A Lower Bound Technique for Restricted Branching Programs and Applications

Philipp Woelfel[⋆]

FB Informatik, LS2, Univ. Dortmund, 44221 Dortmund, Germany
`woelfel@Ls2.cs.uni-dortmund.de`

Abstract. We present a new lower bound technique for two types of restricted Branching Programs (BPs), namely for read-once BPs (BP1s) with restricted amount of nondeterminism and for $(1,+k)$-BPs. For this technique, we introduce the notion of *(strictly) k-wise l-mixed* Boolean functions, which generalizes the concept of l-mixedness defined by Jukna in 1988 [3]. We prove that if a Boolean function $f \in B_n$ is (strictly) k-wise l-mixed, then any nondeterministic BP1 with at most $k-1$ nondeterministic nodes and any $(1,+k)$-BP representing f has a size of at least $2^{\Omega(l)}$. While leading to new exponential lower bounds of well-studied functions (e.g. linear codes), the lower bound technique also shows that the polynomial size hierarchy for BP1s with respect to the available amount of nondeterminism is strict. More precisely, we present a class of functions $g_n^k \in B_n$ which can be represented by polynomial size BP1s with k nondeterministic nodes, but require superpolynomial size if only $k-1$ nondeterministic nodes are available (for $k = o(n^{1/3}/\log^{2/3} n)$). This is the first hierarchy result of this kind where the BP1 does not obey any further restrictions. We also obtain a hierarchy result with respect to k for $(1,+k)$-BPs as long as $k = o(\sqrt{n/\log n})$. This extends the hierarchy result of Savický and Žák [9], where k was bounded above by $\frac{1}{2}\, n^{1/6}/\log^{1/3} n$.

⋆ Supported in part by DFG grant We 1066/10-1.

ISSN 1433-8092

1 Introduction and Results

Branching Programs (BPs) or equivalently Binary Decision Diagrams (BDDs) belong to the most important nonuniform models of computation. Deterministic and nondeterministic BPs can be simulated by the corresponding Turing machines, and the BP complexity of a Boolean function is a measure for the space complexity of the corresponding model of sequential computation. Therefore, one is interested in large lower bounds for BPs.

Definition 1. A (deterministic) *Branching Program* (short: *BP*) on the variable set $X_n = \{x_1, \ldots, x_n\}$ is a directed acyclic graph with one source and two sinks. The internal nodes are marked with variables in X_n and the sinks are labeled with the Boolean constants 0 and 1. Further, each internal node has two outgoing edges, marked with 0 and 1, respectively. A *nondeterministic* (short: *n.d.*) Branching Program is a BP with some additional unmarked nodes with out-degree two, called *nondeterministic nodes*. The *size* of a (possibly n.d.) Branching Program G is the number of its nodes, and ist denoted by $|G|$.

Let G be a (possibly n.d.) BP on X_n and $a = (a_1, \ldots, a_n) \in \{0,1\}^n$ an assignment to the variables in X_n. A source-to-sink path in G is called *computation path* of a, if it leaves any node marked with x_i over the edge labeled with a_i. Note that an input may have multiple computation paths if G is nondeterministic.

Let B_n denote the set of Boolean functions $\{0,1\}^n \to \{0,1\}$. The BP G represents the function $f \in B_n$ for which $f(a) = 1$ if and only if there exists a computation path of a leading to the 1-sink.

Until today, no superpolynomial lower bounds for general BPs representing an explicitly defined function are known. Therefore, various types of restricted BPs have been investigated, and one is interested in refining the proof techniques in order to obtain lower bounds for less restricted BPs. For this paper, the following two common types of restricted BPs are most important (for an in-depth discussion of other restricted BP models we refer to [14]).

Definition 2.

(i) A *(n.d.) read-k-times BP* (short: *BPk*) is a (n.d.) BP where each variable appears on each computation path at most k times.

(ii) A *(n.d.)* $(1, +k)$-*BP* is a (n.d.) BP where for each computation path p there exist at most k variables appearing on p more than once.

Especially deterministic BP1s have been studied to a great extent. The first exponential lower bounds date back to the 80s [13, 15], and today, lower bounds for explicitly defined functions in P are as large as $2^{n-O(\log^2 n)}$ [1].

If one considers BPs which allow multiple tests of the same variable during a computation, then one has to distinguish between *syntactic* and *semantic* restrictions. The restrictions given in the definition above are *semantic*, because they have to hold on each computation path. But since graph theoretical paths may be inconsistent, one may obtain BPs with less computational power if the restriction has to hold even on each graph theoretical path. Such restrictions are called *syntactic*, and the BPs corresponding to Definition 2 but with syntactic restrictions are called *syntactic BPks* and *syntactic* $(1,+k)$*-BPs*, respectively. Note that the class of functions computable by polynomial size syntactic $(1,+k)$-BPs is in fact a proper subclass of the class of functions computable by polynomial size $(1,+k)$-BPs (for $k \leq n^{1/2}/(6\log n)$) [11].

Besides the general interest in finding exponential lower bounds for less and less restricted BP-models, much of the research on Branching Programs has focused on separating the power of different types of Branching Programs. Similarly, it has been of considerable interest, how the computational power of BPs is influenced by e.g. the available amount of nondeterminism or the multiplicity of variable tests (i.e. the term k in BPks or $(1,+k)$-BPs).

Results on the influence of the available amount of nondeterminism have so far been obtained only for n.d. BP1s with additional restrictions.

Definition 3.

(i) An $(\vee,k)$-BP1 is a family of k deterministic BP1s and represents the function $f_1 \vee \ldots \vee f_k$, where f_i is the function represented by the ith BP1.

(ii) A BP1 is called *Ordered Binary Decision Diagram* (short: OBDD), if the nodes can be partitioned into levels such that all edges point only from lower to higher levels and all internal nodes of one level are marked with the same variable. A k*-Partitioned Binary Decision Diagram* (short: k-PBDD) is an $(\vee,k)$-BP1 whose BP1s are in fact OBDDs.

Note that we can regard an $(\vee,k)$-BP1 as a n.d. BP1 having a binary tree of exactly $k-1$ n.d. nodes at the top such that the outgoing edges of the leaves lead to the sources of k disjoint BP1s. Hence, the set of functions which can be represented in polynomial size by $(\vee,k)$-BP1s is a subset of the functions which can be represented in polynomial size by BP1s with at most $k-1$ n.d. nodes. Although not yet proven, the results of [5] indicate that $(\vee,k)$-BP1s might be in fact less powerful than n.d. BP1s with $k-1$ nondeterministic nodes.

Bollig and Wegener [2] have proven the first hierarchy result for k-PBDDs with respect to k, which has been extended later by Sauerhoff [6]. He presented functions being representable by polynomial size $(k+1)$-PBDDs but requiring superpolynomial size k-PBDDs if $k = O\big((n/\log^{1+\epsilon} n)^{1/4}\big)$ for arbitrary $\epsilon > 0$. This means that for these k, the polynomial size hierarchy of k-PBDDs with respect to k is strict. A generalization of this result was obtained by Savický and Sieling [7], who proved that the polynomial size hierarchy of $(\vee,k)$-BP1s with respect to k ist strict for $k \leq 2/3\sqrt{\log n}$.

One of the main contributions of this paper is a hierarchy result for n.d. BP1s without any restrictions except on the number of n.d. nodes. We present a class of multipointer functions $g_n^k \in B_n$, which can be represented by polynomial size n.d. BP1s having k n.d. nodes, but require superpolynomial size if at most $k-1$ n.d. nodes are available (for $k = o(n^{1/3}/\log^{2/3} n)$).

The other main contribution of this paper is an improved hierarchy result for $(1,+k)$-BPs with respect to k. For syntactic $(1,+k)$-BPs, the first hierarchy result was obtained by Sieling in 1996 [10] and later improved as well as generalized for the semantic restriction by Savický and Žák [9]. They showed that the polynomial size hierarchy with respect to k is strict for $(1,+k)$-BPs if $k \le \frac{1}{2} n^{1/6}/\log^{1/3} n$ and for syntactic $(1,+k)$-BPs if $k \le \frac{1}{2}\sqrt{n}/\log n$. We extend their result for both types of restrictions by presenting a class of functions for which polynomial size syntactic $(1,+k)$-BPs but no polynomial size (semantic) $(1,+(k-1))$-BPs exist if $k = o(\sqrt{n/\log n})$.

The hierarchy results for n.d. BP1s and $(1,+k)$-BPs are possible because of a new lower bound technique. Interestingly enough, this technique can be equivalently applied for both types of BPs. It mainly consists of introducing the notion of strictly k-wise l-mixed functions, and of proving for such functions a lower bound of $2^l + 1$ for n.d. BP1s with at most $k-1$ n.d. nodes and a lower bound of $2^{l/2}$ for $(1,+k)$-BPs. This will be done in Section 2. In Section 3, we will show how to prove the k-wise l-mixedness of functions. As an easy example, we show that d-rare m-dense functions, investigated e.g. by Jukna and Razborov [4], are in fact k-wise l-mixed for $l < \min\{d, m/k\}$. We obtain as a corollary exponential lower bounds for linear codes in the $(1,+k)$-BP model, which have already been proven in [4], and new exponential lower bounds in the n.d. BP1 model with restricted amount of nondeterminism. We also show how to construct easily from a 1-wise l-mixed function in B_n a k-wise l/k-mixed function in B_{kn}. This construction helps us in Section 4 to obtain the hierarchy result for $(1,+k)$-BPs. The hierarchy result for n.d. BP1s will finally be stated in Section 5.

2 The Lower Bound Technique

2.1 Notation

In the following text, we consider functions defined on the n Boolean variables in $X_n = \{x_1, \dots, x_n\}$. A *partial input* is an element $\alpha = (\alpha^1, \dots, \alpha^n) \in \{0,1,*\}^n$. While a position α^i with value 0 or 1 means that the input variable x_i is fixed to the corresponding constant, a value of $*$ means that the input variable remains free. If $f \in B_n$ is a Boolean function and α is a partial input, then $f|_\alpha$ means the subfunction of f obtained by restricting all inputs to α. For a partial input $\alpha \in \{0,1,*\}^n$, we denote the *support* of α by $S(\alpha) := \{x_i \mid \alpha^i \ne *\}$. The empty partial input, i.e. the partial input with support $\emptyset$, is written as ε. For two partial inputs α, β with the same support, we let $D(\alpha, \beta)$ be the set which contains all variables $x_i \in X_n$ for which $\alpha^i \ne \beta^i$. If α and β are partial inputs with disjoint supports, then we denote by $\alpha\beta$ the partial input defined by

$$(\alpha\beta)^i = \begin{cases} \alpha^i & \text{if } x_i \in S(\alpha) \\ \beta^i & \text{if } x_i \in S(\beta) \\ * & \text{otherwise.} \end{cases}$$

2.2 The Concept of k-wise l-mixed Functions

Our lower bound technique relies mainly on a generalization of the following property of Boolean functions, which was defined by Jukna in 1988 [3].

Definition 4. Let $l \in \mathbb{N}$. A function $f \in B_n$ is called *l-mixed*, if for all $V \subseteq X_n$ such that $|V| = l$, any two distinct partial inputs α, β with support V yield different subfunctions, i.e. $f|_\alpha \neq f|_\beta$.

Many exponential lower bound proofs for BP1s use the well known fact, that l-mixed functions have a BP1 complexity of $\Omega(2^l)$.

Proposition 1. *The size of any BP1 for an l-mixed function is at least $2^l + 1$.*

In the literature, usually a lower bound of $2^l - 1$ is stated (see e.g. [14]). But since we use this bound later in an induction hypothesis, we give here a slight modified proof.

Proof. In [14] it is shown that each BP1 for an l-mixed function starts with a complete tree of depth $l - 1$, and thus has a size of at least $2^l - 1$. But if one of the nodes of this complete tree is a sink, then the sink can be reached by testing at most $l - 1$ nodes. Thus, there exists a partial input α with support of size $l - 1$ such that $f|_\alpha = c$ for some $c \in \{0, 1\}$. Then for any variable x_i not in the support of α, it holds that $f|_{\alpha,x_i=0} = f|_{\alpha,x_i=1} = c$. This contradicts the l-mixed assumption. Therefore, there is no sink embedded in the complete binary tree described above, and counting both sinks (an l-mixed function is obviously not constant) we get a lower bound of at least $2^l + 1$. □

The following definition generalizes the above definition of l-mixed functions and is fundamental for our lower bound technique.

Definition 5. Let $kl \leq n$. In the following formula, we restrict the choices of $W_1, \ldots, W_k, V_1, \ldots, V_k$ to disjoint subsets of X_n such that $|V_j| = l$ $(1 \leq j \leq k)$. The choices of the partial inputs $\lambda_j, \alpha_j, \beta_j$ are restricted in such a way that $S(\lambda_j) = W_j$ and $S(\alpha_j) = S(\beta_j) = V_j$. Below, we allow for each $1 \leq j \leq k$ the choice of a partial assignment $\gamma_j \in \{\alpha_j, \beta_j\}$. We then denote by γ_j^* the element in $\{\alpha_j, \beta_j\} \setminus \gamma_j$ and let $c = \lambda_1\gamma_1 \ldots \lambda_k\gamma_k$ and c_j be the partial assignment obtained from c by replacing γ_j with γ_j^*.

A Boolean function $f \in B_n$ is called *k-wise l-mixed* if

$$\exists W_1, \lambda_1 \, \forall V_1, \alpha_1 \neq \beta_1 \, \exists \gamma_1 \in \{\alpha_1, \beta_1\} \ \ldots \ \exists W_k, \lambda_k \, \forall V_k, \alpha_k \neq \beta_k \, \exists \gamma_k \in \{\alpha_k, \beta_k\} :$$
$$\sum_{i=1}^{k} |W_i| \leq n - kl \quad \wedge \quad \exists x^* \, \forall 1 \leq j \leq k : \; f|_c(x^*) \; \neq \; f|_{c_j}(x^*),$$

where x^* is an input for the subfunction $f|_c$ (and $f|_{c_j}$), i.e. $S(x^*) = X_n \setminus (W_1 \cup V_1 \cup \ldots \cup W_k \cup V_k)$. If in the above formula we even have $f|_c(x^*) > f|_{c_j}(x^*)$ (instead of "$\neq$"), then f is called *strictly k-wise l-mixed.*

Remark 1. Any strictly k-wise l-mixed function is k-wise l-mixed. Furthermore, any (strictly) $(k+1)$-wise $(l+1)$-mixed function is (strictly) k-wise $(l+1)$-mixed and also (strictly) $(k+1)$-wise l-mixed. Finally, a function f is 1-wise l-mixed if and only if there exists an l-mixed subfunction of f.

2.3 Lower Bounds for BP1s with Restricted Amount of Nondeterminism

The property k-wise l-mixed of a Boolean function implies lower bounds for n.d. BP1s with limited amount of nondeterminism. In order to measure the amount of nondeterminism in BPs, we need an appropriate measurement. Here, we choose the number of n.d. nodes, but another possibility would be to count the maximum number of n.d. nodes on any computation path. The proof of the following theorem though, can be adapted in order to obtain similar results for other measures of nondeterminism.

Theorem 1. *If $f \in B_n$ is a strictly k-wise l-mixed function and G is a n.d. BP1 with at most $k-1$ n.d. nodes representing f, then $|G| \geq 2^l + 1$.*

In order to prove the theorem, we make use of the notion of filters and of the idea behind a lower bound technique of Simon and Szegedy [12] for deterministic BP1s. A *filter* of a set X is a closed upward subset of 2^X (i.e. if $S \in \mathcal{F}$, then all supersets of S are in $\mathcal{F}$). Let $\mathcal{F}$ be a filter of X_n. A subset $B \subseteq X_n$ is said to be in the *boundary* of $\mathcal{F}$ if $B \notin \mathcal{F}$ but $B \cup \{x_i\} \in \mathcal{F}$ for some $x_i \in X_n$.

Let p be a path starting at the source of a n.d. BP1 and leading to an arbitrary edge $e = (v, w)$. We say that a partial input α *induces* the path p, if no variable in $S(\alpha)$ is tested on any path from w to a sink and if α is consistent with p (i.e. for any c-edge of p, $c \in \{0,1\}$, leaving a node marked with x_i, either $\alpha^i = c$ or $\alpha^i = *$).

Lemma 1. *Let G be a (possibly n.d.) BP1 on X_n. For each filter $\mathcal{F}$ of X_n there exists a set B in the boundary of $\mathcal{F}$ for which at least $\lceil 2^{|\overline{B}|}/(|G|-1)\rceil$ different partial assignments with support $\overline{B}$ ($= X_n \setminus B$) induce paths leading to the same edge.*

Proof. A filter $\mathcal{F}$ defines in the following way a cut, called *frontier*, through the edges of G. For a node v except the root let v^+ be the set of all variables that are assigned with a node reachable from v (including v). For the root r let $r^+ = X_n$. An edge (v,w) is in the frontier, and thus called *frontier edge*, if $v^+ \in \mathcal{F}$ but $w^+ \notin \mathcal{F}$. Note that each source-to-sink path passes through exactly one frontier edge, because $r^+ = X_n$ for the root r, $s^+ = \emptyset$ for each sink s and $w^+ \subseteq v^+$ for each edge (v,w). Furthermore, we may associate each frontier edge (v,w) with an arbitrary set $B \subseteq X_n$ in the boundary of $\mathcal{F}$ for which $w^+ \subseteq B \subseteq v^+ \setminus \{x_i\}$ if v is marked with x_i and $w^+ \subseteq B \subsetneq v^+$ if v is a n.d. node. Obviously, the inputs $x \in \{0,1\}^n$ reaching this edge are characterized exactly by their assignments to $\overline{B} := X_n \setminus B$.

Since each source-to-sink path of G contains exactly one frontier edge, we can embed a binary tree into G by letting each frontier edge point to a unique leaf. This tree has at most $|G|-2$ inner nodes (not counting the two sinks of G) and therefore at most $|G|-1$ leaves. Therefore, G contains at most $|G|-1$ frontier edges. Hence, there exists a frontier edge e, associated with a set B in the boundary of $\mathcal{F}$, through which at least $\lceil 2^n/\big(|G|-1\big)\rceil$ inputs pass through. Since the inputs reaching e are characterized exactly by their assignment to $\overline{B}$, there are at least $\left\lceil 2^{|\overline{B}|}/\big(|G|-1\big)\right\rceil$ assignments to $\overline{B}$ leading to this frontier edge. □

We use this lemma in the following proof of Theorem 1.

Proof (of Theorem 1). In the following, we write $f \le g$ for two functions f, g defined on the same domain, if $f(x) \le g(x)$ for all inputs x.

The proof is by induction on k. If $k = 1$, then G contains no n.d. nodes and is deterministic. Then, upon choosing W_1 and λ_1 appropriately, $f|_{\lambda_1}$ is an l-mixed subfunction of f (see Remark 1) and the claim follows from Proposition 1.

Let now $k > 1$ and G be a n.d. BP1 consisting of $L \le 2^l$ nodes, of which at most $k-1$ are nondeterministic. We assume w.l.o.g. that each n.d. node in G has two different successors (if this is not the case, we may replace the n.d. node with its successor). We show that if G computes the function f, then

$$\forall W_1, \lambda_1 \, \exists V_1, \alpha_1 \ne \beta_1 \, \forall \gamma_1 \in \{\alpha_1, \beta_1\} \ldots \; \forall W_k, \lambda_k \, \exists V_k, \alpha_k \ne \beta_k \, \forall \gamma_k \in \{\alpha_k, \beta_k\} :$$
$$\sum_{i=1}^{k} |W_i| > n - kl \quad \vee \quad f|_c \;\le\; \bigvee_{j=1}^{k} f|_{c_j}, \quad (1)$$

where c and c_j are defined as in Definition 5. Hence, f is not strictly k-wise l-mixed.

Let first $W_1 \subseteq X_n$ as well as a partial input λ_1 with support W_1 be chosen arbitrarily. We may assume that $|W_1| \leq n - kl$, since otherwise there is nothing to prove. Consider the restricted n.d. BP1 $G|_{\lambda_1}$ representing the function $f|_{\lambda_1}$ on the variables in $X' = X_n \setminus W_1$ and note that $n' := |X'| \geq kl$. We define a filter $\mathcal{F}$ on X'.

$$\mathcal{F} \;:=\; \bigl\{V \subseteq X' \;\big|\; |V| > n' - l\bigr\}.$$

Each set B in the boundary of $\mathcal{F}$ has a cardinality of $n' - l$, and hence $\bigl|\overline{B}\bigr| = l$ (note that $\overline{B} = X' \setminus B$ in this case). Because of Lemma 1, there exists a set B in the boundary of $\mathcal{F}$ such that at least $\lceil 2^{|\overline{B}|}/(L-1)\rceil \geq \lceil 2^l/(2^l-1)\rceil = 2$ distinct partial assignments with support $\overline{B}$ induce paths leading to the same edge (v, w). We let $V_1 = \overline{B}$ and $\alpha_1 \neq \beta_1$ be two such partial assignments with support V_1. Note that $|V_1| = l$. Finally, let γ_1 be chosen arbitrarily among α_1 and β_1. Let f_w be the subfunction defined by the (possibly n.d.) BP1 rooted at node w. All 1-inputs for f_w obviously are also 1-inputs for $f_{\lambda_1\gamma_1}$ and $f_{\lambda_1\gamma_1^*}$, because the inputs γ_1 and γ_1^* both induce paths leading to the edge (v, w) in $G|_{\lambda_1}$. Hence,

$$f_w \;\leq\; f|_{\lambda_1\gamma_1},\, f|_{\lambda_1\gamma_1^*}. \tag{2}$$

Now let p be the path induced by the partial input γ_1, leading from the source of $G|_{\lambda_1}$ to the edge (v, w). If there is no n.d. node on p, then $f|_{\lambda_1\gamma_1} = f_w$ and thus $f|_{\lambda_1\gamma_1} \leq f|_{\lambda_1\gamma_1^*}$ by (2). No matter how the choice of the remaining $W_i, \lambda_i, V_i, \alpha_i, \beta_i, \gamma_i$ will be, statement (1) is fulfilled.

Therefore, we assume that there is at least one n.d. node on the path p. Let u be the last n.d. node on this path and let (u, u_0) and (u, u_1) be the two outgoing edges of u, where (u, u_0) is the edge on p. We replace the n.d. node u with the node u_1 by redirecting all edges pointing to u in such a way that they point to u_1. Let f' $(= f'|_{\lambda_1})$ be the function computed by the resulting Branching Program. Obviously

$$f'|_{\lambda_1} \;\leq\; f|_{\lambda_1}, \tag{3}$$

because following the (u, u_0)-edge in the original BP1 $G|_{\lambda_1}$ might only allow additional inputs to lead to the 1-sink. Furthermore, since u was the last n.d. node on the path from the source to (v, w) induced by the partial input γ_1,

$$f|_{\lambda_1\gamma_1} \;=\; f'|_{\lambda_1\gamma_1} \vee f_w \;\overset{(2)}{\leq}\; f'|_{\lambda_1\gamma_1} \vee f|_{\lambda_1\gamma_1^*}. \tag{4}$$

As a last step, we restrict the so obtained n.d. BP1 for the function $f'|_{\lambda_1}$ to a n.d. BP1 G' for the function $f'|_{\lambda_1\gamma_1}$. Note that G' is of size smaller than 2^l and that $f'|_{\lambda_1\gamma_1}$ is a function on $n'' = n - l - |W_1| \geq (k-1)l$ variables. Furthermore, since we have removed the n.d. node u from $G|_{\lambda_1}$, G' contains at most $k-2$ n.d.

nodes. This means by the induction hypothesis that the subfunction $f'|_{\lambda_1\gamma_1}$ is not strictly $(k-1)$-wise l-mixed. In other words

$$\forall W_2, \lambda_2\, \exists V_2, \alpha_2 \neq \beta_2\, \forall \gamma_2 \in \{\alpha_2, \beta_2\} \ldots\ \forall W_k, \lambda_k\, \exists V_k, \alpha_k \neq \beta_k\, \forall \gamma_k \in \{\alpha_k, \beta_k\} :$$
$$\sum_{i=2}^{k} |W_i| > n'' - (k-1)l \quad \vee \quad (f'|_{\lambda_1\gamma_1})|_{c'} \leq \bigvee_{j=2}^{k} (f'|_{\lambda_1\gamma_1})|_{c'_j},$$

where c' is the partial input $\lambda_2\gamma_2 \ldots \lambda_k\gamma_k$ and c'_j is obtained from c' by replacing γ_j with γ_j^*.

Assume first that $\sum_{i=2}^{k} |W_i| > n'' - (k-1)l$. Because n'' equals $n - l - |W_1|$ it follows that $\sum_{i=1}^{k} |W_i| > n - kl$, and property (1) is fulfilled. Therefore, we assume $\sum_{i=2}^{k} |W_i| \leq n'' - (k-1)l$ and hence

$$(f'|_{\lambda_1\gamma_1})|_{c'} \leq \bigvee_{j=2}^{k} (f'|_{\lambda_1\gamma_1})|_{c'_j}. \tag{5}$$

Altogether we obtain

$$\begin{aligned}
f|_c = f|_{\lambda_1\gamma_1\ldots\lambda_k\gamma_k} &= (f|_{\lambda_1\gamma_1})|_{\lambda_2\gamma_2\ldots\lambda_k\gamma_k} \\
&\overset{(4)}{\leq} (f'|_{\lambda_1\gamma_1})|_{\lambda_2\gamma_2\ldots\lambda_k\gamma_k} \vee (f|_{\lambda_1\gamma_1^*})|_{\lambda_2\gamma_2\ldots\lambda_k\gamma_k} \\
&\overset{(5)}{\leq} \left(\bigvee_{j=2}^{k} (f'|_{\lambda_1,\gamma_1})|_{\lambda_2\ldots\lambda_k\gamma_2\ldots\gamma_{j-1}\gamma_j^*\gamma_{j+1}\ldots\gamma_k}\right) \vee (f|_{\lambda_1\gamma_1^*})|_{\lambda_2\gamma_2\ldots\lambda_k\gamma_k} \\
&\overset{(3)}{\leq} \left(\bigvee_{j=2}^{k} (f|_{\lambda_1\gamma_1})|_{\lambda_2\ldots\lambda_k\gamma_2\ldots\gamma_{j-1}\gamma_j^*\gamma_{j+1}\ldots\gamma_k}\right) \vee (f|_{\lambda_1\gamma_1^*})|_{\lambda_2\gamma_2\ldots\lambda_k\gamma_k} \\
&= \bigvee_{j=1}^{k} f|_{\lambda_1\ldots\lambda_k\gamma_1\ldots\gamma_{j-1}\gamma_j^*\gamma_{j+1}\ldots\gamma_k} = \bigvee_{j=1}^{k} f|_{c_j}.
\end{aligned}$$

Hence, we have proven (1). □

2.4 Lower Bounds for $(1, +k)$-BPs

The following theorem shows that the k-wise l-mixed property yields also lower bounds for deterministic $(1, +k)$-BPs. Note that the $(1, +k)$-restriction is semantic in this case.

Theorem 2. *Any $\big(1, +(k-1)\big)$-BP computing a k-wise l-mixed function has a size of at least $2^{l/2}$.*

The rest of this section is devoted to the proof of this theorem.

Jukna and Razborov [4] have used in their lower bound proofs a definition of so-called "forgetting pairs" with the intention that different pairs of partial inputs may "forget" the input bits in which they differ if their computation paths reach the same node. Although slightly different, the following definition uses a similar idea.

Let in the following for a deterministic BP G the unique computation path of an input a be denoted by $\mathrm{comp}(a)$.

Definition 6. Let v be a node of a BP G. The set $L(v)$ consists of all pairs (α, β) of partial inputs for which the following three conditions are fulfilled.

1. $S(\alpha) = S(\beta)$ and $\alpha \neq \beta$.
2. If c is a complete input consistent with α or β, then $\mathrm{comp}(c)$ passes through v.
3. If v is an internal node, then all variables in $D(\alpha, \beta)$ are tested on $\mathrm{comp}(c)$ before v is reached.

The set $L(G)$ is the union of all $L(v)$.

The idea behind the set $L(v)$ is that a computation reaching a node v cannot distinguish two partial inputs α, β with $(\alpha, \beta) \in L(v)$. In other words, the BP "forgets" at the node v that α and β are different. The following fact is obvious.

Fact 1. *Let G be a deterministic BP computing $f \in B_n$, $(\alpha, \beta) \in L(G)$ and x^* an assignment to the variables in $X_n \setminus S(\alpha)$. Then either $f|_\alpha(x^*) = f|_\beta(x^*)$ or there exists a variable $x_i \in D(\alpha, \beta)$ which is tested more than once as well on* $\mathrm{comp}(\alpha x^*)$ *as on* $\mathrm{comp}(\beta x^*)$.

Lemma 2. *Let G be a deterministic BP marked with variables in X_n and let $l \geq 2\lfloor \log |G| \rfloor + 1$ and $1 \leq k \leq n/l$. In the following formula we make the same restrictions on $W_j, \lambda_j, V_j, \alpha_j, \beta_j$ as in Definition 5. Then*

$$\forall W_1, \lambda_1 \, \exists V_1, \alpha_1 \neq \beta_1 \, \forall \gamma_1 \in \{\alpha_1, \beta_1\} \; \ldots \; \forall W_k, \lambda_k \exists V_k, \alpha_k \neq \beta_k \, \forall \gamma_k \in \{\alpha_k, \beta_k\} :$$
$$\sum_{i=1}^{k} |W_i| > n - kl \quad \vee \quad \forall 1 \leq j \leq k : \; (c, c_j) \in L(G),$$

where c is the partial input $\lambda_1 \gamma_1 \ldots \lambda_k \gamma_k$ and c_j is obtained from c by replacing γ_j with γ_j^*.

Proof (of Lemma 2). Let first $W_1 \subseteq X_n$ as well as an assignment λ_1 with support W_1 be chosen arbitrarily. Assume further that $|W_1| \leq n - kl$, since otherwise there is nothing to show. We restrict G with respect to λ_1 and obtain a BP $G|_{\lambda_1}$ representing the subfunction $f|_{\lambda_1}$ (note that $G|_{\lambda_1}$ is not larger than G). We show now for $G|_{\lambda_1}$ how to find a set V_1, $|V_1| \leq l$, as well as two distinct partial inputs α_1, β_1 having both support V_1 such that $(\alpha_1, \beta_1) \in L(G|_{\lambda_1})$. Note that if $|V_1| < l$, then we may add arbitrary new variables to V_1 until $|V_1| = l$, and obtain new partial inputs α_1, β_1 by assigning the constants 0 to the new variables. Clearly, after that it still holds $(\alpha_1, \beta_1) \in L(G|_{\lambda_1})$.

Assume first that there exists a computation path reaching a sink s by testing less than $r = \lfloor \log |G|_{\lambda_1}| \rfloor + 1$ different variables and let α be the partial assignment inducing this computation path ($|S(\alpha)| < r$). We choose an arbitrary variable $x_i \notin S(\alpha)$ and let $V_1 = S(\alpha) \cup \{x_i\}$. Then we let α_1 and β_1 be the partial inputs which extend α by the additional assignment $x_i = 0$ and $x_i = 1$, respectively. Obviously, $(\alpha_1, \beta_1) \in L(s)$, α_1 and β_1 both have support V_1 and $|V_1| \leq r \leq l$.

Assume now that on each computation path at least r variables are tested. Then we obtain 2^r computation paths of length r by following each path until exactly r variables have been tested. Since $2^r > |G|_{\lambda_1}|$, there exist two different such paths, induced by two partial inputs $\alpha \neq \beta$, that lead to the same node v. We extend α and β to α_1 and β_1 with support $V_1 = S(\alpha) \cup S(\beta)$ in such a way that α_1 and β_1 differ at most on the variables in $S(\alpha) \cap S(\beta)$. Clearly, it holds again $(\alpha_1, \beta_1) \in L(v)$ and $|V_1| \leq 2r - 1 \leq l$ (this is because $|S(\alpha)| = |S(\beta)| = r$ and because $S(\alpha)$ and $S(\beta)$ share at least the variable which marks the source).

So far, we have constructed upon arbitrary given W_1, λ_1 two distinct inputs α_1, β_1 with support V_1, $|V_1| = l$, such that $(\alpha_1, \beta_1) \in L(G|_{\lambda_1})$. Thus, for G it holds $(\lambda_1\alpha_1, \lambda_1\beta_1) \in L(G)$.

Now let $\gamma_1 \in \{\alpha_1, \beta_1\}$ be chosen arbitrarily. We restrict $G|_{\lambda_1}$ with respect to γ_1 and obtain V_2, α_2, β_2 for the restricted BP $G|_{\lambda_1\gamma_1}$ in the same way as above. This procedure can be repeated k times, because by the requirement that $|W_1| + \ldots + |W_k| \leq n - kl$ it is assured that even after the choice of λ_k there are still l variables not fixed by the assignment $\lambda_1\gamma_1 \ldots \lambda_{k-1}\gamma_{k-1}\lambda_k$. □

We can finally proof the lower bound technique for $(1, +k)$-BPs.

Proof (of Theorem 2). Let $f \in B_n$ and assume that there exists a $(1, +(k-1))$-BP G for f such that $|G| < 2^{l/2}$. We show that f is not k-wise l-mixed. By the assumption, we have $l \geq 2\lfloor \log |G| \rfloor + 1$ and may apply Lemma 2. Thus, in order to show that f is not k-wise l-mixed it suffices to show

$$\Big(\forall 1 \leq j \leq k : (c, c_j) \in L(G)\Big) \Rightarrow \Big(\forall x^* \, \exists j \in \{1, \ldots, k\} : f|_c(x^*) = f|_{c_j}(x^*)\Big),$$

where c and c_j have the same properties as in Lemma 2. Let $(c, c_j) \in L(G)$ for all $1 \leq j \leq k$. We choose an arbitrary assignment x^* to the variables not in $S(c)$ and let p denote the source-to-sink path induced by the complete input cx^*. Note that by the assumptions of Lemma 2 the sets V_j $(1 \leq j \leq k)$ are disjoint and $D(c, c_j) \subseteq V_j$. Hence, the sets $D(c, c_j)$ are disjoint, too. This means that since at most $k - 1$ variables are tested more than once on the path p, there exists by the pigeon hole principle a set $D(c, c_j)$ such that no variable in this set is tested more than once on p. This finally implies by Fact 1 that $f|_c(x^*) = f|_{c_j}(x^*)$. □

3 Applications

3.1 Linear Codes and d-rare m-dense Functions

As a first application of our lower bound technique, we consider d-rare m-dense functions, which have been investigated e.g. by Savický and Žák [8] and by Jukna and Razborov [4]. Such functions have been known to be hard for $(1,+k)$ BPs and our proof method now demonstrates that they are also hard for n.d. BP1s, if not enough nondeterminism is available.

Definition 7. A function $f \in B_n$ is called *d-rare* if any two different inputs $a, b \in f^{-1}(1)$ have a Hamming distance of at least d (i.e. $|D(a,b)| \geq d$). The function f is called *m-dense* if $|S(\alpha)| \geq m$ for any partial input α with $f|_\alpha = 0$.

Theorem 3. *Any d-rare m-dense function is strictly k-wise l-mixed for* $l < \min\{d,\, m/k\}$.

Proof. This proof is simple, because we do not need to bother about the choice of W_i, λ_i or γ_i for $1 \leq i \leq k$. We simply choose $W_i = \emptyset$, $\lambda_i = \varepsilon$ and upon some arbitrarily given $\alpha_i \neq \beta_i$ with support V_i, $|V_i| = l$, we choose $\gamma_i = \alpha_i$. Then we consider the partial input $c = \lambda_1\gamma_1 \dots \lambda_k\gamma_k$ and the inputs c_j which are obtained from c by replacing γ_j with γ_j^*. Since f is m-dense and $S(c) \leq kl < m$ by construction, we know that the subfunction $f|_c$ has an input x^* such that $f|_c(x^*) = 1$. For this x^* on the other hand, the complete inputs cx^* and c_jx^* have a Hamming distance of $|D(c,c_j)| \leq l < d$. Therefore, the d-rareness of f implies $f|_{c_j}(x^*) \neq 1$ for all $1 \leq j \leq k$. □

Corollary 1. *Any n.d. BP1 with at most k n.d. nodes or any* $(1,+k)$*-BP representing a d-rare m-dense function has a size of at least* $\min\left\{2^{(d-1)/2},\, 2^{\lfloor (m-1)/(k+1)\rfloor/2}\right\}$.

Note that the same result for $(1,+k)$-BPs was already obtained by Jukna and Razborov [4] with a different technique. The result for n.d. BP1s is new though. The authors of [4] also show that if C is a linear code over $GF(2)$ with minimal distance d_1 and if $C^\perp$ is its dual with minimal distance d_2, then the characteristic function of C is d_1-rare and d_2-dense. This leads to a lower bound of $\min\left\{2^{(d_1-1)/2},\, 2^{\lfloor (d_2-1)/(k+1)\rfloor/2}\right\}$ for n.d. BP1s with at most k n.d. nodes and for $(1,+k)$-BPs representing such a linear code. We only state one corollary for Reed-Muller codes, which follows instantly from the discussion in [4] and was stated there for $(1,+k)$-BPs.

Corollary 2. *Let* $R(r,\ell)$ *be the rth order binary Reed-Muller code of length* $n = 2^\ell$. *Let further* $0 \leq k \leq n$ *and* $r = \lfloor 1/2(\ell + \log(k+1))\rfloor$. *Then any* $(1,+k)$*-BP and any n.d. BP1 with at most k n.d. nodes representing the characteristic function of* $R(r,\ell)$ *has size at least* $2^{\Omega\left(\sqrt{n/(k+1)}\right)}$.

3.2 Disjoint Conjunctions

We state now a theorem which describes how to obtain k-wise l-mixed functions from l-mixed ones. Let f be a function on variables in X_n. We consider the disjoint conjunction f^k with respect to f on the variables in X_{kn}, which is defined as follows. For $x = (x_1, \ldots, x_{kn}) \in \{0,1\}^{kn}$ let $f^k(x) = f_1(x) \wedge \ldots \wedge f_k(x)$, where $f_i(x) = f(x_{(i-1)n+1}, \ldots, x_{in})$.

Theorem 4. *If f is (kl)-mixed, then f^k is strictly k-wise l-mixed.*

The l-mixedness of a function f implies that f is hard to compute by BP1s or equivalently by $(1,+0)$-BPs. Hence, the above theorem leads to a generalization of this fact in the sense that the disjoint conjunction f^k of a (kl)-mixed function f is hard to compute for a $(1, +(k-1))$-BP.

Proof. Let $N = kn$ and let X_i be the set of variables on which f_i may depend, i.e. $X_i = \{x_{(i-1)n+1}, \ldots, x_{in}\}$. We first choose $W_1 = \emptyset$ and $\lambda_1 = \varepsilon$. Then we consider a k-round game in which we play against an adversary who starts the ith round by choosing V_i, α_i, β_i, after which we are allowed to choose W_{i+1}, λ_{i+1}. We show that we can influence the game by our choices in such a way that after $r \le k$ rounds the following situation is obtained for any $j \in \{1, \ldots, r\}$.

(I1) The set W_{j+1} only consists of variables in X_{i_j} for some $i_j \in \{1, \ldots, k\}$ and the indices $i_1, \ldots, i_r$ are all different.
(I2) All variables in $X_{i_1}, \ldots, X_{i_r}$ are fixed by $\lambda_1\gamma_1 \ldots \lambda_r\gamma_r\lambda_{r+1}$, i.e. $X_{i_1} \cup \ldots \cup X_{i_r} \subseteq W_1 \cup V_1 \cup \ldots \cup W_r \cup V_r \cup W_{r+1}$.
(I3) In each X_i, $i \notin \{i_1, \ldots, i_r\}$, there are at most rl variables fixed.
(I4) $f_{i_j}|_{\lambda_1\gamma_1\ldots\lambda_j\gamma_j\lambda_{j+1}} = 1$ and $f_{i_j}|_{\lambda_1\gamma_1\ldots\lambda_j\gamma_j^*\lambda_{j+1}} = 0$.

Assume that we have played the game k rounds in such a way that (I1)-(I4) are fulfilled for $r = k$. Because all indices $i_1, \ldots, i_k$ are different (I1), and all variables in $X_{i_1}, \ldots, X_{i_k}$ are fixed (I2), the assignment $\lambda_1\gamma_1 \ldots \lambda_k\gamma_k\lambda_{k+1}$ forms a complete input for f^k. We let $x^* = \lambda_{k+1}$ and c, c_j as in Definition 5. Then property (I4) implies on one hand that $f^k|_c(x^*) = 1$, while on the other for each $1 \le j \le k$ it follows from $f_{i_j}|_{c_j}(x^*) = 0$ that $f^k|_{c_j}(x^*) = 0$. Furthermore, by property (I1) each of the sets $W_2, \ldots, W_k$ contains at most n variables. Since in addition $W_1 = \emptyset$, we have $\sum_{i=1}^{k} |W_i| \le (k-1)n = N - n \le N - kl$. For the last inequality we have used $kl \le n$, which follows from the (kl)-mixedness of f. Altogether, the conditions of Definition 5 showing that f^k is strictly k-wise l-mixed are fulfilled.

Therefore, it suffices to show that we can play the game for k rounds such that after each round (I1)-(I4) hold. This is trivially true after 0 rounds, and we show now the claim for the $(r+1)$th round ($1 \le r+1 \le k$).

Let the adversary choose V_{r+1} and $\alpha_{r+1} \ne \beta_{r+1}$. Then there exists a variable $x_i \in V_{r+1}$ which is fixed to different constants by α_{r+1} and β_{r+1}. Let i_{r+1} be the index for which $x_i \in X_{i_{r+1}}$. Note that $i_{r+1} \notin \{i_1, \ldots, i_r\}$, because by (I2) all variables in the sets $X_{i_1}, \ldots, X_{i_r}$ had been fixed in previous rounds and are therefore not contained in V_{r+1}.

Now we restrict the partial input $\lambda_1\gamma_1 \ldots \lambda_r\gamma_r\lambda_{r+1}\alpha_{r+1}$ to the variables in $X_{i_{r+1}}$ and obtain a partial input α. In the same way, we obtain β by restricting $\lambda_1\gamma_1 \ldots \lambda_r\gamma_r\lambda_{r+1}\beta_{r+1}$ to the variables in $X_{i_{r+1}}$. Obviously, $S(\alpha) = S(\beta)$ and by our choice of $X_{i_{r+1}}$, it is $\alpha \neq \beta$. Furthermore, using the fact that $|V_{r+1}| = l$, we know by (I3) that not more than $(r+1)l$ variables are fixed in $X_{i_{r+1}}$, hence $|S(\alpha)| = |S(\beta)| \le kl$. Using the assumption that f is (kl)-mixed, this implies by Proposition 1 that $f|_\alpha \neq f|_\beta$ (we assume here that the input variables for f are in $X_{i_{r+1}}$ instead of in X_n). Thus, there exists an assignment y to the free variables in $X_{i_{r+1}}$ as well as a choice $\gamma \in \{\alpha, \beta\}$ such that $f|_\gamma(y) = 1$ and $f|_{\gamma^*}(y) = 0$. We finally let γ_{r+1} be the element in $\{\alpha_{r+1}, \beta_{r+1}\}$ which corresponds to the above choice of γ. Then obviously $f_{i_{r+1}}|_{\lambda_1\gamma_1\ldots\lambda_r\gamma_r\lambda_{r+1}\gamma_{r+1}}(y) = 1$ and $f_{i_{r+1}}|_{\lambda_1\gamma_1\ldots\lambda_r\gamma_r\lambda_{r+1}\gamma^*_{r+1}}(y) = 0$. Hence, if we let $\lambda_{r+2} := y$ and $W_{r+2} = S(\lambda_{r+2})$, then (I4) is fulfilled. Furthermore, by construction W_{r+2} only consists of variables in $X_{i_{r+1}}$ and all variables in $X_{i_1}, \ldots, X_{i_{r+1}}$ are fixed. Therefore also (I1) and (I2) hold. Condition (I3) follows already from (I1) and (I2), because each of the assignments $\gamma_1, \ldots, \gamma_r$ fixes at most l variables and the assignments $\lambda_1, \ldots, \lambda_{r+1}$ fix only variables in X_i, $i \in \{i_1, \ldots, i_r\}$. We have shown therefore, that there exists a playing strategy such that for any $0 \le r \le k$ after the rth round the conditions (I1)-(I4) are fulfilled. This proves the claim. □

4 Improving the Hierarchy for $(1, +k)$-BPs

We consider now the function weighted sum, which was used by Savický and Žák [9] in order to prove a hierarchy for $(1, +k)$-BPs.

Definition 8. For any positive integer n let $p(n)$ be the smallest prime greater than n. The function $\mathrm{WS}_n \in B_n$ (called *weighted sum*) is defined by

$$\mathrm{WS}_n(x) = \begin{cases} x_s & \text{if } s \in \{1, \ldots, n\} \\ x_1 & \text{otherwise,} \end{cases} \qquad \text{where} \quad s = \left(\sum_{i=1}^{n} i x_i\right) \bmod p(n).$$

Savický and Žák have shown that WS_n is l-mixed for large l, and have used this function in order to obtain a hierarchy for $(1, +k)$-BPs.

Theorem 5 ([9]). *For any $\delta > 0$ and any large enough n, the function WS_n is l-mixed for $l = n - \lfloor (2+\delta)\sqrt{n} \rfloor - 2$.*

Theorem 6 ([9]). *There exists a class of functions $h_{n,k} \in B_n$ representable by polynomial-size $(1, +k)$-BPs but not by polynomial-size $\big(1, +(k-1)\big)$-BPs as long as $k \le n^{1/6}/(\log^{1/3} n)$.*

It is obvious how to construct a syntactic $(1, +1)$-BP representing the function WS_n with at most $O(n^2)$ nodes. On the other hand, Theorem 5 implies that any $(1, +0)$-BP (or equivalently any BP1) for WS_n has exponential size. We may now look at the disjoint conjunction $f_{N,k} := (\mathrm{WS}_n)^k$ with respect to WS_n. Note that $f_{N,k}$ is a function in $N = kn$ variables. Furthermore, it is easy to see that $f_{N,k}$ can be computed by a syntactic $(1, +k)$-BP of size $O(kn^2) = O(N^2/k)$. But by Theorem 1 and Theorem 4 we know that any $\bigl(1, +(k-1)\bigr)$-BP for $f_{N,k}$ has a size of at least $2^{l/(2k)}$, where $l = n - \lfloor (2+\delta)\sqrt{n} \rfloor - 2 = \Omega(n) = \Omega(N/k)$. (for any δ and sufficiently large n). Hence, we get an improved hierarchy as described by the following corollary.

Corollary 3. *The function $f_{N,k} \in B_N$ can be represented by polynomial size syntactic $(1, +k)$-BPs but not by polynomial size $\bigl(1, +(k-1)\bigr)$-BPs for $k = o\Bigl(\sqrt{N/\log N}\Bigr)$.*

5 A Hierarchy for BP1s with Restricted Amount of Nondeterminism

We finally develop a family of multipointer functions g_n^k, which can be easily computed in polynomial size by $(k+1)$-PBDDs. On the other hand, for any BP1 having at most $k-1$ n.d. nodes, an exponential size is required. Recall that a $(k+1)$-PBDD can be regarded as a restricted n.d. BP1 having exactly k n.d. nodes at the top.

The idea behind the following definition of g_n^k is inspired by the functions used by Savický and Sieling [7] for their hierarchy result for $(\vee, k)$-BP1s. Let n and k be arbitrary integers such that $k(k+1) \le n/\lceil \log n \rceil$, and let $b = \lfloor n/(k(k+1)) \rfloor$ and $s = \lfloor b/\lceil \log(kb) \rceil \rfloor$. We partition the n variables in X_n into $k(k+1)$ consecutive blocks $B_{i,j}$ of size b, where $i \in \{0, \dots, k\}$ and $j \in \{0, \dots, k-1\}$, and possibly one block of the remaining variables. Each block $B_{i,j}$ is then partitioned into $\lceil \log(kb) \rceil$ subblocks, each having either size s or size $s+1$. The set of the input variables in the blocks $B_{i,0}, \dots, B_{i,k-1}$ is called the *sector* S_i $(0 \le i \le k)$ of the input and has cardinality kb. For the ease of notation, we enumerate the variables in such a way that the sector S_i contains the variables $x_{i,0}, \dots, x_{i,kb-1}$.

The function value of the function g_n^k is determined as follows. The majority of the setting of the s variables in each subblock of a block $B_{i,j}$ determines a bit. The $\lceil \log(kb) \rceil$ bits obtained this way for a block $B_{i,j}$ are interpreted as an integer in $\bigl\{0, \dots, 2^{\lceil \log(kb) \rceil} - 1\bigr\}$, which is taken modulo kb such that a value $p_{i,j} \in \{0, \dots, kb-1\}$ is obtained. This value $p_{i,j}$ points to the variable $x_{i \oplus j, p_{i,j}}$ in the sector $S_{i \oplus j}$, where $i \oplus j := (i+j+1) \bmod (k+1)$. Let $h_{i,j}$ be the function which computes the value of the input variable the pointer $p_{i,j}$ points to, i.e.

$h_{i,j}(x) = x_{i\oplus j, p_{i,j}}$. Then

$$h_i \;:=\; \bigwedge_{j=0}^{k-1} h_{i,j} \quad \text{and} \quad g_n^k \;:=\; \bigvee_{i=0}^{k} h_i.$$

Since the function g_n^k is the disjunction of $k+1$ functions h_i, a correct $(k+1)$-PBDD may consist of $k+1$ OBDDs, each representing a function h_i $(0 \le i \le k)$. The idea behind constructing an OBDD which represents h_i is that the OBDD reads a block $B_{i,j}$ of the sector S_i, determines the corresponding pointer $p_{i,j}$ and finally may obtain the value of the function $h_{i,j}$. Depending on whether this value is 0 or 1, the OBDD stops with output 0 or continues this proceeding with the next block $B_{i,j+1}$ in the sector S_i. Before we make this idea more precise, we note a simple fact about the functions g_n^k.

Fact 2. *For any i, the pointers $p_{i,0}, \dots, p_{i,k-1}$ all point to variables in different sectors in $\{S_0, \dots, S_k\} \setminus \{S_i\}$.*

This is clear from the definition of $i \oplus j$.

Theorem 7. *There exists a $(k+1)$-PBDD for g_n^k of size $O(n^3/k^3)$.*

Proof. We describe for arbitrary $i \in \{0, \dots, k\}$ the construction of a BP G_i of size $O(n^3/k^4)$ representing the function h_i. We argue that each G_i is in fact an OBDD such that the desired $(k+1)$-PBDD consists of $G_0, \dots, G_k$.

In order to evaluate h_i for an input x, the BP G_i first tests all variables in the block $B_{i,0}$ and computes the value of the pointer $p_{i,0} \in \{0, \dots, kb-1\}$. The majority of each subblock can be computed with $O(s^2)$ nodes by simply counting the number of variables which are set to 1. In order to compute the pointer $p_{i,j}$, the BP reads a subblock, stores the resulting majority and continues reading the next subblock until all $\lceil \log kb \rceil$ bits are stored. It is easy to verify that this can be done with

$$O\Big(s^2\Big(1 + 2 + 4 + \ldots + 2^{\lceil \log kb \rceil}\Big)\Big) \;=\; O(s^2 kb)$$

nodes. After that, the value of $p_{i,0}$ is uniquely defined by the bits computed this way, and the BP may finally read in sector $S_{i\oplus 0}$ the variable to which this pointer points to. If this variable is set to 0, then $h_{i,0}(x) = 0$, and the BP reaches the 0-sink. Otherwise, the BP repeats the procedure described above with the blocks $B_{i,1}, \dots, B_{i,k-1}$, and returns 1 if no 0-sink is reached in this proceeding.

It is clear from the description that G_i computes the function h_i. Note that by Fact 2 the sectors $S_i, S_{i\oplus 0}, \dots, S_{i\oplus(k-1)}$ are all different such that no variable needs to be tested more than once. Furthermore, the sectors $S_{i\oplus 0}, \dots, S_{i\oplus(k-1)}$ are read in this order, and each sector $S_{i\oplus j}$ is read right after the block $B_{i,j}$. Thus, the variable ordering can be fixed in advance and G_i is an OBDD. Since the processing of one block in the sector S_i requires at most $O(s^2kb)$ nodes, the BP G_i may be constructed out of $O(s^2k^2b) = O(n^3/k^4)$ nodes. □

As we have shown above, it is easy for $(k+1)$-PBDDs to compute the functions g_n^k, and therefore also for n.d. BP1s with k n.d. nodes. We prove now that g_n^k is k-wise l-mixed for large l, and hence is hard to compute for n.d. BP1s with at most $k-1$ n.d. nodes.

Theorem 8. *Any n.d. BP1 computing g_n^k has a size of at least $2^{\Omega\left(n/(k^3 \log n)\right)}$, if the BP1 contains at most $k-1$ n.d. nodes. This is not polynomial for $k = o\left(n^{1/3}/\log^{2/3} n\right)$*

Before we prove the theorem, we state another simple fact about the functions g_n^k.

Fact 3. *Let $V \subseteq X_n$. Then for any block $B_{i,j}$ such that $|B_{i,j} \cap V| \leq s/2 - 1$ and any variable $x_{i\oplus j,q}$ in the sector $S_{i\oplus j}$, there exists an assignment λ to the variables in $B_{i,j} \setminus V$ such that independently of any assignment to V, the pointer $p_{i,j}$ points to $x_{i\oplus j,q}$ (i.e. $h_{i,j}|_\lambda = x_{i\oplus j,q}$).*

Proof. Recall that each subblock of $B_{i,j}$ consists of either s or $s+1$ variables. Since at least $s/2+1$ variables are available in $B_{i,j} \setminus V$, the majority in each subblock of $B_{i,j}$ can be influenced by assigning appropriate values to these variables independently from any setting to V. This way, the value of the pointer $p_{i,j}$ can be set to all possible integers in $\{0, \ldots, kb-1\}$. □

Let in the rest of this section

$$l = \left\lfloor \frac{s/2-2}{k} \right\rfloor = \Omega\left(\frac{n}{k^3 \log(n/k)}\right).$$

We show below that g_n^k is strictly k-wise l-mixed.

Similar to the proof of Theorem 4, we play a k-round game against an adversary. Before the game starts, we choose λ_1 in such a way that it fixes the first variable of each sector (i.e. $x_{i,0}$ for $0 \leq i \leq k$) to the constant 0. Then k rounds are played where the adversary starts the ith round by choosing $V_i, \alpha_i \neq \beta_i$ upon which we are allowed to choose $\gamma_i \in \{\alpha_i, \beta_i\}$ and W_{i+1}, λ_{i+1}.

Claim 1. *Let W_1, λ_1 be chosen as described above before the game starts. There is a playing strategy for the game guaranteeing that the following situation is obtained after $r \leq k$ rounds for all $t \in \{1, \ldots, r\}$.*

(I1) For some $i_t \in \{0, \ldots, k\}$ and $j_t \in \{0, \ldots, bk\}$, there is a variable x_{i_t,j_t} such that the assignment γ_t sets the variable x_{i_t,j_t} to 1 and the assignment γ_t^* sets it to 0.

(I2) The indices $i_1, \ldots, i_r$ are all different.

(I3) All variables in the sectors $S_{i_1}, \ldots, S_{i_r}$ are fixed by $\lambda_1\gamma_1 \ldots \lambda_r\gamma_r\lambda_{r+1}$, i.e.

$$S_{i_1} \cup \ldots \cup S_{i_r} \subseteq W_1 \cup V_1 \cup \ldots \cup W_r \cup V_r \cup W_{r+1}.$$

(I4) In each sector S_i, $i \notin \{i_1, \ldots, i_r\}$, there are at most $rl+1$ variables fixed.

(I5) Each pointer $p_{i_t,j}$ $(0 \leq j \leq k-1)$ of the sector S_{i_t} points to $x_{i_t\oplus j,0}$ (which is the first variable in the sector $S_{i_t\oplus j}$).

Proof. For $r = 0$, the claim is trivially fulfilled. Assume therefore that (I1)-(I5) hold after $r - 1$ rounds. We describe the strategy in the rth round.

Let $V_r, \alpha_r \neq \beta_r$ be chosen by the adversary and let i_r, j_r be arbitrary indices for which α_r and β_r differ in their assignment to x_{i_r,j_r}. Choose $\gamma_r \in \{\alpha_r, \beta_r\}$ to be the assignment which sets x_{i_r,j_r} to 1. This choice already fulfills (I1). Since by (I3) after the previous round all variables in the sectors $S_{i_1}, \ldots, S_{i_{r-1}}$ had been fixed, we know that i_r is different from $i_1, \ldots, i_{r-1}$, and thus (I2) holds, too.

Now we choose W_{r+1} to contain all variables in the sector S_{i_r} which have not yet been fixed (we determine the choice of λ_{r+1} later). This satisfies (I3). Furthermore, by (I4), after the previous round there have been in any sector S_i, $i \notin \{i_1, \ldots, i_r\}$ at most $(r-1)l + 1$ variables been fixed. But W_{r+1} contains by definition no variables in any of these sectors and V_r contains only l variables at all. Hence, (I4) also holds after this round.

We finally have to show that (I5) can be fulfilled by the appropriate choice of λ_{r+1}, where $S(\lambda_{r+1}) = W_{r+1}$. Let V be the variables being fixed by the assignments in the previous rounds as well as by the assignment γ_r. We know – because (I4) was fulfilled after the previous round and because γ_r fixes l variables – that for any block $B_{i_r,j}$ $(0 \leq j \leq k-1)$ it holds

$$|V \cap B_{i_r,j}| \;\leq\; rl + 1 \;\leq\; r(s/2-2)/k + 1 \;\leq\; s/2 - 1.$$

Hence, by Fact 3 we may choose λ_{r+1} in such a way that all pointers $p_{i_r,j}$ point to $x_{i_t \oplus j,0}$. Therefore, also (I5) is satisfied. □

This k-round strategy allows us finally to prove Theorem 8.

Proof (of Theorem 8). Assume that we have played the game k rounds and that w.l.o.g. $i_t = t$ for $1 \leq t \leq k$. I.e., 0 is the only index in $\{0, \ldots, k\}$ that is not contained in $\{i_1, \ldots, i_k\}$. Note that by our choice of λ_1 before the game started, we know that all the first variables of each sector are set to 0. Property (I5) means that all pointers determined by the sectors $S_1, \ldots, S_k$ point to these variables and hence all subfunctions obtained from $h_1, \ldots, h_k$ by the restriction $\lambda_1\gamma_1 \ldots \lambda_k\gamma_k\lambda_{k+1}$ equal 0. Since h_0 is the only function that is not degenerated to 0 by this restriction, this means that for $1 \leq j \leq k$ and for c, c_j as in Definition 5,

$$g_n^k|_{c\lambda_{k+1}} \;=\; h_0|_{c\lambda_{k+1}} \quad \text{and} \quad g_n^k|_{c_j\lambda_{k+1}} \;=\; h_0|_{c_j\lambda_{k+1}}. \tag{6}$$

Note that by (I4) there are in each subblock of the sector S_0 at most $kl+1 \leq s/2-1$ variables fixed. Hence, by Fact 3 we may find an assignment y to the free variables in such a way that each of the pointers $p_{0,z}$ $(0 \leq z \leq k-1)$ points to one of the variables x_{i_t,j_t}, for $t = 1, \ldots, k$. Then we know that $h_{0,z}|_{c\lambda_{k+1}}(y) = x_{i_t,j_t}|_{c\lambda_{k+1}} = 1$ because of (I1). Therefore, $h_0|_{c\lambda_{k+1}}(y) = 1$. On the other hand, for each $1 \le j \le k$

$$h_0|_{c_j\lambda_{k+1}}(y) \;=\; \bigwedge_{t=1}^{k} x_{i_t,j_t}|_{c_j\lambda_{k+1}} \;=\; 0,$$

which is because by (I1) $x_{i_t,j_t}|_{\gamma_j^*} = 0$ for $i_t = j$. Thus, letting $x^* = y\lambda_{k+1}$ and invoking (6) shows that $g_n^k|_c(x^*) = g_n^k|_{c\lambda_{k+1}}(y) = 1$ and $g_n^k|_{c_j}(x^*) = g_n^k|_{c_j\lambda_{k+1}}(y) = 0$ for each $1 \le j \le k$. Further, the sets $W_2, \dots, W_k$ contain no variables in S_0 and W_1 contains only one variable in S_0. Hence,

$$\sum_{i=1}^{k} |W_i| \;\le\; n - |S_0| + 1 \;=\; n - kb + 1 \le n - kl.$$

Altogether we have shown that g_n^k is k-wise l-mixed. $\square$

Obviously, the Theorems 7 and 8 yield a polynomial size hierarchy for nondeterministic BP1s with respect to the number of nondeterministic nodes. We summarize this result by the following corollary.

Corollary 4. *Let* $\mathrm{NP}_k(\mathrm{BP1})$ *be the class of Boolean functions which can be represented in polynomial size by n.d. BP1s with at most k n.d. nodes. Then* $\mathrm{NP}_k(\mathrm{BP1}) \subsetneq \mathrm{NP}_{k+1}(\mathrm{BP1})$ *if* $k = o\big(n^{1/3}/\log^{2/3} n\big)$.

Acknowledgment

I thank Beate Bollig, Martin Sauerhoff, Detlef Sieling and Ingo Wegener for proofreading and helpful comments.

References

1. A. Andreev, J. Baskakov, A. Clementi, and J. Rolim. Small pseudo-random sets yield hard functions: New tight explict lower bounds for branching programs. In *Proceedings of the 26th International Colloquium on Automata, Languages, and Programming*, vol. 1644 of *Lecture Notes in Computer Science*, pp. 179–189. 1999.

2. B. Bollig and I. Wegener. Complexity theoretical results on partitioned (nondeterministic) binary decision diagrams. *Theory of Computing Systems*, 32:487–503, 1999.

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

4. S. Jukna and A. Razborov. Neither reading few bits twice nor reading illegally helps much. *Discrete Applied Mathematics*, 85:223–238, 1998.

5. M. Sauerhoff. Computing with restricted nondeterminism: The dependence of the OBDD size on the number of nondeterministic variables. In *Proceedings of the 19th Conference on Foundations of Software Technology and Theoretical Computer Science*, vol. 1738 of *Lecture Notes in Computer Science*, pp. 342–355. 1999.

6. M. Sauerhoff. An improved hierarchy result for partitioned BDDs. *Theory of Computing Systems*, 33:313–329, 2000.

7. P. Savický and D. Sieling. A hierarchy result for read-once branching programs with restricted parity nondeterminism. In *Mathematical Foundations of Computer Science: 25th International Symposium*, vol. 1893 of *Lecture Notes in Computer Science*, pp. 650–659. 2000.

8. P. Savický and S. Žák. A lower bound on branching programs reading some bits twice. *Theoretical Computer Science*, 172:293–301, 1997.

9. P. Savický and S. Žák. A read-once lower bound and a $(1,+k)$-hierarchy for branching programs. *Theoretical Computer Science*, 238:347–362, 2000.

10. D. Sieling. New lower bounds and hierarchy results for restricted branching programs. *Journal of Computer and System Sciences*, 53:79–87, 1996.

11. D. Sieling. A separation of syntactic and nonsyntactic $(1,+k)$-branching programs. 9:247–263, 2000.

12. J. Simon and M. Szegedy. A new lower bound theorem for read-only-once branching programs and its applications. In *Advances in Computational Complexity Theory*, vol. 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pp. 183–193. AMS, 1993.

13. I. Wegener. On the complexity of branching programs and decision trees for clique functions. *Journal of the ACM*, 35:461–471, 1988.

14. I. Wegener. *Branching Programs and Binary Decision Diagrams - Theory and Applications*. Siam, first edition, 2000.

15. S. Žák. An exponential lower bound for one-time-only branching programs. In *Mathematical Foundations of Computer Science: 11th International Symposium*, vol. 176 of *Lecture Notes in Computer Science*, pp. 562–566. 1984.