3}{2}(k+1)}}{4 \cdot 2^{\frac{3}{2}k}} = 2\sqrt{2}$ for all $k \in \mathbb{N}_{>0}$. Therefore, by Lemma 4.1, any $b \in \mathcal{N}_{\mathbf{NBP}} \cup \mathcal{N}_{\oplus\mathbf{BP}}$ verifies

$$b(m) \leq 2\sqrt{2} \cdot \frac{g(\log_2 \log_2 m)}{\log_2 m} = 2\sqrt{2} \cdot \frac{4 \cdot 2^{\frac{3}{2}\log_2 \log_2 m}}{\log_2 m} = 8\sqrt{2} \cdot \sqrt{\log_2 m}$$

for all $m \in \mathbb{N}, m \geq 4$. □

Finally, using this and Lemma 4.2, we get the following result, showing that the asymptotically greatest lower bound we may expect using Nečiporuk's method for **NBP** is (asymptotically) equivalent to the lower bound for ISA given in Proposition 6.2.

Theorem 6.4. *For any family of Boolean functions $F = \{f_n\}_{n\in\mathbb{N}}$, $\mathrm{N}_F^{\mathbf{NBP}}(n), \mathrm{N}_F^{\oplus\mathbf{BP}}(n) \in \mathrm{O}\left(\frac{n^{3/2}}{\log_2 n}\right)$.*

Proof. We aim at applying Lemma 4.2 which requires four hypotheses, (i) to (iv).

For (i), let $h\colon [4,+\infty[\to \mathbb{R}_{\geq 0}$ be the function defined by $h(x) = \sqrt{\log_2 x}$ for all $x \in [4,+\infty[$ and $x_0 = 2^8$; as required, h is non-decreasing on $\left[2^8,+\infty\right[$.

For (ii), notice that $h(2^x) = \sqrt{x} \geq \log_2 x$ for all $x \in \left[2^8,+\infty\right[$.

For (iii), for all $v, v' \in \mathbb{N}$ verifying $2^{2^v} \geq 2^8$ and $2^{2^{v'}} \geq 2^8$, we have $h(2^{2^v}) + h(2^{2^{v'}}) = \sqrt{2^v} + \sqrt{2^{v'}} \leq \sqrt{2^{v+v'}} = h(2^{2^{v+v'}})$ because $x + y \leq xy$ when $x, y \geq 2$.

For (iv), by Proposition 6.3, we know that any $b \in \mathcal{N}_{\mathbf{NBP}} \cup \mathcal{N}_{\oplus\mathbf{BP}}$ is such that $b(m) \leq \alpha \cdot \sqrt{\log_2 m} = \alpha \cdot h(m)$ for all $m \in \mathbb{N}, m \geq 4$.

We can therefore apply Lemma 4.2 with $x_0 = 2^8$ and get that for any family of Boolean functions $F = \{f_n\}_{n\in\mathbb{N}}$ and all $n \in \mathbb{N}, n \geq 8$,

$$\mathrm{N}_F^{\mathbf{NBP}}(n), \mathrm{N}_F^{\oplus\mathbf{BP}}(n) \leq \alpha \cdot \left(4 + h(\left\lfloor 2^8 \right\rfloor)\right) \cdot \frac{n}{\log_2 n} \cdot h(2^n) = c \cdot \frac{n}{\log_2 n} \cdot \sqrt{n} = c \cdot \frac{n^{3/2}}{\log_2 n} \ ,$$

which implies that $\mathrm{N}_F^{\mathbf{NBP}}(n), \mathrm{N}_F^{\oplus\mathbf{BP}}(n) \in \mathrm{O}\left(\frac{n^{3/2}}{\log_2 n}\right)$. □

7 Deterministic and Limited Nondeterministic Branching Programs

In this section, we focus on the model of Boolean deterministic branching programs, as well as its limited nondeterministic counterpart. In the case of **BP**, results related to the Nečiporuk method have been well-known for a long time (see for instance [20, Chapter 14, Section 3] or [1]). Reproving these results using what we presented in Section 4 is an opportunity to confirm the usability and validity of our approach.

Concerning limited nondeterministic branching programs, the definition of the model itself, as well as the results presented in this section concerning Nečiporuk's method for the associated measure seem to be novel.

For all $\delta \in \mathbb{N}$, let us define the functions $b_{\mathbf{LNBP}_\delta}\colon \mathbb{N}_{>0} \to \mathbb{N}$ and $b_{\mathbf{BP}}\colon \mathbb{N}_{>0} \to \mathbb{N}$ given by

$$b_{\mathbf{LNBP}_\delta}(m) = \begin{cases} \left\lceil \frac{1}{6} h_{\mathbf{LNBP}_\delta}(m) \right\rceil & \text{if } m \geq 4 \\ 0 & \text{otherwise} \end{cases}$$

and

$$b_{\mathbf{BP}}(m) = \begin{cases} \left\lceil \frac{1}{6} \frac{\log_2 m}{\log_2 \log_2 m} \right\rceil & \text{if } m \geq 4 \\ 0 & \text{otherwise} \end{cases}$$

for all $m \in \mathbb{N}_{>0}$, where $h_{\mathbf{LNBP}_\delta}\colon [4,+\infty[\to \mathbb{R}$ is defined as

$$h_{\mathbf{LNBP}_\delta}(x) = \begin{cases} \max\left\{ \frac{\log_2 x}{2^\delta (\log_2 \log_2 x - \delta)}, \log_2 \log_2 x \right\} & \text{if } 2^{2^{\delta+1}} \leq x \\ \log_2 \log_2 x & \text{otherwise .} \end{cases}$$

It is straightforward to see that $b_{\mathbf{BP}}(m) \leq b_{\mathbf{LNBP}_0}(m)$ for all $m \in \mathbb{N}_{>0}$ and that equality holds as soon as $b_{\mathbf{BP}}(m) \geq \log_2 \log_2 m$.

To prove that $b_{\mathbf{BP}} \in \mathcal{N}_{\mathbf{BP}}$, we use the well known idea that is usually used (see for instance [20], [1] or [8]) to derive a specific function $b \in \mathcal{N}_{\mathbf{BP}}$, which is the fact that, given a Boolean function f and a Boolean BP P that computes it, we can compute any subfunction $f|_\rho$ of f with a Boolean BP obtained from P by "fixing" the values of the variables to which a value is affected by ρ (removing the associated vertices and directly linking their predecessors to their

successors through the arcs labelled accordingly). Therefore, if we denote by s the number of vertices in P labelled by elements from V, we get that an upper bound on the maximum number of subfunctions computed by BPs with s vertices obtained by "fixing" the values of a given set of variables in a given BP implies a lower bound on s depending on $r_V(f)$, as this number must be at least as big as $r_V(f)$. For the case of limited nondeterminism, it suffices to observe that a Boolean δ-LNBP (for $\delta \in \mathbb{N}$) computing some Boolean function f, does in fact deterministically compute a proof-checker function g for f. We can then combine the aforementioned technique with Lemma 2.7 binding the number of subfunctions of f on V and the number of subfunctions of g on V.

Proposition 7.1. *$b_{\mathbf{LNBP}_\delta} \in \mathcal{N}_{\mathbf{LNBP}_\delta}$ for all $\delta \in \mathbb{N}$. In particular, $b_{\mathbf{BP}} \in \mathcal{N}_{\mathbf{BP}}$.*

Proof. Let $\delta \in \mathbb{N}$. It is not too difficult to show that $b_{\mathbf{LNBP}_\delta}$ and $b_{\mathbf{BP}}$ are non-decreasing, we leave this to the reader.

Let f be a n-ary Boolean function on V and $V_1, \ldots, V_p$ a partition of V. Let P be a Boolean δ-LNBP computing f and let g be the $(n+\delta)$-ary Boolean function computed by P when considering the δ nondeterministic bits as regular input variables (that is, g is such that, for all $\vec{a} \in \{0,1\}^{V \cup [\delta]}$, $g(\vec{a}) = 1$ if, and only if, $P[\vec{a}]$ contains a path from s to t_1). g is a proof-checker function for f.

For all $i \in [p]$ we will denote by $s_i \in \mathbb{N}$ the number of vertices in P labelled by elements in V_i, as well as $q \in \mathbb{N}$ the number of vertices labelled by elements in $U = [\delta]$. It is clear that P is of size $\sum_{i=1}^{p} s_i + q \geq \sum_{i=1}^{p} s_i$.

We now claim that $s_i \geq b_{\mathbf{LNBP}_\delta}(r_{V_i}(f))$ for all $i \in [p]$.

Let $i \in [p]$. Let V_i' be the subset of V_i containing all indices of variables on which f depends. Then, by Lemma 2.2, $r_{V_i}(f) = r_{V_i'}(f)$. Moreover for each element $l \in V_i'$, P contains at least one vertex labelled by l. By Lemma 2.1, it follows that $r_{V_i}(f) = r_{V_i'}(f) \leq 2^{2^{|V_i'|}} \leq 2^{2^{s_i}}$ and $s_i \geq \log_2 \log_2(r_{V_i}(f))$.

If $r_{V_i}(f) \leq 3$ the claim is obvious from the definition of $b_{\mathbf{LNBP}_\delta}$.

In the case where $4 \leq r_{V_i}(f) < 4^{2^\delta}$ we have $\left\lceil \frac{1}{6} \log_2 \log_2(r_{V_i}(f)) \right\rceil = b_{\mathbf{LNBP}_\delta}(r_{V_i}(f))$ and we are also done as s_i is an integer and $s_i \geq \log_2 \log_2(r_{V_i}(f))$.

We now assume $r_{V_i}(f) \geq 4^{2^\delta}$. In particular this implies that $s_i \geq 1$.

Observe that for all $h \colon \{0,1\}^{V_i} \to \{0,1\}$ a subfunction of g on V_i, by definition, there exists a partial assignment $\rho \in \{0,1\}^{V \setminus V_i \cup [\delta]}$ such that $g|_\rho = h$, so it is not too difficult to see that h is computed by the Boolean BP of size s_i obtained from P by:

1. removing all non sink vertices labelled by variables not in V_i;
2. defining the new start vertex as the only vertex whose label is in V_i and connected to the start vertex of P by a path of nodes labelled by a variable outside of V_i and arcs labelled consistently with ρ;
3. connecting a vertex u to a vertex v by an arc labelled by $a \in \{0,1\}$ if, and only if, there exists a path from u to v in P verifying that any intermediate vertex of the path is labelled by a variable outside of V_i, the first arc is labelled by a and each arc (but the first one) is labelled consistently with ρ.

Thus, $r_{V_i}(g)$ is necessarily upper-bounded by the number of syntactically distinct such BPs we can build from P that way. Since, for such a BP, there are at most $s_i + 2$ possible choices for the start vertex and by functionality of the set of arcs labelled 0 and the set of arcs labelled 1 seen as successor relations, there are at most $(s_i+1)^{s_i}$ possible choices for the set of arcs labelled

0, as well as at most $(s_i+1)^{s_i}$ possible choices for the set of arcs labelled 1, $r_{V_i}(g)$ is at most $(s_i+2)(s_i+1)^{2s_i}$. Assuming $2 \leq s_i$ we get:

$$\begin{aligned} r_{V_i}(g) \leq & (s_i+2)(s_i+1)^{2s_i} && = 2^{\log_2(s_i+2)+2s_i\log_2(s_i+1)} \\ & && \leq 2^{3s_i\log_2(s_i+2)} \\ & && \leq 2^{6s_i\log_2(s_i)} && \text{as } 2 \leq s_i \ . \end{aligned}$$

It follows that $s_i \geq \frac{1}{6} \cdot \frac{\log_2(r_{V_i}(g))}{\log_2\log_2(r_{V_i}(g))}$. If $s_i = 1$ it is clear as $r_{V_i}(g)$ is then at most 4, and if $s_i \geq 2$ we would otherwise have

$$\begin{aligned} s_i\log_2(s_i) &< \frac{1}{6} \cdot \frac{\log_2(r_{V_i}(g))}{\log_2\log_2(r_{V_i}(g))} \log_2\Big(\frac{1}{6} \cdot \frac{\log_2(r_{V_i}(g))}{\log_2\log_2(r_{V_i}(g))}\Big) \\ &= \frac{\log_2(r_{V_i}(g))}{6} - \frac{\log_2(r_{V_i}(g))\log_2\big(6\log_2\log_2(r_{V_i}(g))\big)}{6\log_2\log_2(r_{V_i}(g))} \\ &< \frac{\log_2(r_{V_i}(g))}{6} \end{aligned}$$

(observe that the last inequality follows from the fact that the subtracted member must necessarily be positive since $r_{V_i}(g) \geq 4$). From Lemma 2.7 we have $r_{V_i}(g) \geq r_{V_i}(f)^{\frac{1}{2^\delta}}$. The function $\frac{\log_2(x)}{\log_2\log_2(x)}$ being non-decreasing on $\big[e^{e\ln(2)}, +\infty\big[$ and as $\frac{\log_2(x)}{\log_2\log_2(x)} \leq 2$ for $x \in \big[4, e^{e\ln(2)}\big]$, we get for $r_{V_i}(f)^{\frac{1}{2^\delta}} \geq 4$:

$$s_i \geq \frac{1}{6} \cdot \frac{\log_2(r_{V_i}(f))}{2^\delta\big(\log_2\log_2(r_{V_i}(f)) - \delta\big)} \ .$$

In conclusion, for all $\delta, n \in \mathbb{N}$, and any n-ary Boolean function f on V and any partition $V_1, \ldots, V_p$ of V, it holds that $\mathbf{LNBP}_\delta(f) \geq \sum_{i=1}^p b_{\mathbf{LNBP}_\delta}(r_{V_i}(f))$, hence $b_{\mathbf{LNBP}_\delta} \in \mathcal{N}_{\mathbf{LNBP}_\delta}$. It also directly follows that $b_{\mathbf{BP}} \in \mathcal{N}_{\mathbf{BP}}$ because $b_{\mathbf{BP}}$ is non-decreasing and $b_{\mathbf{BP}}(m) \leq b_{\mathbf{LNBP}_0}(m)$ for all $m \in \mathbb{N}_{>0}$. □

Let us define $\Gamma_{\mathbf{LNBP}} \colon [2, +\infty) \times \mathbb{N}_{>0} \to \mathbb{R}$ by

$$\Gamma(x, \delta) = \begin{cases} \max\big\{\frac{x^2}{2^\delta(\log_2 x - \delta)\log_2 x}, x\big\} & \text{if } 2^{\delta+1} \leq x \\ x & \text{otherwise .} \end{cases}$$

Using the previous proposition and Lemma 2.6, we can immediately derive the following asymptotic lower bound on $\mathrm{N}_{\mathrm{ISA}}^{\mathbf{LNBP}_{\Delta(n)}}$ for any $\Delta \colon \mathbb{N} \to \mathbb{N}$.

Proposition 7.2. $\mathrm{N}_{\mathrm{ISA}}^{\mathbf{LNBP}_{\Delta(n)}}(n) \in \Omega\big(\Gamma_{\mathbf{LNBP}}(n, \Delta(n))\big)$ *for any* $\Delta \colon \mathbb{N} \to \mathbb{N}$. *In particular,* $\mathrm{N}_{\mathrm{ISA}}^{\mathbf{BP}}(n) \in \Omega\big(\frac{n^2}{\log_2^2 n}\big)$.

Proof. Let $\Delta \colon \mathbb{N} \to \mathbb{N}$. Let $n \in \mathbb{N}, n \geq 32$. Let $V_1, \ldots, V_p, U$ be a partition of $[n]$ such that $r_{V_i}(\mathrm{ISA}_n) = 2^q$ for all $i \in [p]$ where $p, q \in \mathbb{N}_{>0}$ verify $p \geq \frac{1}{32} \cdot \frac{n}{\log_2 n}$ and $q \geq \frac{n}{16}$ as given by

Lemma 2.6. We have

$$
\begin{aligned}
&\mathrm{N}_{\mathrm{ISA}}^{\mathbf{LNBP}_{\Delta(n)}}(n)\\
\geq&\sum_{i=1}^{p} b_{\mathbf{LNBP}_{\Delta(n)}}(r_{V_i}(\mathrm{ISA}_n)) + b_{\mathbf{LNBP}_{\Delta(n)}}(r_U(\mathrm{ISA}_n))\\
\geq&\sum_{i=1}^{p} b_{\mathbf{LNBP}_{\Delta(n)}}(2^q)\\
\geq&\frac{1}{6}\cdot\frac{1}{32}\cdot\frac{n}{\log_2 n}\cdot\begin{cases}\max\{\frac{\frac{n}{16}}{2^{\Delta(n)}(\log_2(\frac{n}{16})-\Delta(n))},\log_2(\frac{n}{16})\} & \text{if } 4^{2^{\Delta(n)}}\leq 2^{\frac{n}{16}}\\ \log_2(\frac{n}{16}) & \text{otherwise}\end{cases}\\
\geq&\frac{1}{192}\cdot\frac{n}{\log_2 n}\cdot\frac{1}{16}\cdot\begin{cases}\max\{\frac{n}{2^{\Delta(n)}(\log_2(n)-\Delta(n))},\log_2 n\} & \text{if } 2^{\Delta(n)+5}\leq n\\ \log_2 n & \text{otherwise}\end{cases} && \text{as } \log_2\Big(\frac{n}{16}\Big)\geq\frac{\log_2 n}{16}\\
\geq&\frac{c}{3072}\cdot\frac{n}{\log_2 n}\cdot\begin{cases}\max\{\frac{n}{2^{\Delta(n)}(\log_2(n)-\Delta(n))},\log_2 n\} & \text{if } 2^{\Delta(n)+1}\leq n\\ \log_2 n & \text{otherwise}\end{cases} && \text{for some } c\text{, see below}\\
=&\frac{c}{3072}\cdot\begin{cases}\max\{\frac{n^2}{2^{\Delta(n)}(\log_2(n)-\Delta(n))\log_2 n},n\} & \text{if } 2^{\Delta(n)+1}\leq n\\ n & \text{otherwise}\end{cases}\\
=&\frac{c}{3072}\cdot\Gamma_{\mathbf{LNBP}}(n,\Delta(n)) && \text{as desired .}
\end{aligned}
$$

In order to show the inequality above it suffices to show that $\log_2 x \geq \frac{cx}{2^\alpha(\log_2(x)-\alpha)}$ for $x\in I=\big[2^{\alpha+1},2^{\alpha+5}\big]$, $\alpha\geq 0$ and some constant c.

It suffices to show that the function $f(x)=2^\alpha\log_2(x)\log_2(\frac{x}{2^\alpha})-cx$ is non-decreasing on I. This concludes the claim as $f(2^{\alpha+1})=2^\alpha(\alpha+1)-c2^{\alpha+1}\geq 0$ when $\alpha\geq 0$ and $c\leq\frac{1}{2}$. To see this notice that the derivative of f is $2^\alpha\big(\frac{\log_2 x}{x\ln 2}+\frac{\log_2(\frac{x}{2^\alpha})}{x\ln 2}\big)-c$ that has the same sign as $g(x)=2^\alpha\log_2\big(\frac{x^2}{2^\alpha}\big)-xc\ln 2$ for $x\in I$.

The derivative of g is $\frac{2^{\alpha+1}}{x\ln 2}-c\ln 2$ that vanishes for a value $x_0=\frac{2^{\alpha+1}}{c(\ln 2)^2}$. Assuming $c\leq\frac{1}{2^4(\ln 2)^2}$ we have $x_0\geq 2^{\alpha+5}$ and the derivative of g is always non-negative on I.

We have $g(2^{\alpha+1})=2^\alpha(\alpha+2-2c\ln 2)$ which is non-negative as soon as $c\leq\frac{1}{\ln 2}$. Hence g is non-negative on I.

Hence taking $c=\frac{1}{2^4(\ln 2)^2}$ yields the desired result. □

Now we show that for all $\delta\in\mathbb{N}$, $b_{\mathbf{LNBP}_\delta}$ is in fact an asymptotically largest function in $\mathcal{N}_{\mathbf{LNBP}_\delta}$ (as well as for $b_{\mathbf{BP}}$ and $\mathcal{N}_{\mathbf{BP}}$) and that the previous bound is in fact also the asymptotically largest we may obtain, using the meta-results of Section 4. To do this, we appeal to our upper bound from Theorem 5.1 on the size of a δ-LNBP (or a deterministic BP) computing $\mathrm{ISA}_{k,\ell}$ and apply Lemma 4.1.

Proposition 7.3. *There exists a constant $c\in\mathbb{R}_{>0}$ verifying that for each $\delta\in\mathbb{N}$, any $b\in\mathcal{N}_{\mathbf{LNBP}_\delta}$ is such that $b(m)\leq c\cdot b_{\mathbf{LNBP}_\delta}(m)$ for all $m\in\mathbb{N}, m\geq 4$. In particular, there exists a constant $c'\in\mathbb{R}_{>0}$ verifying that any $b\in\mathcal{N}_{\mathbf{BP}}$ is such that $b(m)\leq c'\cdot b_{\mathbf{BP}}(m)$ for all $m\in\mathbb{N}, m\geq 4$.*

Proof. Let $\delta\in\mathbb{N}$. Let $g\colon[1,+\infty[\to\mathbb{R}_{\geq 0}$ be the non-decreasing function defined by

$$g(x)=13\cdot 2^x\cdot\begin{cases}\max\{\frac{2^{x-\delta}}{x-\delta},x\} & \text{if } \delta+1\leq x\\ x & \text{otherwise}\end{cases}$$

Theorem 5.1 tells us that for all $k \in \mathbb{N}_{>0}$, we have

$$\begin{aligned}
\mathbf{LNBP}_\delta(\mathrm{ISA}_{k,k}) &\leq \begin{cases} 12 \cdot 2^k \max\{\frac{2^{k-\delta}}{k-\delta}, k\} + \frac{2^{2k-\delta}}{k-\delta} & \text{if } \delta+1 \leq k \\ 2^k(3k+1) + 2 \cdot 2^k & \text{otherwise} \end{cases} \\
&\leq \begin{cases} 2^k\big(12 \max\{\frac{2^{k-\delta}}{k-\delta}, k\} + \frac{2^{k-\delta}}{k-\delta}\big) & \text{if } \delta+1 \leq k \\ 6 \cdot 2^k k & \text{otherwise} \end{cases} \qquad \text{as } k \geq 1 \\
&\leq g(k)
\end{aligned}$$

and moreover, $\frac{g(k+1)}{g(k)} \leq 4$ for all $k \in \mathbb{N}_{>0}$. Indeed, let $k \in \mathbb{N}_{>0}$, there are two cases to consider:

- if $g(k+1) = 13 \cdot 2^{k+1}(k+1)$ then notice that we always have $g(k) \geq 13 \cdot 2^k \cdot k$. Therefore we get $\frac{g(k+1)}{g(k)} \leq \frac{13 \cdot 2^{k+1}(k+1)}{13 \cdot 2^k k} = 2(1+\frac{1}{k}) \leq 4$;
- otherwise $g(k+1) = 13 \cdot 2^{k+1} \frac{2^{k+1-\delta}}{k+1-\delta}$ and notice that either $g(k) \geq 13 \cdot 2^k \frac{2^{k-\delta}}{k-\delta}$ or $k = \delta$. If $k = \delta$ it is simple to check that $\frac{g(k+1)}{g(k)} \leq 4$, otherwise we have $\frac{g(k+1)}{g(k)} \leq \frac{13 \cdot 2^{k+1} \frac{2^{k+1-\delta}}{k+1-\delta}}{13 \cdot 2^k \frac{2^{k-\delta}}{k-\delta}} = 4 \cdot \frac{k-\delta}{k+1-\delta} \leq 4$.

Therefore, by Lemma 4.1, any $b \in \mathcal{N}_{\mathbf{LNBP}_\delta}$ verifies

$$\begin{aligned}
b(m) &\leq 4 \cdot \frac{g(\log_2 \log_2 m)}{\log_2 m} \\
&= 4 \cdot \frac{13 \cdot 2^{\log_2 \log_2 m} \begin{cases} \max\{\frac{2^{\log_2 \log_2(m)-\delta}}{\log_2 \log_2(m)-\delta}, \log_2 \log_2 m\} & \text{if } \delta+1 \leq \log_2 \log_2 m \\ \log_2 \log_2 m & \text{otherwise} \end{cases}}{\log_2 m} \\
&= 52 \cdot \begin{cases} \max\{\frac{\log_2 m}{2^\delta(\log_2 \log_2(m)-\delta)}, \log_2 \log_2 m\} & \text{if } 2^{2^{\delta+1}} \leq m \\ \log_2 \log_2 m & \text{otherwise} \end{cases} \\
&\leq c \cdot b_{\mathbf{LNBP}_\delta}(m)
\end{aligned}$$

for all $m \in \mathbb{N}, m \geq 4$, where $c \in \mathbb{R}_{>0}$ is a sufficiently large constant.

In the case where $\delta = 0$ notice that for $m \geq 4$ we have
$\log_2 \log_2 m \leq d \cdot \frac{\log_2 x}{\log_2 \log_2 x}$ for some suitable constant d.
So we can also conclude that for any $b \in \mathcal{N}_{\mathbf{BP}} = \mathcal{N}_{\mathbf{LNBP}_0}$, we have

$$b(m) \leq 52 \cdot \max\Big\{\frac{\log_2 m}{\log_2 \log_2 m}, \log_2 \log_2 m\Big\} \leq c' \cdot \frac{\log_2 m}{\log_2 \log_2 m}$$

for all $m \in \mathbb{N}, m \geq 4$, where $c' \in \mathbb{R}_{>0}$ is a sufficiently large constant. □

Finally, using this and Lemma 4.2, we get the following result, showing that the asymptotically greatest lower bound we may expect using Nečiporuk's method for $\mathbf{LNBP}_\delta$ for any $\delta \in \mathbb{N}$ is (asymptotically) equivalent to the lower bound for ISA given in Proposition 7.2.

Theorem 7.4. *For any family of Boolean functions $F = \{f_n\}_{n\in\mathbb{N}}$ and any $\Delta\colon \mathbb{N} \to \mathbb{N}$, $\mathrm{N}_F^{\mathbf{LNBP}_{\Delta(n)}}(n) \in \mathrm{O}\big(\Gamma_{\mathbf{LNBP}}(n, \Delta(n))\big)$.*

In particular, $\mathrm{N}_F^{\mathbf{BP}}(n) \in \mathrm{O}\big(\frac{n^2}{\log_2^2 n}\big)$.

Proof. Let $\delta \in \mathbb{N}$. We aim at applying Lemma 4.2 which requires four hypotheses, (i) to (iv).

For (i), we set h as $h_{\mathbf{LNBP}_\delta}(x) = \begin{cases} \max\{\frac{\log_2 x}{2^\delta(\log_2\log_2(x)-\delta)}, \log_2\log_2 x\} & \text{if } 2^{2^{\delta+1}} \leq x \\ \log_2\log_2 x & \text{otherwise .} \end{cases}$ and $x_0 = 2^8$. One can verify that h is non-decreasing on $\big[2^8, +\infty\big[$.

For (ii), for all $x \in [4, +\infty[$, we have $h(2^x) \geq \log_2\log_2(2^x) = \log_2 x$.

For (iii), for all $v, v' \in \mathbb{N}$ verifying $2^{2^v} \geq 2^8$ and $2^{2^{v'}} \geq 2^8$, we need to show that $h(2^{2^v}) + h(2^{2^{v'}}) \leq h(2^{2^{v+v'}})$. There are three cases to consider.

- If $h(2^{2^v}) = \log_2\log_2(2^{2^v}) = v$ and $h(2^{2^{v'}}) = \log_2\log_2(2^{2^{v'}}) = v'$, then $h(2^{2^v}) + h(2^{2^{v'}}) = v + v' = \log_2\log_2(2^{2^{v+v'}}) \leq h(2^{2^{v+v'}})$.

- If $h(2^{2^v}) = \frac{\log_2(2^{2^v})}{2^\delta(\log_2\log_2(2^{2^v})-\delta)} = \frac{2^{v-\delta}}{v-\delta} > v$ and $h(2^{2^{v'}}) = \log_2\log_2(2^{2^{v'}}) = v'$, then we necessarily have $v = \delta + \eta$ for some $\eta > 0$.

  Notice that $\eta \geq 2$ because if $\eta = 1$ then $v < 2$, a contradiction.

  We conclude by showing that $h(2^{2^v}) + h(2^{2^{v'}}) = \frac{2^{v-\delta}}{v-\delta} + v' \leq \frac{2^{\eta+v'}}{\eta+v'} \leq h(2^{2^{v+v'}})$.

  Only the first inequality is non immediate. To see it, consider the function $f(x) = 2^{\eta+x} - (\frac{2^\eta}{\eta} + x)(\eta + x)$. A simple calculation shows that it is non-decreasing for $x \geq 2$ and $\eta \geq 2$. The inequality follows as $f(2)$ is non-negative when $\eta \geq 2$.

- In the remaining case $h(2^{2^v}) = \frac{2^{v-\delta}}{v-\delta} > v$ and $h(2^{2^{v'}}) = \frac{2^{v'-\delta}}{v'-\delta} > v'$. It implies that $v \geq \delta+1$ and $v' \geq \delta + 1$. Arguing as above we actually have $v \geq \delta + 2$ and $v' \geq \delta + 2$, otherwise v or v' would be smaller than 2. We then have:
  $h(2^{2^v}) + h(2^{2^{v'}}) = \frac{2^{v-\delta}}{v-\delta} + \frac{2^{v'-\delta}}{v'-\delta} \leq \frac{2^{v+v'-2\delta}}{(v-\delta)(v'-\delta)} \leq \frac{2^{v+v'-2\delta}}{v+v'-2\delta} \leq h(2^{2^{v+v'-\delta}}) \leq h(2^{2^{v+v'}})$.
  The first and second inequality are because $x + y \leq xy$ when both x and y are greater than 2 (in the second case we use that $v - \delta \geq 2$ and $v' - \delta \geq 2$). The third one is by definition of h and the last one by monotonicity of h.

For (iv), by Proposition 7.3, we know that any $b \in \mathcal{N}_{\mathbf{LNBP}_\delta}$ is such that $b(m) \leq \alpha \cdot h(m)$ for all $m \geq 4$.

We can therefore apply Lemma 4.2 with $x_0 = 2^8$ and get that for any family of Boolean functions $F = \{f_n\}_{n\in\mathbb{N}}$ and all $n \in \mathbb{N}, n \geq 8$,

$$\begin{aligned} \mathrm{N}_F^{\mathbf{LNBP}_\delta}(n) &\leq \alpha \cdot \big(4 + h(2^8)\big) \cdot \frac{n}{\log_2 n} \cdot h(2^n) \\ &= c \cdot \frac{n}{\log_2 n} \cdot \begin{cases} \max\{\frac{n}{2^\delta(\log_2(n)-\delta)}, \log_2 n\} & \text{if } 2^{2^{\delta+1}} \leq 2^n \\ \log_2 n & \text{otherwise} \end{cases} \\ &= c \cdot \begin{cases} \max\{\frac{n^2}{2^\delta(\log_2(n)-\delta)\log_2 n}, n\} & \text{if } 2^{\delta+1} \leq n \\ n & \text{otherwise} \end{cases} \\ &= c \cdot \Gamma_{\mathbf{LNBP}}(n, \delta) \ . \end{aligned}$$

Thus, since this holds for all $\delta \in \mathbb{N}$, we get the desired result. □

8 Deterministic and Limited Nondeterministic Formulas

In this section, we focus on the model of Boolean binary formulas and its limited nondeterministic variant. **L** is one of the two measures that were considered in Nečiporuk's original

article [15] who gave an $\Omega(\frac{n^2}{\log_2 n})$ lower bound for this complexity measure. If the model is restricted to the case of binary formulas where only 2-ary AND and OR gates can be used, stronger lower bounds can be proven, the best known for instance being almost cubic and due to Håstad (see [5]). Just as in Section 7, results for the Nečiporuk method for binary formulas are known (see for instance [20, Chapter 8, Section 7]), but we do not know about any attempt to consider the method in its full generality: an approach that would explicitly try to find the best Nečiporuk function rather than just giving one, as done in [1] for the case of BPs.

Concerning limited nondeterministic binary formulas, Nečiporuk's lower bound method never seems to have been applied to the associated complexity measure, at least in a direct combinatorial sense that excludes Klauck's communication complexity formulation of the method [10].

For all $\delta \in \mathbb{N}$, let us define the function $b_{\mathbf{LL}_\delta} \colon \mathbb{N}_{>0} \to \mathbb{N}$ given by

$$b_{\mathbf{LL}_\delta}(m) = \begin{cases} \left\lceil \frac{1}{4} \max\left\{ \frac{\log_2 m}{2^\delta}, \log_2 \log_2 m \right\} \right\rceil & \text{if } m \geq 4 \\ 0 & \text{otherwise} \end{cases}$$

for all $m \in \mathbb{N}_{>0}$. We denote by $b_{\mathbf{L}}$ the case of $b_{\mathbf{LL}_0}$.

We first prove that $b_{\mathbf{L}} \in \mathcal{N}_{\mathbf{L}}$ and $b_{\mathbf{LL}_\delta} \in \mathcal{N}_{\mathbf{LL}_\delta}$. This is similar to the limited nondeterministic branching program case.

Proposition 8.1. *$b_{\mathbf{LL}_\delta} \in \mathcal{N}_{\mathbf{LL}_\delta}$ for all $\delta \in \mathbb{N}$. In particular, $b_{\mathbf{L}} \in \mathcal{N}_{\mathbf{L}}$.*

Proof. Let $\delta \in \mathbb{N}$. It is fairly obvious that $b_{\mathbf{LL}_\delta}$ is non-decreasing.

Let f be a n-ary boolean function on V and let $V_1, \ldots, V_p$ a partition of V. Let ϕ be a Boolean δ-LNBF computing f and let g be the $(n+\delta)$-ary Boolean function computed by ϕ when considering the δ nondeterministic bits as regular input variables.

For all $i \in [p]$ we will denote by $s_i \in \mathbb{N}$ the number of leaves in ϕ labelled by literals whose variable indices are in V_i, as well as $q \in \mathbb{N}$ the number of leaves in ϕ labelled by literals whose variable indices are not in V. It is clear that $|\phi| = \sum_{i=1}^{p} s_i + q \geq \sum_{i=1}^{p} s_i$. To conclude it remains to show that $s_i \geq b_{\mathbf{LL}_\delta}(r_{V_i}(f))$ for all $i \in [p]$.

Fix $i \in [p]$. The claim is obvious if $r_{V_i}(f) \leq 3$ hence we assume $r_{V_i}(f) \geq 4$. Let V_i' be the subset of V_i containing all indices of variables on which f depends. Then, by Lemma 2.2, $r_{V_i}(f) = r_{V_i'}(f)$. Moreover for each $l \in V_i'$, ϕ contains at least one leaf labelled by l. By Lemma 2.1, it follows that $r_{V_i}(f) = r_{V_i'}(f) \leq 2^{2^{|V_i'|}} \leq 2^{2^{s_i}}$. So we can conclude that $s_i \geq \log_2 \log_2(r_{V_i}(f))$.

If $r_{V_i}(f) \leq 2^{2^{\delta+1}}$, we have

$$\left\lceil \frac{1}{4} \cdot \frac{\log_2(r_{V_i}(f))}{2^\delta} \right\rceil \leq \left\lceil \frac{1}{4} \cdot \frac{\log_2\left(2^{2^{\delta+1}}\right)}{2^\delta} \right\rceil = 1 \leq \left\lceil \frac{1}{4} \cdot \log_2 \log_2(r_{V_i}(f)) \right\rceil \ .$$

and therefore $s_i \geq b_{\mathbf{LL}_\delta}(r_{V_i}(f))$.

It remains to consider the case where $r_{V_i}(f) > 2^{2^{\delta+1}}$. Notice that this implies $s_i > 0$, as $2^{2^{s_i}} \geq r_{V_i}(f)$.

This part of the proof is taken from classical references, e.g. [20, Proof of Theorem 7.1] or [8, Proof of Theorem 6.16]. We denote by T_i the sub-tree of ϕ consisting of all paths from a leaf with a label in V_i to the root of ϕ. This tree has nodes of fan-in 0, 1 or 2 and is non-empty since $s_i > 0$. Let W_i be the set of nodes of T_i that have fan-in 2 and notice that $|W_i| \leq s_i - 1$. Let P_i be the set of paths in T_i starting from a leaf or a node in W_i and ending in a node in W_i or in the root of T_i and containing no node in W_i as inner node. Notice that $|P_i| \leq 2\,|W_i| + 1 \leq 2s_i$.

For any partial assignment $\rho \in \{0,1\}^{V \setminus V_i \cup [\delta]}$, we obtain a formula $\phi|_\rho$ of size s_i computing $g|_\rho$ by replacing each variable in $V \setminus V_i \cup [\delta]$ by the appropriate constant given by ρ. This assignment induces that any part of $\phi|_\rho$ corresponding to a path p in P_i, either computes a constant function, or is the identity or negates its input. Reciprocally any of these four choices on p induces a subfunction of g. Hence we have $r_{V_i}(g) \leq 4^{|P_i|} \leq 2^{4s_i}$.

As g is a proof-checker function for f, from Lemma 2.7 it follows that $r_{V_i}(g) \geq r_{V_i}(f)^{\frac{1}{2^\delta}}$, therefore

$$s_i \geq \frac{1}{4}\log_2(r_{V_i}(g)) \geq \frac{1}{4}\log_2\Big(r_{V_i}(f)^{\frac{1}{2^\delta}}\Big) = \frac{1}{4}\cdot\frac{\log_2(r_{V_i}(f))}{2^\delta}\ .$$

Altogether, we have

$$s_i \geq \frac{1}{4}\max\Big\{\frac{\log_2(r_{V_i}(f))}{2^\delta}, \log_2\log_2(r_{V_i}(f))\Big\}\ ,$$

which implies that $s_i \geq b_{\mathbf{LL}_\delta}(r_{V_i}(f))$ as s_i is integral.

In conclusion for any n-ary Boolean function f on V and any partition $V_1, \ldots, V_p$ of V, it holds that $\mathbf{LL}_\delta(f) \geq \sum_{i=1}^p b_{\mathbf{LL}_\delta}(r_{V_i}(f))$, hence $b_{\mathbf{LL}_\delta} \in \mathcal{N}_{\mathbf{LL}_\delta}$. □

Using this and Lemma 2.6, we can immediately derive the following asymptotic lower bound on $\mathrm{N}_{\mathrm{ISA}}^{\mathbf{LL}_{\Delta(n)}}$.

Proposition 8.2. *$\mathrm{N}_{\mathrm{ISA}}^{\mathbf{LL}_{\Delta(n)}}(n) \in \Omega\big(\max\{\frac{n^2}{2^{\Delta(n)}\log_2 n}, n\}\big)$ for any $\Delta\colon \mathbb{N} \to \mathbb{N}$. In particular, $\mathrm{N}_{\mathrm{ISA}}^{\mathbf{L}}(n) \in \Omega\big(\frac{n^2}{\log_2 n}\big)$.*

Proof. Let $\Delta\colon \mathbb{N} \to \mathbb{N}$. Let $n \in \mathbb{N}, n \geq 32$. Let $V_1, \ldots, V_p, U$ be a partition of V such that $r_{V_i}(\mathrm{ISA}_n) = 2^q$ for all $i \in [p]$ where $p, q \in \mathbb{N}_{>0}$ verify $p \geq \frac{1}{32}\cdot\frac{n}{\log_2 n}$ and $q \geq \frac{n}{16}$ as given by Lemma 2.6. We have

$$\begin{aligned}
\mathrm{N}_{\mathrm{ISA}}^{\mathbf{LL}_{\Delta(n)}}(n) &\geq \sum_{i=1}^p b_{\mathbf{LL}_{\Delta(n)}}(r_{V_i}(\mathrm{ISA}_n)) + b_{\mathbf{LL}_{\Delta(n)}}(r_U(\mathrm{ISA}_n)) \\
&\geq \sum_{i=1}^p \frac{1}{4}\cdot\max\Big\{\frac{\log_2(2^q)}{2^{\Delta(n)}}, \log_2\log_2(2^q)\Big\} \\
&= p\cdot\frac{1}{4}\cdot\max\Big\{\frac{q}{2^{\Delta(n)}}, \log_2 q\Big\} \\
&\geq c_1\cdot\frac{n}{\log_2 n}\cdot\max\Big\{\frac{n}{16\cdot 2^{\Delta(n)}}, \log_2\Big(\frac{n}{16}\Big)\Big\} \\
&\geq c_2\cdot\max\Big\{\frac{n^2}{2^{\Delta(n)}\log_2 n}, n\Big\} \quad \text{because } n \geq 32 \text{ implies } \log_2(\tfrac{n}{16}) \geq \frac{\log_2 n}{16}\ .
\end{aligned}$$

□

We now show that for all $\delta \in \mathbb{N}$, $b_{\mathbf{LL}_\delta}$ is in fact an asymptotically largest function in $\mathcal{N}_{\mathbf{LL}_\delta}$. To this end, we appeal to the upper bound on $\mathbf{LL}_\delta(\mathrm{ISA}_{k,\ell})$ from Theorem 5.1 and apply Lemma 4.1.

Proposition 8.3. *There exists a constant $c \in \mathbb{R}_{>0}$ verifying that for each $\delta \in \mathbb{N}$, any $b \in \mathcal{N}_{\mathbf{LL}_\delta}$ is such that $b(m) \leq c\cdot b_{\mathbf{LL}_\delta}(m)$ for all $m \in \mathbb{N}, m \geq 4$.*

Proof. Fix $\delta \in \mathbb{N}$. Let $g\colon [1, +\infty[\to \mathbb{R}_{\geq 0}$ be the non-decreasing function defined by $g(x) = 15\cdot 2^x\cdot\max\{2^{x-\delta}, x\}$.

Notice that $\frac{g(k+1)}{g(k)} \leq 4$ for all $k \in \mathbb{N}_{>0}$.

Moreover, from Theorem 5.1 we have:

$$\mathbf{LL}_\delta(\mathrm{ISA}_{k,k}) \leq 12 \cdot 2^k \cdot \max\{2^{k-\delta}, k\} + 3 \cdot 2^k \leq g(k).$$

Therefore, by Lemma 4.1, any $b \in \mathcal{N}_{\mathbf{LL}_\delta}$ verifies

$$\begin{aligned} b(m) &\leq 4 \cdot \frac{g(\log_2 \log_2 m)}{\log_2 m} \\ &= 4 \cdot \frac{15 \cdot 2^{\log_2 \log_2 m} \cdot \max\{2^{\log_2 \log_2(m)-\delta}, \log_2 \log_2 m\}}{\log_2 m} \\ &= 60 \cdot \max\Big\{\frac{\log_2 m}{2^\delta}, \log_2 \log_2 m\Big\} \end{aligned}$$

for all $m \geq 4$. □

Finally, using this and Lemma 4.2, we show that asymptotically the greatest lower bound we may expect using Nečiporuk's method is the one obtained for ISA in Proposition 8.2.

Theorem 8.4. *For any family of Boolean functions $F = \{f_n\}_{n\in\mathbb{N}}$ and any $\Delta\colon \mathbb{N} \to \mathbb{N}$, $\mathrm{N}_F^{\mathbf{LL}_{\Delta(n)}}(n) \in \mathrm{O}\big(\max\{\frac{n^2}{2^{\Delta(n)}\log_2 n}, n\}\big)$.*

In particular, $\mathrm{N}_F^{\mathbf{L}}(n) \in \mathrm{O}\big(\frac{n^2}{\log_2 n}\big)$.

Proof. Fix $\delta \in \mathbb{N}$. We aim at applying Lemma 4.2 which requires four hypotheses.

For (i), let $h\colon [4, +\infty[\to \mathbb{R}_{\geq 0}$ be the function defined by

$$h(x) = \max\Big\{\frac{\log_2 x}{2^\delta}, \log_2 \log_2 x\Big\}$$

and $x_0 = 2^8$; as required, h is non-decreasing on $\big[2^8, +\infty\big[$.

For (ii), for all $x \in [4, +\infty[$, we have $h(2^x) \geq \log_2 \log_2(2^x) = \log_2 x$.

For (iii), for all $v, v' \in \mathbb{N}$ verifying $2^{2^v} \geq 2^8$ and $2^{2^{v'}} \geq 2^8$, we have $h(2^{2^v}) + h(2^{2^{v'}}) \leq h(2^{2^{v+v'}})$. Indeed, let $v, v' \in \mathbb{N}$ such that $2^{2^v} \geq 2^8$ and $2^{2^{v'}} \geq 2^8$, there are two cases to consider.

- If $h(2^{2^v}) = \log_2 \log_2(2^{2^v}) = v$ and $h(2^{2^{v'}}) = \log_2 \log_2(2^{2^{v'}}) = v'$, then $h(2^{2^v}) + h(2^{2^{v'}}) = v + v' = \log_2 \log_2(2^{2^{v+v'}}) \leq h(2^{2^{v+v'}})$.
- Otherwise, there is at least one $w \in \{v, v'\}$ such that $h(2^{2^w}) = \frac{\log_2(2^{2^w})}{2^\delta} = 2^{w-\delta}$: assume without loss of generality that it is v. Then, since $2^{2^v} \geq 16$, we have $v \geq 2$, so by hypothesis, it follows that $2^{v-\delta} > v \geq 2$. Moreover, $h(2^{2^{v'}}) = \max\{2^{v'-\delta}, v'\} \leq 2^{v'}$, so using our usual observation about the relationship between the sum and the product of two real numbers greater than or equal to 2, we get $h(2^{2^v}) + h(2^{2^{v'}}) \leq 2^{v-\delta} + 2^{v'} \leq 2^{v+v'-\delta} = \frac{\log_2(2^{2^{v+v'}})}{2^\delta} \leq h(2^{2^{v+v'}})$.

For (iv), by Proposition 8.3, we know that any $b \in \mathcal{N}_{\mathbf{LL}_\delta}$ is such that $b(m) \leq \alpha \cdot h(m)$ for all $m \geq 4$.

Therefore, by Lemma 4.2, for any family of Boolean functions $F = \{f_n\}_{n\in\mathbb{N}}$ and all $n \geq 8$, we have:

$$\begin{aligned} \mathrm{N}_F^{\mathbf{LL}_\delta}(n) &\leq \alpha \cdot \big(4 + h(\lfloor 2^8 \rfloor)\big) \cdot \frac{n}{\log_2 n} \cdot h(2^n) \\ &\leq c \cdot \frac{n}{\log_2 n} \cdot \max\Big\{\frac{n}{2^\delta}, \log_2 n\Big\} \\ &= c \cdot \max\Big\{\frac{n^2}{2^\delta \log_2 n}, n\Big\}\,. \end{aligned}$$

□

9 Conclusion

We have proposed a general interpretation of what it means to say "the method of Nečiporuk". We have applied the method to several complexity measures, as reported in Table 1, and shown in particular that the limitations of the method are very much determined by the complexity of the Indirect Storage Access function under each measure, at least for those we studied in this paper. In passing, we note that this is also the case for the size measure of in-degree 2 Boolean circuits: while one can show that $m \to \lceil \log_2 \log_2 m \rceil - 1$ is a Nečiporuk function for that measure, one sees that $\mathrm{ISA}_{k,\ell}$ can be computed by a Boolean circuit of size at most $2^k(k + 2\ell) + 3 \cdot 2^\ell$ (so in linear size). This implies asymptotic optimality of that Nečiporuk function and further implies that the best lower bound one might expect using Nečiporuk's method for the size of Boolean circuits is linear. Thus, our framework also applies to Boolean circuits, even though it only allows to re-derive the well-known result that Nečiporuk's method is unable prove super-linear circuit size lower bounds. Indeed, as shown by Uhlig in 1991, there exist Boolean functions having exponentially many different subfunctions that can be computed by linear-size Boolean circuits (see [8, Remark 6.19]). Note that our focus was not on optimizing the constant factors in the bounds obtained, most of which can certainly be improved.

Our abstract definition of a Nečiporuk function is inspired by Alon and Zwick [1]. It has the benefit of not specifying the way in which such a function is obtained, be it some "semantic count" of the number of different Boolean functions computable with a given cost, some "syntactic count" of the number of different devices of that cost as done usually, or any other technique. While in the literature, "Nečiporuk-style theorems" refer to giving an explicit Nečiporuk function as defined in step 1 in Definition 4.1 [20, 8, 1], it is natural to ask whether we could even further twist the definition of a Nečiporuk function to get more out of the method.

Looking at our meta-results and how we draw the limitation results for Nečiporuk's method used for a specific measure $\mathbf{M}$, namely using an upper bound on the $\mathrm{ISA}_{k,k}$ function for all $k \in \mathbb{N}_{>0}$, we observe that the main weakness of the method is that a Nečiporuk function for $\mathbf{M}$ should verify the conditions presented in step 1 of Definition 4.1 for *any* Boolean function. The natural question is therefore whether restricting the class of Boolean functions for which these conditions should be verified by a Nečiporuk function for $\mathbf{M}$ would allow to get stronger Nečiporuk functions (and thus, lower bounds) for $\mathbf{M}$ for this specific class of Boolean functions. This seems to be an interesting question to us, but is not treated in this paper. Another interesting similar question that was suggested to us by one of the anonymous referees is whether relaxing the condition "for any partition" in step 1 of Definition 4.1 to "there exists a partition" would allow to get stronger Nečiporuk functions.

Acknowledgements We would like to thank the anonymous referees for their helpful comments and suggestions.

References

[1] Alon, N., Zwick, U.: On Nečiporuk's theorem for branching programs. Theor. Comput. Sci. **64**(3) (1989) 331–342

[2] Boppana, R.B., Sipser, M.: The complexity of finite functions. In van Leeuwen, J., ed.: Handbook of Theoretical Computer Science. Volume A. Elsevier (1990) 757–804

[3] Cobham, A.: The recognition problem for the set of perfect squares. In: 7th Annual Symposium on Switching and Automata Theory, Berkeley, California, USA, October 23-25, 1966. (1966) 78–87

[4] Goldsmith, J., Levy, M.A., Mundhenk, M.: Limited nondeterminism. SIGACT News **27**(2) (1996) 20–29

[5] Håstad, J.: The shrinkage exponent of de Morgan formulas is 2. SIAM J. Comput. **27**(1) (1998) 48–64

[6] Hromkovic, J., Schnitger, G.: Nondeterministic communication with a limited number of advice bits. SIAM J. Comput. **33**(1) (2003) 43–68

[7] Jukna, S.: Extremal combinatorics - with applications in computer science. Springer (2001)

[8] Jukna, S.: Boolean function complexity: advances and frontiers. Volume 27. Springer (2012)

[9] Karchmer, M., Wigderson, A.: On span programs. In: Proceedings 8th Structure in Complexity Theory, IEEE Computer Society Press (1993) 102–111

[10] Klauck, H.: One-way communication complexity and the Nečiporuk lower bound on formula size. SIAM J. Comput. **37**(2) (2007) 552–583

[11] Lupanov, O.B.: A method of circuit synthesis. Izvestia V.U.Z. Radiofizika **1** (1958) 120–140

[12] Lupanov, O.B.: On the complexity of the realization of the functions of an algebra of logic by formulas. Problemy Kibernet. Vyp **3** (1960) 61–80

[13] Masek, W.: A fast algorithm for string editing problem and decision graph complexity. Technical report, Massachussetts Institute of Technology (1976)

[14] Nečiporuk, È.: On the complexity of schemes in some bases containing nontrivial elements with zero weights. Problemy Kibernetiki **8** (1962) 123–160 (in Russian).

[15] Nečiporuk, È.: On a boolean function. Doklady of the Academy of the USSR **169**(4) (1966) 765–766 Translation: *Soviet Math. Doklady* 7:4, pp. 999-1000.

[16] Paul, W.: Komplexitätstheorie. Leitfäden der angewandten Mathematik und Mechanik LAMM. Teubner Studienbücher (1978)

[17] Pudlák, P.: The hierarchy of boolean circuits. Computers and artificial intelligence **6**(5) (1987) 449–468

[18] Razborov, A.: Lower bounds for deterministic and nondeterministic branching programs. In: 8th Internat. Symp. on Fundamentals of Computation Theory. (1991) 47–60

[19] Savage, J.E.: The Complexity of Computing. John Wily, New York (1976)

[20] Wegener, I.: The Complexity of Boolean Functions. Wiley-Teubner series in computer science. B. G. Teubner & John Wiley, Stuttgart (1987)

[21] Wegener, I.: Branching Programs and Binary Decision Diagrams: Theory and Applications. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia (2000)