Electronic Colloquium on Computational Complexity, Report No. 57 (2000)

# An Improved Hierarchy Result for Partitioned BDDs[*]

Martin Sauerhoff[†]
FB Informatik, LS 2, Univ. Dortmund, 44221 Dortmund, Germany
e-mail: sauerhof@ls2.cs.uni-dortmund.de

**Abstract.** One of the great challenges of complexity theory is the problem of analyzing the dependence of the complexity of Boolean functions on the resources nondeterminism and randomness. So far, this problem could be solved only for very few models of computation. For so-called *partitioned binary decision diagrams,* which are a restricted variant of nondeterministic read-once branching programs, Bollig and Wegener have proven an astonishing hierarchy result which shows that the smallest possible decrease of the available amount of nondeterminism may incur an exponential blow-up of the branching program size.

They have shown that $k$-partitioned BDDs which may nondeterministically choose between $k$ alternative subprograms may be exponentially larger than $(k+1)$-partitioned BDDs for the same function if $k = o\big((\log n/\operatorname{loglog} n)^{1/2}\big)$, where $n$ is the input size. In this paper, an improved hierarchy result is established which still works if the number of nondeterministic decisions is $O\big((n/\log^{1+\varepsilon} n)^{1/4}\big)$, where $\varepsilon > 0$ is an arbitrary small constant.

**Keywords:** Branching programs, partitioned BDDs, communication complexity, nondeterminism, hierarchies, lower bounds

## 1 Introduction and Definitions

Besides circuits and formulae, branching programs belong to the most important nonuniform models of computation. For complexity theory, this model is interesting since branching programs are combinatorially easier to handle than, e. g., Turing machines, and facilitate the development of new lower bound techniques for space-bounded computation. On the other hand, several restricted variants of branching programs have turned out to be useful as data structures for Boolean functions which have found widespread application, most prominently perhaps for hardware verification (see Bryant [6] for a survey on some application aspects).

**Definition 1.** A *(deterministic) branching program (BP)* on the variable set $\{x_1, \ldots, x_n\}$ is a directed, acyclic graph with one source and two sinks, the latter labeled by $0$ and $1$, resp. The interior nodes are labeled by a variable and have two outgoing edges labeled by $0$ and $1$, resp.

ISSN 1433-8092

[*]To appear in *Theory of Computing Systems*.

[†]This work has been supported by DFG grant We 1066/8-2.

This graph represents a function $f\colon \{0,1\}^n \to \{0,1\}$ defined on $\{x_1, \ldots, x_n\}$ in the following way. For a given input assignment $a \in \{0,1\}^n$, call an edge labeled by $c$ and leaving an $x_i$-node *activated by* $a$ if $a_i = c$. The assignment $a$ defines a unique path of activated edges in the branching program, starting at the source and ending in one of the sinks whose label is the output value $f(a)$. The *size of* $G$, $|G|$, is the number of its nodes. The *branching program size of* $f$ is the minimum size of a branching program representing $f$.

For a thorough introduction of this model, we have to refer to the literature, e. g., the monographs [19, 20] of Wegener. Nondeterminism can be incorporated into branching programs in the following, straightforward way.

**Definition 2.** A *nondeterministic branching program* has the same structure as a deterministic branching program, but may additionally contain *nondeterministic nodes* which are not labeled by a variable and may have an arbitrary number of outgoing unlabeled edges. The edges leaving nondeterministic nodes are always activated. The nondeterministic branching program computes $1$ on a given input if there is a path of activated edges leading from the source to the $1$-sink. The *size of a nondeterministic branching program* is the sum of the contributions of all nodes, where a nondeterministic node with $k$ outgoing edges contributes $k-1$ to the size while a usual node contributes $1$.

For a nondeterministic branching program which only contains nondeterministic nodes with two outgoing edges, the amount of nondeterminism used by the branching program is given by the maximal number of nondeterministic nodes on a path from the source to one of the sinks, which we refer to as the *(worst-case) number of nondeterministic guesses* of the branching program. For an arbitrary nondeterministic branching program, the number of nondeterministic guesses is defined as the minimal number used by an equivalent graph of equal size where all nondeterministic nodes with $k > 2$ outgoing edges are replaced by subgraphs of nondeterministic nodes with two outgoing edges.

It is a fundamental open problem to prove superpolynomial lower bounds on the size of branching programs even in the deterministic case, and the nondeterministic case seems to be harder still (see, e. g., Razborov [15]). Nevertheless, several interesting restricted variants of branching programs could be analyzed quite successfully, and for some of these models even exponential lower bounds could be proven. Among these are the following two basic restricted types of deterministic branching programs.

**Definition 3.** A *read-once branching program* is a branching program with the restriction that on each path from the source to a sink each variable is allowed to appear at most once as the label of a node.

Let $\pi$ be a permutation of the set $\{1, \ldots, n\}$. A $\pi$-*OBDD* (OBDD = ordered binary decision diagram) on the variable set $\{x_1, \ldots, x_n\}$ is a read-once branching program with the following additional *ordering restriction*: For each edge leading from a node labeled by some variable $x_i$ to a node labeled by $x_j$ it must hold that $\pi(i) < \pi(j)$. We call a graph an *OBDD* if it is a $\pi$-OBDD for some permutation $\pi$. The permutation $\pi$ is called the *variable ordering* of the OBDD.

Nondeterministic variants of these restricted types of branching programs are obtained in the obvious way by requiring that the nodes labeled by variables in a nondeterministic branching program fulfill the described restrictions.

Read-once branching programs were originally introduced in complexity theory, whereas OBDDs have become popular as a data structure for the representation of Boolean functions. For both of these types of branching programs, it is well-known how lower bounds can be proven in the deterministic case, and some exponential bounds are known even for nondeterministic and randomized variants (see, e. g., the papers [1, 2, 4, 5, 11, 12, 16–18]).

Apart from the question of whether nondeterminism allows an exponential decrease of the branching program size compared with the deterministic model (which could be shown, e. g., for OBDDs and read-once branching programs), it is a much more difficult task to analyze the exact dependence of the size on the available amount of nondeterminism. In this paper, we are concerned with the following variant of restricted nondeterministic branching programs which allows a very fine control of the available amount of nondeterminism.

**Definition 4.** A *$k$-partitioned BDD with variable orderings* $(\pi_1, \dots, \pi_k)$ is a nondeterministic read-once branching program with a single nondeterministic node at the top which has $k$ outgoing edges leading to the sources of OBDDs $G_1, \dots, G_k$. The OBDD $G_i$ is ordered according to $\pi_i$, where $i = 1, \dots, k$. A branching program is simply called a *partitioned BDD*, if a $k$ and $(\pi_1, \dots, \pi_k)$ exist such that it is a $k$-partitioned BDD with variable orderings $(\pi_1, \dots, \pi_k)$.

Jain, Bitner, Abraham, and Fussell [10] have introduced partitioned BDDs as a representation of Boolean functions for practical purposes. Bollig and Wegener [3] have analyzed this model from the complexity theoretical point of view. Among other results, they have proven that the classes of functions representable by $k$-partitioned BDDs in polynomial size form a proper hierarchy with respect to $k$. Hromkovič and the author [8] have investigated the relationship between partitioned BDDs and nondeterministic OBDDs to some extent and proven the following incomparability result. On the one hand, there is a sequence of functions with polynomial size for 2-partitioned BDDs, but exponential size for all nondeterministic OBDDs. On the other hand, there is also a sequence of functions which requires exponential size for $k$-partitioned BDDs if $k$ is logarithmically bounded in the input size, while nondeterministic OBDDs for these functions are only polynomially large.

For their hierarchy result, Bollig and Wegener have introduced the function $P_{k,n}$ ("path function") which tests a property of a cleverly constructed graph. The output is obtained by following directed paths in this graph which are determined by the input of the function.

**Theorem 1 (Bollig and Wegener, 1997).** *There is a sequence of (explicitly defined) Boolean functions $(P_{k,n})_{n \in \mathbb{N}}$ with the following properties.*

*(1) The function $P_{k,n}$ can be represented by $k$-partitioned BDDs of size $O(2^k k^2 n^k)$; but*

*(2) requires size $2^{\Omega(n^{1/(2k)} k^{-5} \log^{-1} n)}$ for $(k-1)$-partitioned BDDs.*

The upper bound in this theorem is only polynomial for constant $k$. By introducing "dummy vertices" in the graph for $P_{k,n}$, Bollig and Wegener have obtained a modified function for which the upper bound is polynomial while the lower bound is still superpolynomial for values of $k$ with $k = o\big((\log n/\operatorname{loglog} n)^{1/2}\big)$.

Theorem 1 shows that already the addition of a single edge to the nondeterministic node at the top may decrease the size of $k$-partitioned BDDs from an exponential to a polynomial function in $n$ (for small values of $k$). This is in contrast to the situation for usual nondeterministic OBDDs, where allowing a constant number of additional nondeterministic guesses may decrease the size only polynomially. (This is discussed in more detail in Section 3.)

In this paper, we improve the result of Bollig and Wegener by re-using some key ideas from their paper [3]. The new function which we consider here instead of $P_{k,n}$ allows us to obtain a polynomial upper bound directly, without having to rely on padding techniques. As a consequence, we obtain a larger gap between lower and upper bound.

The improved hierarchy now still works for values of $k$ which are exponentially larger than the maximal values in the result of Bollig and Wegener, namely for $k$ with $k = \Theta\left((N/\log^{1+\varepsilon} N)^{1/4}\right)$, where $N$ is the input size of the considered function and $\varepsilon > 0$ is an arbitrary small constant. Our proof of the hierarchy result explicitly uses tools from communication complexity theory (this is also implicitly done in the proof of Bollig and Wegener).

The rest of the paper is organized as follows. In Section 2, we review definitions from communication complexity theory and prove a combinatorial lemma which is an essential ingredient in the proof of the lower bound for the hierarchy result. Then we discuss some results concerning the dependence of communication complexity and the size of usual nondeterministic OBDDs on the resource nondeterminism as background information (Section 3). Finally, in Section 4, the improved hierarchy result for partitioned BDDs is presented and proven.

## 2 Tools from Communication Complexity Theory

We first give some basic definitions from communication complexity theory (for a thorough introduction to this field, see the monographs of Hromkovič [7] and Kushilevitz and Nisan [14]).

A *two-party communication protocol* is an algorithm by which two players, called Alice and Bob, cooperatively evaluate a function $f\colon X \times Y \rightarrow \{0,1\}$, where $X$ and $Y$ are finite sets. Alice obtains an input $x \in X$ and Bob an input $y \in Y$. The players determine $f(x,y)$ by sending messages to each other. Each player is assumed to have unlimited (but deterministic) computational power to compute their messages. The *(deterministic) communication complexity of $f$*, $D(f)$, is the minimal number of bits exchanged by a communication protocol by which Alice and Bob compute $f(x,y)$ for each input $(x,y) \in X \times Y$.

We give a more detailed definition of randomized and nondeterministic communication protocols and corresponding complexity measures.

**Definition 5.** A *randomized communication protocol* $P$ is a communication protocol where player Alice has inputs $(x, r_A)$ and player Bob inputs $(y, r_B)$. Here $x \in X$ and $y \in Y$ are as above in the deterministic case, and $r_A$ and $r_B$ are interpreted as (private) strings of *random bits* (whose lengths may depend on the length of $x$ and $y$, resp.).

The protocol $P$ is said to *compute* $f\colon X \times Y \rightarrow \{0,1\}$ *with error* $\varepsilon\colon X \times Y \rightarrow [0,1]$, if for all $(x,y) \in X \times Y$ the probability to choose assignments to $r_A$ and $r_B$ randomly (according to the uniform distribution) such that $P$ yields the output $f(x,y)$ on the combined input assignments $(x, r_A)$ and $(x, r_B)$ is $1 - \varepsilon(x,y)$. The error of $P$ is called *bounded* if there is some constant $\varepsilon_0 < 1/2$ such that $\varepsilon(x,y) \leq \varepsilon_0$ for all $(x,y) \in X \times Y$.

As in the deterministic case, the *complexity of* $P$, $c(P)$, is defined as the maximum number of bits exchanged by Alice and Bob according to $P$, taken over all choices of $(x, r_A)$ and $(y, r_B)$. The *number of random bits used by* $P$ is defined as the maximum of the number of bits in the strings $r_A$ and $r_B$ taken over all inputs $(x,y) \in X \times Y$.

Nondeterministic protocols are randomized protocols with (unbounded) one-sided error smaller than 1, by which we mean that $\varepsilon(x, y) < 1$ for all $(x, y) \in f^{-1}(1)$ and $\varepsilon(x, y) = 0$ for all $(x, y) \in f^{-1}(0)$. In this context, the random bits are usually called *(nondeterministic) advice bits*. The *nondeterministic communication complexity of $f$ with restriction to $r$ advice bits, $N_r(f)$*, is defined as the minimum of $c(P)$ over all nondeterministic protocols $P$ computing $f$ and using at most $r$ advice bits.

Here we are mainly concerned with the restricted variant of communication protocols known as *one-way communication protocols*. In a one-way communication protocol, Alice sends a single message to Bob who has to output the result of the protocol, which may depend on his input and the message he has obtained. We use $D^{\mathrm{A}\to\mathrm{B}}$ and $N_r^{\mathrm{A}\to\mathrm{B}}$ to denote the complexity measures for deterministic and nondeterministic one-way protocols. For technical reasons, it is convenient that for these measures we only count the number of bits sent by Alice alone (ignoring the output bit sent by Bob).

Next, we present a lower bound result for one-way protocols which will be used in the proof of the hierarchy for partitioned BDDs. We consider the following well-known function.

**Definition 6.** Let $|a|$ denote the integer with binary representation $a \in \{0, 1\}^*$. Define $\mathrm{INDEX}_n\colon \{0, 1\}^n \times \{0, 1\}^{\lceil \log n\rceil} \to \{0, 1\}$ on inputs $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_{\lceil \log n\rceil})$ by $\mathrm{INDEX}_n(x, y) := x_{|y|+1}$, if $|y| \in \{0, \ldots, n - 1\}$, and $\mathrm{INDEX}_n(x, y) := 0$, otherwise.

This function is referred to as the "index" or "pointer function" in the literature. We may interpret it as the description of direct storage access: The $x$-vector plays the role of the "memory contents," whereas the $y$-vector is an "address" in the memory.

Kremer, Nisan, and Ron [13] have shown that $\mathrm{INDEX}_n$ has complexity $\Omega(n)$ for randomized one-way communication protocols with bounded error. For the deterministic case, one even obtains $D^{\mathrm{A}\to\mathrm{B}}(\mathrm{INDEX}_n) = n$. In the following, we extend this by showing that functions $g$ with $g \le \mathrm{INDEX}_n$ (i. e., $g(x, y) \le \mathrm{INDEX}_n(x, y)$ for all $(x, y)$) also have large deterministic one-way complexity if they have to cover many 1-inputs of the function. We consider a slightly generalized function which will be useful for the proof of the hierarchy result.

**Lemma 1.** *Let $h\colon \{0, 1\}^{\lceil \log n\rceil} \to \{0, 1\}$ be a function with $h(y) = 0$ if $|y| \ge n$. We consider the function $\mathrm{INDEX}(h)_n\colon \{0, 1\}^n \times \{0, 1\}^{\lceil \log n\rceil} \to \{0, 1\}$ defined by $\mathrm{INDEX}(h)_n(x, y) := \mathrm{INDEX}_n(x, y) \oplus h(y)$. Let $g\colon \{0, 1\}^n \times \{0, 1\}^{\lceil \log n\rceil} \to \{0, 1\}$ be a function with $g \le \mathrm{INDEX}(h)_n$. Then*

$$D^{\mathrm{A}\to\mathrm{B}}(g) \ge |g^{-1}(1)|/2^n.$$

**Proof:** For an arbitrary function $f\colon X \times Y \to \{0, 1\}$, where $X, Y$ are finite sets, we define the *communication matrix $M_f$ of $f$* by $M_f(x, y) := f(x, y)$ for $x \in X$ and $y \in Y$. Thus $M_f$ is an $|X| \times |Y|$-matrix with entries from $\{0, 1\}$. Let $\mathrm{nrows}(M_f)$ be the number of different rows in $M_f$. The proof of the lemma is based on the simple fact that $D^{\mathrm{A}\to\mathrm{B}}(f) = \lceil \log(\mathrm{nrows}(M_f))\rceil$.

We first consider the case $h \equiv 0$, i.e., $\mathrm{INDEX}(h)_n = \mathrm{INDEX}_n$. Let $g$ be any function with $g \le \mathrm{INDEX}_n$. Consider the communication matrix $M_g$ of $g$. Our goal is to derive a lower bound on $\mathrm{nrows}(M_g)$ in terms of $|g^{-1}(1)|$. For this, we observe that a vector with many ones cannot occur in too many rows of $M_g$, since $g \le \mathrm{INDEX}_n$. Hence, $M_g$ will have many different rows if $|g^{-1}(1)|$ is large.

Now we make these ideas more precise. Let $a_1, \ldots, a_k \in \{0,1\}^{2^{\lceil \log n \rceil}}$ be the different vectors occurring as rows of $M_g$ (thus $k = \mathrm{nrows}(M_g)$). Let $R_i \subseteq \{0,1\}^n$ be the set of indices of rows in $M_g$ which are equal to $a_i$, for $i = 1, \ldots, k$. Then

$$\sum_{i=1}^{k} |R_i| \cdot \|a_i\| = |g^{-1}(1)|,$$

where $\|a_i\|$ denotes the number of ones in the vector $a_i$. It holds that $|R_1| + \cdots + |R_k| = 2^n$.

By the definition of $\mathrm{INDEX}_n$ and the fact that $g \leq \mathrm{INDEX}_n$, it holds that $a_{i,n} = a_{i,n+1} = \cdots = a_{i,2^{\lceil \log n \rceil}-1} = 0$. Hence, $\|a_i\| \leq n$.

Furthermore, we claim that $|R_i| \leq 2^{n-\|a_i\|}$, and thus also $\|a_i\| \leq n - \log |R_i|$. To see this, identify the rows of the communication matrices $M_g$ and $M_{\mathrm{INDEX}_n}$ with subsets of $\{1, \ldots, n\}$ in the obvious way. Notice that all subsets of $\{1, \ldots, n\}$ occur exactly once as rows of $M_{\mathrm{INDEX}_n}$. Due to the fact that $g \leq \mathrm{INDEX}_n$, the number of rows in $M_g$ where a vector with $\ell$ fixed one entries may occur is exactly the number of different subsets of $\{1, \ldots, n\}$ with $\ell$ fixed elements, which is $2^{n-\ell}$.

Taking all these facts together, we see that the number of 1-inputs of $g$ can be bounded from above by maximizing

$$\sum_{i=1}^{k} x_i \cdot (n - \log x_i)$$

with respect to $x_1, \ldots, x_k \in \{1, \ldots, 2^n\}$ under the constraint $x_1 + \cdots + x_k = 2^n$. By the method of Lagrange multipliers, we obtain $x_i = 2^n/k$ for $i = 1, \ldots, k$ and thus

$$|g^{-1}(1)| \leq \sum_{i=1}^{k} \frac{2^n}{k} \cdot \log k = 2^n \cdot \log k.$$

From this, the claim for $g \leq \mathrm{INDEX}_n$ follows, since $k = \mathrm{nrows}(M_g)$.

Finally, we consider the case of an arbitrary function $h$ with $h(y) = 0$ for $|y| \geq n$. We observe that the communication matrix for the modified function $\mathrm{INDEX}(h)_n$ is obtained from the communication matrix for $\mathrm{INDEX}_n$ by negating some of the columns with index in $\{0, \ldots, n-1\}$. The columns with index in $\{n, \ldots, 2^{\lceil \log n \rceil} - 1\}$ in the matrix of $\mathrm{INDEX}_n$ only contain zeros. Restricted to the first $n$ columns, the matrix contains each vector from $\{0,1\}^n$ exactly once as a row. It is easy to see that a negation of one of the first $n$ columns simply leads to a permutation of the vectors in the rows of the communication matrix. Hence, the general result follows by the same proof as that for the simple function $\mathrm{INDEX}_n$. $\square$

Finally, we describe how the above result will be used in the proof of the hierarchy result for partitioned BDDs. Although a partitioned BDD consists of OBDDs as subparts, its more general structure prevents the direct use of the known standard technique for proving lower bounds on the size of OBDDs. Nevertheless, we will see that the standard technique *can* be applied if it is combined with several combinatorial tricks to simplify the structure of the partitioned BDD.

It is well-known how lower bounds on the size of OBDDs can be proven by using lower bound results for one-way communication complexity (see, e. g., the monograph of Kushilevitz and Nisan [14]). This is called the *reduction technique* here for easier reference. We give a description of the technique using the formalism from [17]. This makes use of the following standard reducibility concept from communication complexity theory.

**Definition 7 (Rectangular reduction).** Let $X_f, Y_f$ and $X_g, Y_g$ be finite sets. Let $f\colon X_f \times Y_f \to \{0,1\}$ and $g\colon X_g \times Y_g \to \{0,1\}$ be arbitrary functions. Then we call a pair $(\varphi_1, \varphi_2)$ of functions $\varphi_1\colon X_f \to X_g$ and $\varphi_2\colon Y_f \to Y_g$ a *rectangular reduction from $f$ to $g$* (or simply "reduction" for short) if $g(\varphi_1(x), \varphi_2(y)) = f(x,y)$ for all $(x,y) \in X_f \times Y_f$. If such a pair of functions exists for $f$ and $g$, we say that $f$ *is reducible to* $g$.

**Lemma 2 (Reduction Technique).** *Let $g\colon \{0,1\}^n \to \{0,1\}$ be defined on the variable set $X = \{x_1, \ldots, x_n\}$. Let $\pi$ be a variable ordering on $X$. W. l. o. g. (by renumbering) we may assume that $\pi$ is described by $x_1, \ldots, x_n$.*

*Assume that there is a function $f\colon U \times V \to \{0,1\}$, where $U$ and $V$ are finite sets, and a parameter $p$ with $1 \le p \le n-1$ such that $f$ is reducible to $g\colon \{0,1\}^p \times \{0,1\}^{n-p} \to \{0,1\}$.*

*(1) Let $G$ be a deterministic OBDD for $g$ which is ordered according to $\pi$. Then $\lceil \log |G| \rceil \ge D^{\mathrm{A}\to\mathrm{B}}(f)$.*

*(2) Let $G$ be a nondeterministic OBDD for $g$ which is ordered according to $\pi$ and which uses at most $r$ nondeterministic guesses. Then $\lceil \log |G| \rceil \ge N_r^{\mathrm{A}\to\mathrm{B}}(f)$.*

For the sake of completeness, we repeat the easy proof of these facts.

**Proof:** Since $f\colon U \times V \to \{0,1\}$ is reducible to $g\colon \{0,1\}^p \times \{0,1\}^{n-p} \to \{0,1\}$ by assumption, it follows that $D^{\mathrm{A}\to\mathrm{B}}(g) \ge D^{\mathrm{A}\to\mathrm{B}}(f)$ and $N_r^{\mathrm{A}\to\mathrm{B}}(g) \ge N_r^{\mathrm{A}\to\mathrm{B}}(f)$ (with respect to the chosen partitions of the inputs).

It remains to prove an upper bound on the one-way communication complexity of $g$ in terms of the OBDD size for $g$. This is done by explicitly constructing a one-way protocol (deterministic or nondeterministic) for $g$ from a given OBDD $G$ for $g$ in the obvious way. Alice follows the path starting at the source of $G$ which is determined by her part of the input (and her advice bits) and sends the number of the node reached to Bob, who in turn follows the path from this node to one of the sinks determined by his input (and his advice bits). The output of the protocol is the value at the reached sink. Obviously $\lceil \log |G| \rceil$ bits are sufficient to encode the numbers of the nodes sent by Alice to Bob. □

## 3 The Influence of the Amount of Nondeterminism on Communication Complexity and the Size of OBDDs

The main result of this paper deals with the question of how the size of partitioned BDDs varies dependent on the amount of available nondeterminism. In this section we discuss the same question for complexity measures closely related to the size of partitioned BDDs, namely (one-way) communication complexity and the size of OBDDs.

There are several concrete functions which are known to be difficult for nondeterministic communication protocols in the case of unlimited nondeterminism ([7], [14]). The respective lower bounds are obtained by standard techniques. Here we are interested in tradeoffs between the number of advice bits and the nondeterministic communication complexity. The following fact is due to Hromkovič and Schnitger [9] and is easily proven by direct simulations.

**Proposition 1.** *Let $f\colon X \times Y \to \{0,1\}$ be an arbitrary function and let $r \ge 0$ be an integer. Then*

$$D(f)/2^r + r \le N_r(f) \le 2 \cdot N_{r+1}(f) - r - 2.$$

The same holds for one-way protocols and the respective complexity measures. The above bounds are tight, e. g., for the well-known *string nonequality function* which checks whether two $n$-bit strings are not equal (see [9]) and with respect to one-way complexity also for the function $\mathrm{INDEX}_n$ from the last section.

The second inequality of Proposition 1 shows that increasing the number of available advice bits by 1 can at most halve the nondeterministic communication complexity. There are no "jumps" in the nondeterministic communication complexity if the number of advice bits is varied.

Since it holds for all $f\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ that $D(f) \le n+1$, Proposition 1 only yields interesting lower bounds if $r = r(n) = O(\log n)$. It is much harder to prove tradeoffs between $N_r(f)$ and the number of advice bits $r(n)$ if $r(n) = \omega(\log n)$. A result of this type has been established by Hromkovič and Schnitger [9]. They have shown that for every number of advice bits $r(n) = O(\log^c n)$, $c \ge 1$ an arbitrary constant, there is a function $f_{r(n)}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ which has nondeterministic communication complexity $O(\log^c n)$ if at least $r(n)$ advice bits are available; but $\Omega(n/\log n)$, if only $o(r(n)/\log n)$ advice bits may be used. For the special case of one-way communication, an asymptotically optimal tradeoff result has been proven in [8].

Now we consider the size of OBDDs as a complexity measure. Bollig and Wegener [3] have observed that for partitioned BDDs where the variable ordering is the same for all parts, which are obviously special nondeterministic OBDDs, the size may only increase polynomially if the number of parts is decreased by a constant number. A similar assertion also holds for general nondeterministic OBDDs. Let $\mathrm{NOBDD}_r(f)$ denote the minimal size of a nondeterministic OBDD which represents $f$ and uses at most $r$ nondeterministic guesses.

**Proposition 2.** *Let $f$ be an arbitrary Boolean function, and let $r \ge 0$ be an integer. Then*

$$\mathrm{NOBDD}_r(f) \le \mathrm{NOBDD}_{r+1}(f)^2.$$

**Proof:** Let $G$ be a nondeterministic OBDD for $f$ with at most $r+1$ nondeterministic nodes on each path from the source to a sink. W. l. o. g., assume that each nondeterministic node in $G$ has only two outgoing edges. Then the size of the nondeterministic OBDD $G$ is equal to the total number of its nodes. Our goal is to eliminate the last nondeterministic node on each path from the source to a sink.

Let $v_1, \ldots, v_k$ be the last nondeterministic nodes on the paths from the source to a sink in $G$. The two successors of these nodes are roots of deterministic sub-OBDDs of $G$. For $i = 1, \ldots, k$, we apply the well-known OBDD synthesis algorithm to compute an OBDD $G_i$ representing the disjunction of the functions represented at the successors of $v_i$. We replace $v_i$ by the source of $G_i$. The new OBDD $G'$ obtained in this way represents the same function as $G$ and has at most $r$ nondeterministic nodes on each path from the source to the sinks. The number of nodes in $G'$ can be bounded by the size of the product graph $G \times G$, i. e., $|G'| \le |G|^2$. □

This bound is essentially tight, as the following function (introduced in [8]) shows. First, let $\mathrm{UINDEX}_n\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be defined as the variant of the function $\mathrm{INDEX}_n$ from the last section (Definition 6) where a unary encoding for the "address" or "pointer" is used instead of a binary one. We consider the function $\mathrm{MUINDEX}_n$ ("masked unary index") with input size $N = 6n$ defined on $s = (s_1, \ldots, s_{2n})$, $t = (t_1, \ldots, t_{2n})$, and $v = (v_1, \ldots, v_{2n})$ as follows. If the vectors $s$ or $t$ do not contain exactly $n$ ones, or there is an index $i$ such that $s_i = t_i = 1$, let $\mathrm{MUINDEX}_n(s,t,v) := 0$. Otherwise, let $i_1 < \cdots < i_n$ and $j_1 < \cdots < j_n$ be the positions of ones in $s$ and $t$, resp., and define $\mathrm{MUINDEX}_n(s,t,v) := \mathrm{UINDEX}_n\left((v_{i_1},\ldots,v_{i_n}),(v_{j_1},\ldots,v_{j_n})\right)$.

**Theorem 2.** *Let $r \geq 0$ be an integer. Then*

$$2^{\lfloor n/2^r \rfloor + r - 1} \leq \mathrm{NOBDD}_r(\mathrm{MUINDEX}_n) = O\bigl(n^2 \cdot 2^{n/2^r + r}\bigr), \quad \textit{hence}$$
$$\mathrm{NOBDD}_r(\mathrm{MUINDEX}_n) = \Omega\bigl(n^{-4} \cdot \mathrm{NOBDD}_{r+1}(\mathrm{MUINDEX}_n)^2\bigr).$$

**Proof:** *Upper bound:* The essence of the construction is as follows. We divide the $v$-variables which play the role of the $x$-variables for $\mathrm{UINDEX}_n$ into $2^r$ blocks of size at most $\lceil n/2^r \rceil$. Using the available amount of nondeterminism, we then guess the block within which the output bit addressed by the $y$-variables for $\mathrm{UINDEX}_n$ lies, and afterwards verify whether this guess has been correct.

We construct a nondeterministic OBDD for $\mathrm{MUINDEX}_n$ where the variables are ordered according to $s_1, t_1, v_1, \ldots, s_{2n}, t_{2n}, v_{2n}$. The OBDD starts with a nondeterministic node with $2^r$ outgoing edges at the top. The $i$th edge leads to a deterministic sub-OBDD which evaluates $\mathrm{MUINDEX}_n$ under the assumption that the output bit lies in the $i$th block of the $x$-variables for $\mathrm{UINDEX}_n$.

During the evaluation, we count the number of ones in the $s$-vector and additionally make sure that $s_i \oplus t_i = 1$ for all $i = 1, \ldots, 2n$. If after some test it is known that this condition is violated or that the number of ones in the $s$-vector is not $n$, then the respective edge is directed to the $0$-sink. Using the number of ones in $s_1, \ldots, s_{i-1}$ and the values of $s_i$ and $t_i$, we can find out which $x$- or $y$-variable is identified with $v_i$ for the evaluation of $\mathrm{UINDEX}_n$. For storing the number of ones, it is obviously sufficient to increase the number of nodes on each level of the OBDD by a factor of $n+1$. It remains to evaluate the subfunction of $\mathrm{UINDEX}_n$ where the output bit lies in the $i$th block of the $x$-variables with respect to the variable ordering on the $x$- and $y$-variables determined by the bit vectors $s$ and $t$. It is easy to see that this can be done using $O(n \cdot 2^{n/2^r})$ nodes.

The size required for each of the deterministic sub-OBDDs is $O(n^2 \cdot 2^{n/2^r})$, and the overall size of the nondeterministic OBDD is $O(n^2 \cdot 2^{n/2^r} \cdot 2^r)$.

*Lower bound:* Let $G$ be a nondeterministic OBDD for $f$ with variable ordering $\pi$ and at most $r$ nondeterministic guesses. Let $p$ be the least index such that the first $p$ variables according to $\pi$ contain exactly $n$ $v$-variables. Fix the $s$-variables such that exactly these variables are selected as memory variables for $\mathrm{UINDEX}_n$. Furthermore, the last $6n - p + 1$ variables according to $\pi$ also contain $n$ $v$-variables, which we select as address variables for $\mathrm{UINDEX}_n$ by fixing the $t$-variables appropriately.

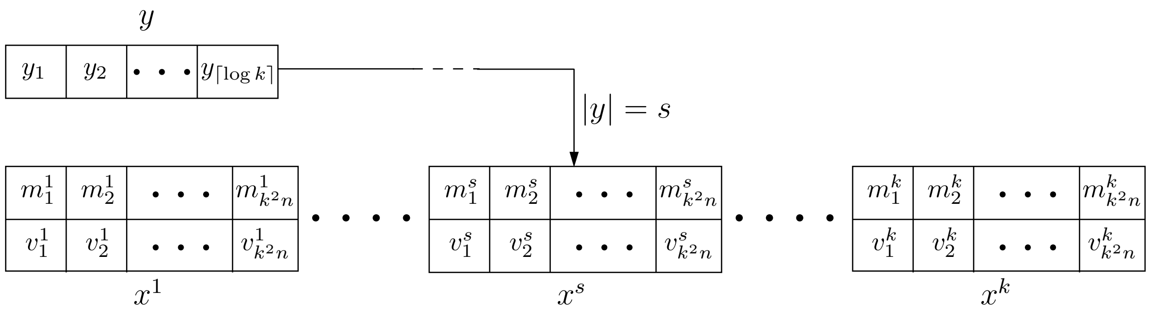
In this way, we obtain a rectangular reduction from $\mathrm{UINDEX}_n$ to $\mathrm{MUINDEX}_n$. By the reduction technique (Lemma 2), Proposition 1 and the known lower bound $D^{\mathrm{A}\to\mathrm{B}}(\mathrm{INDEX}_n) \geq n$ it follows that $\lceil \log |G| \rceil \geq \lfloor n/2^r \rfloor + r$ and hence the claimed lower bound. $\square$

To summarize the facts presented in this section, communication complexity as well as the size of OBDDs decrease only moderately if the number of advice bits or nondeterministic guesses, resp., is increased by a constant. This also holds for partitioned BDDs with the same variable ordering for all parts, since they are special nondeterministic OBDDs, but not for partitioned BDDs with arbitrary variable orderings in the different parts.

## 4 The New Hierarchy Result

In this section we state and prove the main result of the paper, the improved hierarchy for partitioned BDDs. We consider the following function.

**Definition 8.** Let $k$ and $n$ be integers with $k, n \geq 2$, and let $N = N(k,n) = 2k^3n + \lceil \log k \rceil$. We define the function $f_{k,n} \colon \{0,1\}^N \to \{0,1\}$ on blocks of variables $x^1, \dots, x^k$ of size $2k^2n$ each and a vector $y = (y_1, \dots, y_{\lceil \log k \rceil})$ (the "selection pointer"). Let $x^i = (m^i, v^i)$ where $m^i = (m^i_1, \dots, m^i_{k^2n})$ (the "bit mask") and $v^i = (v^i_1, \dots, v^i_{k^2n})$ (the "variable pool") for $i = 1, \dots, k$. (See the following diagram.)


$y$
$y_1$
$y_2$
$y_{\lceil \log k \rceil}$
$|y| = s$
$m^1_1$ $m^1_2$ $m^1_{k^2n}$
$v^1_1$ $v^1_2$ $v^1_{k^2n}$
$m^s_1$ $m^s_2$ $m^s_{k^2n}$
$v^s_1$ $v^s_2$ $v^s_{k^2n}$
$m^k_1$ $m^k_2$ $m^k_{k^2n}$
$v^k_1$ $v^k_2$ $v^k_{k^2n}$
$x^1$
$x^s$
$x^k$

Consider an input assignment for $f_{k,n}$ consisting of the vectors $x^1, \dots, x^k, y$ as described above. Let $s := |y|$ be the value of $y$ interpreted as an encoding of a number from $\{1, \dots, k\}$. We require that $m^s$ contains exactly $\lceil \log n \rceil$ ones, and that each $m^i$ with $i \neq s$ contains exactly $n$ ones. If this is not fulfilled, we define $f_{k,n}(x^1, \dots, x^k, y) := 0$. Otherwise, let $j_{s,1} < \cdots < j_{s,\lceil \log n \rceil}$ and $j_{i,1} < \cdots < j_{i,n}$, $i \neq s$, be the indices of ones in the vectors $m^s$ and $m^i$, $i \neq s$, resp. Define

$$f_{k,n}\left(x^1, \dots, x^k, y\right) := \bigoplus_{1 \leq i \leq k,\, i \neq s} \mathrm{INDEX}_n\left((v^i_{j_{i,1}}, \dots, v^i_{j_{i,n}}), (v^s_{j_{s,1}}, \dots, v^s_{j_{s,\lceil \log n \rceil}})\right).$$

The following theorem contains the main result.

**Theorem 3 (Improved Partitioned BDD Hierarchy).** *Let $k$ and $n$ be integers with $k, n \geq 2$.*

*(1) The function $f_{k,n}$ can be represented by $k$-partitioned BDDs of size $O(k^4n^3)$;*

*(2) every $(k-1)$-partitioned BDD for $f_{k,n}$ has size $2^{\Omega(n/k)}$.*

**Corollary 1.** *Let $\varepsilon > 0$ be an arbitrary constant. There is a sequence of functions $F_N \colon \{0,1\}^N \to \{0,1\}$ and a sequence of integers $K = K(N)$ with $K = \Theta\left((N/\log^{1+\varepsilon} N)^{1/4}\right)$, both defined for infinitely many $N$, such that, for $N$ large enough,*

*(1) the function $F_N$ can be represented by $K$-partitioned BDDs of size polynomial in $N$; and*

*(2) every $(K-1)$-partitioned BDD for $F_N$ has at least size $2^{\Omega(\log^{1+\varepsilon} N)} = N^{\Omega(\log^{\varepsilon} N)}$.*

**Proof:** For arbitrary $n \geq 2$, let $k(n) := \lceil n/\log^{1+\varepsilon} n \rceil$. Define $N$ as a function of $n$ by $N(n) := 2k(n)^3n + \lceil \log k(n) \rceil$. By this definition, $N$ and $n$ are strictly increasing functions of each other. Define $F_{N(n)} := f_{k(n),n}$ and $K(N(n)) := k(n)$. Using that, for $N(n)$ (and thus $n$) large enough,

$$2n^4/\log^{3(1+\varepsilon)} n \leq N(n) \leq 3n^4/\log^{3(1+\varepsilon)} n,$$

one easily verifies that $K(N(n)) = \Theta\left((N(n)/\log^{1+\varepsilon} N(n))^{1/4}\right)$.

Since $n \le N(n)/(2k(n)^3)$, we have $k(n)^4 n^3 \le N(n)^3/8k(n)^5 \le N(n)^3$. Hence, Part (1) follows from the upper bound from Theorem 3. On the other hand, $n \ge N(n)/(3k(n)^3)$, which gives $n/k(n) \ge N/(3k(n)^4) = \Omega\left(\log^{1+\varepsilon} N(n)\right)$ (for $N(n) \to \infty$). Substituting this into the lower bound from Theorem 3, Part (2) follows. □

In the remainder of the section, we prove Theorem 3.

**Proof of Theorem 3(1)—The Upper Bound:** We describe the construction of the OBDD for the $s$th part of the partitioned BDD, where $s \in \{1, \ldots, k\}$. The $s$th part is responsible for the evaluation of $f_{k,n}$ in the case $|y| = s$. The variable ordering starts with $y, x^s$, then all blocks $x^i$ with $i \ne s$ follow. The ordering within a block $x^i$, $1 \le i \le k$, is $m_1^i, v_1^i, \ \ldots, \ m_{k^2 n}^i, v_{k^2 n}^i$.

We first evaluate $|y|$ by a binary tree on the $y$-variables. If $|y| \ne s$, the $0$-sink is reached. The main part of the graph is reached only if $|y| = s$. In this case, we want to evaluate

$$\bigoplus_{1 \le i \le k,\, i \ne s} \mathrm{INDEX}_n \left( (v^i_{j_{i,1}}, \ldots, v^i_{j_{i,n}}), (v^s_{j_{s,1}}, \ldots, v^s_{j_{s,\lceil \log n \rceil}}) \right),$$

if $j_{s,1} < \cdots < j_{s,\lceil \log n \rceil}$ and $j_{i,1} < \cdots < j_{i,n}$, $i \ne s$, are the positions of $\lceil \log n \rceil$ and $n$ ones, resp., in the bit masks as in Definition 8, and the function is zero if a bit mask contains the "wrong" number of ones.

We first read the variables in $x^s = (m^s, v^s)$ in the prescribed order and compute the binary number represented by $(v^s_{j_{s,1}}, \ldots, v^s_{j_{s,\lceil \log n \rceil}})$ which serves as the address for all index functions. We store the number of ones in $m^s$ already seen (which is at most $\lceil \log n \rceil$, or we know that the function is zero) and the partial address already computed. For this, we need only $O(\log n \cdot n)$ nodes per level of the OBDD and $O(k^2 n \cdot \log n \cdot n)$ nodes altogether. At the bottom of this part, we know the index $a \in \{1, \ldots, n\}$ of the addressed bit for the $k-1$ index functions.

We then evaluate the memory contents for the index functions encoded in the vectors $v^i$ with $i \ne s$. This is done analogously to the construction for $v^s$ above. While we read the variables of a block with number $i \ne s$, we count the number of ones in the $m^i$-vector (which is at most $n$, or the function is zero) and store the parity of the addressed bits found so far ($v^i_{j_{i,a}}$ for the current block), i. e., a single bit. Thus, we need only $O(k^2 n \cdot n \cdot 1)$ nodes for the evaluation of a single block, and $O(k^3 n^2)$ nodes altogether.

The $s$th part constructed above has total size $O(k^3 n^3)$. Since we have $k$ parts, the complete partitioned BDD has the claimed size. □

Now we turn to the proof of the lower bound. The simple technique for proving lower bounds on the size of OBDDs, i. e., reducing communication complexity to the size, does not seem to apply for partitioned BDDs with arbitrary variable orderings for the different parts. Furthermore, we have seen in the last section that communication complexity does not show the strong dependence on the available amount of nondeterminism which we claim for partitioned BDDs. Hence, a different approach is required.

The definition of the function $f_{k,n}$ still captures the essence of the more complicated construction of Bollig and Wegener [3]. We know that the function $\mathrm{INDEX}_n$ is easy to evaluate if the "right" variable ordering, where the address variables come before the memory variables, is chosen. On the other hand, the function is hard if the memory variables are tested first. Now the function $f_{k,n}$ is constructed in such a way that an arbitrary block $x^s$, $s = 1, \ldots, k$, called the *address block* in the following, may be chosen to supply the address variables for $k-1$ copies of $\mathrm{INDEX}_n$, and each of the other $k-1$ blocks serves as the memory contents for one of these copies of $\mathrm{INDEX}_n$.

In a $k$-partitioned BDD, we may have a part with the "right" variable ordering for each of the $k$ choices of the address block. However if only $k-1$ variable orderings are available, at least one choice of the address block is "bad," i. e., at least some of the memory vectors have to be evaluated before the address in the partitioned BDD. Thus, the partitioned BDD computes at least one copy of $\mathrm{INDEX}_n$ according to the "wrong" variable ordering. It remains to show that at least one part of the partitioned BDD, which is an OBDD, has to evaluate many inputs of this copy of $\mathrm{INDEX}_n$ correctly. Then we can apply our knowledge on the deterministic one-way complexity of $\mathrm{INDEX}_n$ to show that this OBDD has large size.

We now formalize these ideas. The proof of the lower bound consists of two parts, a simplification of the structure of the partitioned BDD by combinatorial tools and the final application of the lower bound on communication complexity. The key observations for the combinatorial part are captured within the following two lemmas which have been extracted from the paper [3] of Bollig and Wegener.

**Lemma 3 (Bollig and Wegener, 1997).** *Let $V := B_1 \cup \cdots \cup B_k$ with $B_i := \{v_1^i, \ldots, v_m^i\}$, $m = k^2 n$ and $i = 1, \ldots, k$. Let $\pi_1, \ldots, \pi_{k-1}$ be arbitrary orderings of the variables from $V$. Let $L_i$ be the set of the first $m$ variables according to $\pi_i$ for $i = 1, \ldots, k-1$, and let $L := L_1 \cup \cdots \cup L_{k-1}$, $R := V \setminus L$. Then there are numbers $b_0, b_1, \ldots, b_{k-1} \in \{1, \ldots, k\}$ (where $b_1, \ldots, b_{k-1}$ are not necessarily different) such that the following holds.*

*(1) $b_0 \notin \{b_1, \ldots, b_{k-1}\}$ and $|B_{b_0} \cap R| \geq kn$;*

*(2) $|B_{b_i} \cap L_i| \geq n$ for all $i = 1, \ldots, k-1$.*

**Proof:** We have $|V| = km$. Since $|L| = |L_1 \cup \cdots \cup L_{k-1}| \leq (k-1)m$, it follows that $|R| \geq m$. By the pigeonhole principle, there is a $b_0 \in \{1, \ldots, k\}$ such that $|B_{b_0} \cap R| \geq m/k = kn$. On the other hand, it follows for arbitrary $i \in \{1, \ldots, k-1\}$ that $\left|\bigcup_{j \neq b_0} B_j \cap L_i\right| \geq m - (m - m/k) = m/k$. Again by the pigeonhole principle, there is a $b_i \in \{1, \ldots, k\}$ such that $|B_{b_i} \cap L_i| \geq m/k^2 = n$. □

**Lemma 4 (Bollig and Wegener, 1997).** *Let $X_1, \ldots, X_k$ be finite sets. Let $f\colon X_1 \times \cdots \times X_k \to \{0,1\}$ be defined on variables $x_1, \ldots, x_k$, where $x_i \in X_i$ for $i = 1, \ldots, k$. Let $|f^{-1}(1)| \geq \alpha |X_1||X_2| * \cdots * |X_k|$, $\alpha > 0$. Then there are assignments $a_2, \ldots, a_k$ to $x_2, \ldots, x_k$ such that $\left|\left(f|_{x_2=a_2,\ldots,x_k=a_k}\right)^{-1}(1)\right| \geq \alpha |X_1|$.*

**Proof:** We obviously have $|(f^{-1})(1)| = \sum_{a_2 \in X_2,\, \ldots,\, a_k \in X_k} \left|\left(f|_{x_2=a_2,\ldots,x_k=a_k}\right)^{-1}(1)\right|$. Since the sum contains $|X_2| * \cdots * |X_k|$ terms altogether, the claim follows by the pigeonhole principle. □

Now we are ready to complete the proof of the hierarchy result.

**Proof of Theorem 3(2)—The Lower Bound:** Let $G$ be an arbitrary $(k-1)$-partitioned BDD for $f_{k,n}$ with the orderings $(\pi_1, \ldots, \pi_{k-1})$. In the combinatorial part of the proof, we set several variables to constants. By applying these assignments to $f_{k,n}$ as well as to the $(k-1)$-partitioned BDD $G$, we obtain a $(k-1)$-partitioned BDD which represents a subfunction of $f_{k,n}$. First, we choose the variable block which will contain the address for the index functions. Then we isolate an OBDD-part of the remaining partitioned BDD where these address variables are tested after the memory variables for one of the copies of $\mathrm{INDEX}_n$ and which computes many 1-inputs of the function.

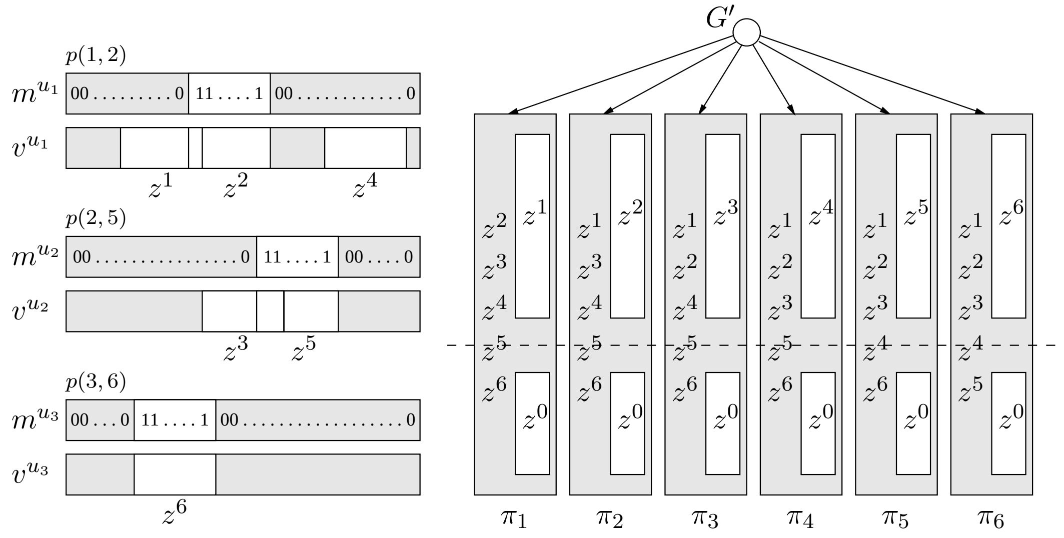


Figure 1: Variables in $G'$ (symbolic).

We start by applying Lemma 3 to the suborderings $\pi'_1, \ldots, \pi'_{k-1}$ on $V = \{v^i_1, \ldots, v^i_{k^2n} \mid i = 1, \ldots, k\}$. As before, we call the vectors $x^i = \big((m^i_1, v^i_1), \ldots, (m^i_{k^2n}, v^i_{k^2n})\big)$ *variable blocks*. Assuming the notation of the lemma, we have block numbers $b_0, b_1, \ldots, b_k$ such that

(1) $b_0 \notin \{b_1, \ldots, b_{k-1}\}$ and $|B_{b_0} \cap R| \geq kn$;

(2) $|B_{b_i} \cap L_i| \geq n$ for all $i = 1, \ldots, k-1$.

Let $u_1, \ldots, u_r$, $r \leq k-1$, be the different (unique) block numbers among $b_1, \ldots, b_{k-1}$.

We fix the $y$-variables such that $|y| = b_0$. We then fix the bit mask $m^{b_0}$ such that exactly $\lceil \log n \rceil$ variables from $B_{b_0} \cap R$ are selected. Let $z^0$ be the vector of these variables. Furthermore, fix all variables in blocks $x^i$ with $i \notin \{b_0, u_1, \ldots, u_r\}$. We select $n$ arbitrary variables from the respective $v$-vectors by an appropriate setting of the bit masks, and we set all $v$-variables in these blocks to $0$.

By applying these assignments of constants to $G$, we obtain a $(k-1)$-partitioned BDD $G'$ which represents a subfunction $f'$ of $f_{k,n}$ which only depends on the variables $m^{u_1}, v^{u_1}, \ldots, m^{u_r}, v^{u_r}$ and $z^0$. We do not fix the remaining variables yet, but we consider only a small set of possible assignments to them which we define now.

For each $i = 1, \ldots, k-1$, choose $n$ variables from $B_{b_i} \cap L_i$ and define $z^i$ as the vector of these variables (notice that $z^p$ and $z^q$ need not be disjoint for $p \neq q$). For $i = 1, \ldots, r$, define $C_i := \{j \mid b_j = u_i\}$ (the parts of $G$ which contain a $z$-vector belonging to the variable block with number $u_i$). For each $j \in C_i$, define $p(i,j)$ as the assignment to the bit mask $m^{u_i}$ by which the variables in the vector $z^j$ are selected. (See the example in Figure 1.)

For each variable block with number $u_1, \ldots, u_r$, we have the possibility of choosing one of the $z$-vectors which lies in this block by using an appropriate bit mask. For $i = 1, \ldots, r$, let $c_i \in C_i$ be the number of the chosen $z$-vector.

We consider the subfunction $f'_{c_1,\ldots,c_r}$ of $f'$ which is obtained by fixing the bit mask $m^{u_i}$ according to $p(i, c_i)$, for $i = 1, \ldots, r$. By the definitions, this function only depends on

$z^0, z^{c_1}, \ldots, z^{c_r}$ and

$$f'_{c_1,\ldots,c_r}(z^{c_1}, \ldots, z^{c_r}, z^0) = \bigoplus_{1\le i\le r} \mathrm{INDEX}_n\left(z^{c_i}, z^0\right).$$

Furthermore, the sets of variables in the vectors $z^0, z^{c_1}, \ldots, z^{c_r}$ are pairwise disjoint.

It is easy to verify that $|(f'_{c_1,\ldots,c_r})^{-1}(1)| = \frac{1}{2} \cdot n \cdot 2^{rn}$. Hence, also $|(f')^{-1}(1)| \ge \frac{1}{2} \cdot n \cdot 2^{rn}$. Since $G'$ consists of $k-1$ parts, it follows (again) by the pigeonhole principle that there is an OBDD-part, w. l. o. g. the first one which we call $G_1$, such that for the function $g_1$ represented by this part

$$|(g_1)^{-1}(1)| \ge \frac{1}{2k} \cdot n \cdot 2^{rn}.$$

W. l. o. g., the $z$-vector in the upper part of $G_1$ is $z^1$ and belongs to the variables in $v^{u_1}$. Set $c_1 := 1$ and choose $c_i \in C_i$ arbitrarily for $i = 2, \ldots, r$. Consider the subfunction $f'_{c_1,\ldots,c_r}$. Since the variable vectors $z^{c_1}, z^{c_2}, \ldots, z^{c_r}$ are disjoint, Lemma 4 can be applied to $f'_{c_1,\ldots,c_r}$. We obtain assignments $a^2, \ldots, a^r$ to the variable vectors $z^{c_2}, \ldots, z^{c_r}$ such that the resulting subfunction $f''$ of $f'_{c_1,\ldots,c_r}$ fulfills

$$|(f'')^{-1}(1)| \ge \frac{1}{2k} \cdot n \cdot 2^n$$

and still depends on all variables from $z^0$ and $z^{c_1}$. This subfunction is

$$\begin{aligned} f''(z^{c_1}, z^0) &= \mathrm{INDEX}_n(z^{c_1}, z^0) \oplus \mathrm{INDEX}_n(a^2, z^0) \oplus \cdots \oplus \mathrm{INDEX}_n(a^r, z^0) \\ &= \mathrm{INDEX}_n(z^{c_1}, z^0) \oplus h(z^0), \end{aligned}$$

where $h(z^0) = 1$ for $|z^0| \in I \subseteq \{0, \ldots, n-1\}$, $I$ appropriately defined, and $h(z^0) = 0$, otherwise.

By applying the assignments $p(1, c_1), p(2, c_2), \ldots, p(r, c_r)$ and $a^2, \ldots, a^r$ to $G_1$, we obtain a deterministic OBDD which represents $f''$ and only depends on the variable vectors $z^0$ and $z^{c_1}$. Furthermore, all $z^{c_1}$-variables are tested before the $z^0$-variables in the respective variable ordering. By Lemma 1 and the reduction technique (Lemma 2), the claimed lower bound follows. $\square$

## Acknowledgments

I would like to thank Ingo Wegener for checking the proofs of the draft versions of this paper and for discussion of the results. Further thanks to the referees for several helpful hints.

## References

[1] F. Ablayev. Randomization and nondeterminism are incomparable for polynomial ordered binary decision diagrams. In *Proc. of the 24th Internat. Coll. on Automata, Languages, and Programming (ICALP), LNCS 1256*, 195–202. Springer, Berlin, 1997.

[2] P. Beame, M. Saks, and J. S. Thathachar. Time-space tradeoffs for branching programs. In *Proc. of the 39th IEEE Symp. on Foundations of Computer Science (FOCS)*, 254–263, 1998.

[3] B. Bollig and I. Wegener. Complexity theoretical results on partitioned (nondeterministic) binary decision diagrams. *Theory of Computing Systems*, 32:487–503, 1999. Earlier version in *Proc. of 22nd MFCS, LNCS 1295*, 159–168. Springer, 1997.

[4] A. Borodin, A. A. Razborov, and R. Smolensky. On lower bounds for read-$k$-times branching programs. *Computational Complexity*, 3:1–18, 1993.

[5] R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Trans. Computers*, C-35(8):677–691, Aug. 1986.

[6] R. E. Bryant. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, Sept. 1992.

[7] J. Hromkovič. *Communication Complexity and Parallel Computing*. EATCS Texts in Theoretical Computer Science. Springer, Berlin, 1997.

[8] J. Hromkovič and M. Sauerhoff. Tradeoffs between nondeterminism and complexity for communication protocols and branching programs. In *Proc. of the 17th Ann. Symp. on Theoretical Aspects of Computer Science (STACS), LNCS 1770*, 145–156. Springer, Berlin, 1999.

[9] J. Hromkovič and G. Schnitger. Nondeterministic communication with a limited number of advice bits. In *Proc. of the 28th Ann. ACM Symp. on Theory of Computing (STOC)*, 551 – 560, 1996.

[10] J. Jain, J. Bitner, J. A. Abraham, and D. S. Fussell. Functional partitioning for verification and related problems. In T. Knight and J. Savage, editors, *Advanced Research in VLSI and Parallel Systems: Proceedings of the 1992 Brown/MIT Conference*, 210–226, 1992.

[11] S. P. Jukna. Entropy of contact circuits and lower bounds on their complexity. *Theoretical Computer Science*, 57:113 – 129, 1988.

[12] M. Krause. Lower bounds for depth-restricted branching programs. *Information and Computation*, 91(1):1–14, Mar. 1991.

[13] I. Kremer, N. Nisan, and D. Ron. On randomized one-round communication complexity. In *Proc. of the 27th Ann. ACM Symp. on Theory of Computing (STOC)*, 596 – 605, 1995.

[14] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, Cambridge, 1997.

[15] A. A. Razborov. Lower bounds for deterministic and nondeterministic branching programs. In *Proc. of Fundamentals of Computation Theory (FCT), LNCS 529*, 47–60. Springer, Berlin, 1991.

[16] M. Sauerhoff. Lower bounds for randomized read-$k$-times branching programs. In *Proc. of the 15th Ann. Symp. on Theoretical Aspects of Computer Science (STACS), LNCS 1373,* 105 – 115. Springer, Berlin, 1998.

[17] M. Sauerhoff. On the size of randomized OBDDs and read-once branching programs for $k$-stable functions. In *Proc. of the 16th Ann. Symp. on Theoretical Aspects of Computer Science (STACS), LNCS 1563*, 488–499. Springer, Berlin, 1999.

[18] J. Thathachar. On separating the read-$k$-times branching program hierarchy. In *Proc. of the 30th Ann. ACM Symp. on Theory of Computing (STOC)*, 653–662, 1998.

[19] I. Wegener. *The Complexity of Boolean Functions*. Wiley-Teubner, 1987.

[20] I. Wegener. *Branching Programs and Binary Decision Diagrams—Theory and Applications*. Monographs on Discrete and Applied Mathematics. SIAM, Philadelphia, PA, 2000.