Changing Partitions in Rectangle Decision Lists

Stefan Mengel

Changing Partitions in Rectangle Decision Lists

Stefan Mengel
Univ. Artois, CNRS, Centre de Recherche en Informatique de Lens (CRIL), Lens, France

Abstract

Rectangle decision lists are a form of decision lists that were recently shown to have applications in the proof complexity of certain OBDD-based QBF-solvers. We consider a version of rectangle decision lists with changing partitions, which corresponds to QBF-solvers that may change the variable order of the OBDDs they produce. We show that even allowing one single partition change generally leads to exponentially more succinct decision lists. More generally, we show that there is a succinctness hierarchy: for every $k \in \mathbb{N}$, when going from k partition changes to $k+1$, there are functions that can be represented exponentially more succinctly. As an application, we show a similar hierarchy for OBDD-based QBF-solvers.

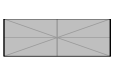
2012 ACM Subject Classification Theory of computation → Complexity theory and logic; Theory of computation → Proof complexity

Keywords and phrases rectangle decision lists, QBF proof complexity, OBDD

Digital Object Identifier 10.4230/LIPIcs.SAT.2022.17

Funding *Stefan Mengel*: This work has been partly supported by the PING/ACK project of the French National Agency for Research (ANR-18-CE40-0011).

Acknowledgements The author would like to thank the reviewers for their generous and very detailed comments that greatly improved the presentation of this paper.



1 Introduction

Decision lists are a classical formalism for encoding Boolean functions. Intuitively, they consist of a list of lines of the form "if $L_i(X)$ then c_i" where $L_i(X)$ is a condition on the input variables X and c_i is a constant from $\{0,1\}$. To evaluate a decision list on an input, one traverses the lines from the first to the last line and checks the respective condition $L_i(X)$. As soon as one meets a condition $L_i(X)$ that evaluates to true, one stops this process and returns the corresponding value c_i. Decision lists were originally introduced by Rivest in learning theory [22] and have since then been studied in different areas.

There are different types of decision lists, depending on which form the conditions $L_i(X)$ have. In the classical work of Rivest [22] they are terms of width k which already makes the model powerful enough to strictly generalize k-DNF, k-CNF and k-decision trees. Other works consider different classes of functions as the $L_i(X)$, e.g. linear threshold functions [8, 24], bounded depth circuits [2], and combinatorial rectangles [15, 21].

In recent years, decision lists have become important in the context of proof complexity of quantified Boolean formulas (QBF). The idea is that for several proof systems, one can from a refutation of an input formula efficiently extract a winning strategy of the universal player that is encoded as a decision list. The type of the decision list depends on the respective proof system, see e.g. [1, 2, 3]. Since the length of the decision list depends on the size of the refutation, lower bounds on decision lists then translate to lower bounds for proofs in the proof system at hand.

In this paper, we focus on so-called rectangle decision lists, i.e., decision lists in which all of the $L_i(X)$ are combinatorial rectangles, so functions of the form $L_i(X) = r_i^1(X_1) \wedge r_i^2(X_2)$ where (X_1, X_2) is a partition of X into sets of equal size and r_i^1, r_i^2 are arbitrary Boolean functions in X_1 and X_2, respectively. Rectangle decision lists were originally introduced in communication complexity theory [15, 21][1]. Recently, they were also considered in QBF proof complexity in [19] where they were used to show lower bounds to certain OBDD-based proof systems.

While in settings from communication complexity it makes sense to assume that the partition (X^1, X^2) is part of the problem statement and is thus fixed from the outside and the same throughout the decision list, this is not the case in the QBF setting: the solvers modeled by the OBDD-based proof system can freely choose the variable order of the OBDDs they construct. On the side of the decision lists, this results in a choice of the partition (X_1, X_2) that is used throughout the proof. Thus, to show lower bounds, unlike in [15, 21] which considered one fixed partition, in [19] it was shown that there are functions for which all possible balanced partitions decision lists must be long. This yields lower bounds for solvers that choose the variable order of the OBDDs optimally at the beginning and then work with that order throughout their run.

While this is in fact what the QBF-solver QBDD [20] does, this result is somewhat unsatisfying since it does not use the full power of modern practical OBDD-libraries: state-of-the-art OBDD-implementations like CUDD [23] allow reordering the variables of OBDDs. So it is conceivable to implement OBDD-based solvers for QBF that adapt the variable order throughout their run when this appears useful. Could this make the resulting solvers more powerful?

This paper answers the above question positively: even if we allow only one change of

[1] Note that while those papers introduced the concept of rectangle decision lists they did not use that name which seems to first have been introduced independently in [8] and in [12]

the variable order in the computation of the solver, this leads to exponentially faster runtime on some formulas. More generally, we show that for every constant $k \in \mathbb{N}$, allowing $k+1$ variable order changes yields exponential runtime savings over k variable order changes. We show this by using the translation of lower bound questions for OBDD-based QBF-solvers into lower bounds for rectangle decision lists from [19]. In this translation, every variable order change on the OBDDs becomes a potential partition change in the decision list. Thus, we study lower bounds for rectangle decision lists with changing partitions and show that their length decreases exponentially for some functions whenever one additional partition change is allowed.

Note that different partitions for rectangles have been studied before in so-called multi-partition communication complexity. In particular, there is a strict hierarchy with respect to the number of partitions in that setting as well [10]. Since rectangle decision lists are easily seen to simulate multi-partition communication protocols, our lower bound results for fixed k are qualitatively stronger than those of [10], even though the dependence on k is far better in [10] and k is not needed to be constant there. Other related work is found in [6] which considers refutation lower bounds in an OBDD-based proof system for SAT that allows variable order changes. As it is common for lower bounds for QBF-systems based on strategy extraction, see e.g. [1, 2, 3], we do not take into account the cost of reasoning inside **NP** but only measure the hardness that stems from the addition of quantifiers. Thus, our results are incomparable to those of [6].

Structure of the paper:

We start with some preliminaries and necessary background in Section 2. In Section 3, we showcase our approach by separating rectangle decision lists with a single partition change from those with none. The lower bound in this part of the paper is relatively easy with the results from [19] but can be seen as a warm-up for the more technical bounds later on. In Section 4, we develop the general technique for lower bounds for rectangle decision lists with partition changes. Afterwards, in Section 5, we show how to use this technique to prove that there is a strict hierarchy for rectangle decision lists with respect to the number of partition changes. In Section 6, we apply our techniques to QBF proof systems.

2 Preliminaries

Graphs.

We assume that the reader knows some basic graph theory, see e.g. [9]. All graphs in this paper are finite and have no self-loops but in some cases parallel edges. Given a graph $G = (V, E)$ and a partition (V_1, V_2) of V, we denote by $E(V_1, V_2)$ the set of edges in E that have one end-point in V_1 and the other in V_2. By $G[V_1, V_2]$ we denote the bipartite subgraph of G with vertex set V and edge set $E(V_1, V_2)$. We call a graph $G = (V, E)$ a (c, d)-expander if all vertices of G have degree bounded by d and for every vertex set $V' \subseteq V$ with $|V'| \leq |V|/2$ we have $|E(V', V \setminus V')| \geq c|V'|$. If the exact values of c and d are not important or clear from the context, we simply speak of a class of expander graphs. It is well-known that for every $d \geq 3$ there is a constant $c > 0$ such that there is an infinite class of expander graphs, see e.g. [14] for background on this.

Boolean Functions.

We use standard notation for Boolean functions, i.e., functions $f : \{0,1\}^n \to \{0,1\}$ for some $n \in \mathbb{N}$. In particular, as usual, $\wedge$ denotes conjunction, $\vee$ disjunction, and $\oplus$ exclusive or.

An *assignment* of a set X of variables is a mapping $\tau : X \to \{0,1\}$ of variables to truth values. Given assignments $\tau : X \to \{0,1\}$ and $\sigma : Y \to \{0,1\}$ such that X and Y are disjoint, we let $\tau \cup \sigma$ denote the assignment of $X \cup Y$ such that $(\tau \cup \sigma)(x) = \tau(x)$ if $x \in X$ and $(\tau \cup \sigma)(x) = \sigma(x)$ if $x \in Y$. The result of applying an assignment τ to formula φ and propagating constants is denoted $\varphi[\tau]$.

A partition (X_1, X_2) of the variable set X is called *balanced* if $|X_1| = |X_2|$. Unless stated explicitly otherwise, we assume in the remainder of this paper that all variable partitions are balanced. A combinatorial rectangle on variable set X respecting the partition (X_1, X_2) of X is defined to be a function $R(X)$ that can be written as

$$R(X) = R_1(X_1) \wedge R_2(X_2)$$

where R_1 and R_2 are arbitrary Boolean functions depending only on X_1 and X_2, respectively. When writing the value table of R as a matrix where rows are indexed by assignments to X_1 and columns are indexed by assignments to X_2, then, after arranging the rows and columns in such a way that models of $R_1(X_1)$, resp. $R_2(X_2)$, are listed as consecutive rows, resp. columns, the models of R form a rectangle in the matrix, which explains the name of the concept. Due to this representation, we call the models of R_1 and R_2 the rows and columns of R, respectively. In a slight abuse of notation, it is often convenient to identify combinatorial rectangles with their set of models $R^{-1}(1)$ in which case we simply write $R = R_1 \times R_2$. We say that a Boolean function f has the *monochromatic rectangle* R if $R \subseteq f^{-1}(0)$ or $R \subseteq f^{-1}(1)$.

Rectangle Decision Lists.

A *rectangle decision list* L in variables X is a sequence $(R^1, c^1), \ldots, (R^\ell, c^\ell)$ where every R^i is a combinatorial rectangle in X and $c^i \in \{0,1\}$ is a constant. The Boolean function f_L computed by L is defined as follows: given an assignment τ to X, for $i = 1, \ldots, \ell$ we evaluate R^i on τ. Let i^* be the first index such that $R^{i^*}(\tau) = 1$, then the value computed by f_L on τ is c^{i^*}. For this to be well-defined, we assume that R^ℓ is the constant 1-function. We say that ℓ is the length of L and denote it by $|\mathsf{L}|$. If all rectangles R^i respect the partition $\Pi = (X_1, X_2)$ of X, we say that L respects Π. We say that L has k partition changes if there are exactly k indices $i \in [\ell - 1]$ such that the partition of R^i is different from that of R^{i+1}.

Given a rectangle decision list L and a partial assignment τ to a subset Y of X, one can construct from L a rectangle decision list L' that computes the function one gets from f_L by fixing the variables in Y according to τ. The list L' is constructed by simply fixing all rectangles of L according to τ and thus $|\mathsf{L}'| \leq |\mathsf{L}|$.

We generally assume that rectangle decision lists respect a balanced partition Π, but in some cases it will be convenient to allow for some slight imbalance. The following observation shows that this cannot decrease the length of the decision lists by much.

▶ **Observation 1.** *Let L be a rectangle decision list respecting a partition $\Pi = (X_1, X_2)$ such that $|X_1| = |X_2| + k$. Then for every subset $X' \subseteq X_1$ with $|X'| = k/2$ we have that there is a rectangle decision list L' respecting $(X_1 \setminus X', X_2 \cup X')$ computing the same function f_L such that $|\mathsf{L}'| \leq 2^{k/2}|\mathsf{L}|$.*

Proof (sketch). For every rectangle R^i in L and every assignment τ to X', consider the sub-rectangle R^i_τ containing the models that are consistent with τ. Clearly, R^i can be

partitioned into at most $2^{k/2}$ such R^i_τ, so we can substitute (R_i, c^i) by $2^{k/2}$ entries (R^i_τ, c^i). But since the models of (R^i_τ) all take the same values on X', the rectangle R^i_τ can also be rewritten to respect $(X_1 \setminus X', X_2 \cup X')$. ◀

In lower bounds, we often tacitly ignore slight imbalances in partitions due to Observation 1. We will use the following relation between the length of rectangle decision lists and size of monochromatic rectangles from [15] which we slightly reformulate.

▶ **Theorem 2.** *Let f be a function in variables X and let Π be a balanced partition of X. If f has a rectangle decision list with partition Π of length s, then f has a monochromatic rectangle with partition Π of size at least $\frac{1}{4es}2^{|X|}$ where $e \approx 2.718$ denotes Euler's number.*

The inner product function $\mathsf{IP_n}$ in variables $X_n := \{x_1, \dots, x_n\}$ and $Y_n := \{y_1, \dots, y_n\}$ is defined as the inner product of the two-element field, so

$$\mathsf{IP_n}(X_n, Y_n) := \bigoplus_{i \in [n]} x_i \wedge y_n.$$

We use a generalization of the inner product function IP with respect to an underlying graph structure from [19, 13], see also [17, Chapter 5.8]. So let X be a set of Boolean variables and let G be a graph with vertex set X and edge set E. Then we define

$$\mathsf{IP}_G(X) := \bigoplus_{xy \in E} x \wedge y.$$

Note that with this definition $\mathsf{IP} = \mathsf{IP}_{M_n}$ where M_n is a matching with n edges. For the statement of the following lemma, recall that a matching is *induced* if it can be obtained as the subgraph induced by the endpoints of its edges. We will use the following result from [19].

▶ **Lemma 3.** *Let $G = (X, E)$ be a graph with n vertices. Let $\{e_1, \dots, e_m\}$ be an induced matching of G and let (X_1, X_2) be a partition of X such that for every e_i we have $e_i \in E(X_1, X_2)$. Then every monochromatic rectangle of IP_G respecting the partition (X_1, X_2) has size at most 2^{n-m}.*

Ordered Binary Decision Diagrams.

Ordered binary decision diagrams (short OBDDs) are a classical representation of Boolean functions [5]. we only give a very short introduction here; see [25] for a textbook treatment.

Let X be a set of variables and π an ordering of X. An OBDD on variables X with variable order π is defined to be a directed acyclic graph B with one source s and two sinks labeled 0 and 1, called the 0- and 1-sink respectively. All non-sink nodes are labeled with variables from X such that on every path P in B the variables appear in the order π. Moreover, all non-sink nodes have two outgoing edges, one labeled with 0, the other with 1. The size of B, denoted by $|B|$, is defined as the number of nodes in B. For every assignment τ to X, the OBDD B computes a value $B(\tau)$ as follows: starting in the source, we construct a path by taking for every node v labeled by a variable x the edge labeled with $\tau(x)$. We continue until we end up in a sink, and the label of the sink is the value of B on τ denoted by $B(\tau)$. This way B computes a Boolean function and every Boolean function can be computed by an OBDD. The OBDD B is called complete if on every source-sink path P all variables in X appear as node labels. The *width* of a complete OBDD B is defined as the maximal number of nodes that are labeled with the same variable.

▶ **Observation 4.** *There is a polynomial time algorithm that, given an OBDD B, computes an equivalent complete OBDD B'. Moreover, $|B'| \leq (|X|+1)|B|$.*

The algorithm for Observation 4 can e.g. be found in the proof of [25, Lemma 6.2.2].

OBDDs can be combined by arbitrary binary Boolean functions efficiently, see e.g. [25, Chapter 3]:

▶ **Lemma 5.** *Let $f : \{0,1\}^2 \to \{0,1\}$ be a binary Boolean function. Then there is an algorithm that, given two OBDDs B_1 and B_2 with order π over the variable set X, computes in time polynomial in $|B_1| + |B_2|$ an OBDD B with order π such that B computes on every assignment $\tau : X \to \{0,1\}$ the value $B(\tau) := f(B_1(\tau), B_2(\tau))$. In particular, the size of B is polynomial in that of B_1 and B_2.*

OBDDs are well-known to be *canonical* in the sense that, for fixed variable order π, there is a unique minimal representation of any Boolean function f by an OBDD with order π, see again [25, Chapter 3] for background and proofs of this.

▶ **Lemma 6.** *Let f be a Boolean function on variables X and let π be a variable order of X. Then there is a unique OBDD (up to isomorphism) of minimal size with order π computing f. Moreover, given an OBDD with order π representing f, this unique OBDD can be computed in polynomial time. The same is true for complete OBDDs.*

We always assume that OBDDs are minimized with the help of the algorithm of Lemma 6.

▶ **Lemma 7** ([7])**.** *Let B be an OBDD of width w and let Y be a subset of the variables in B. Then there is an OBDD B' of width 2^w that encodes $\exists Y.B$ with the same variable order as B.*

Quantified Boolean Formulas.

We consider quantified Boolean formulas (QBF) in prefix conjunctive normal form (PCNF), i.e., in the form $\Phi = Q_1 x_1 \dots Q_n x_n . C_1 \wedge \dots \wedge C_m$ where the Q_i are quantifiers $\exists$ or $\forall$ and the C_j are clauses. We call $Q_1 x_1 \dots Q_n x_n$ the prefix of Φ and $C_1 \wedge \dots \wedge C_m$ the matrix.

We write $D_\Phi(x_i) = \{x_1, \dots, x_{i-1}\}$ for the set of variables that come before x_i in the quantifier prefix. A variable x_i is *existential* if $Q_i = \exists$, and *universal* if $Q_i = \forall$. We write $\mathrm{var}_\exists(\Phi)$ for the set of existential variables, $\mathrm{var}_\forall(\Phi)$ for the set of universal variables, and $\mathrm{var}(\Phi)$ for the set of all variables occurring in Φ. A *universal strategy* for a PCNF Φ is a family $\vec{f} = \{f_u\}_{u \in \mathrm{var}_\forall(\Phi)}$ of functions $f_u : [\mathrm{var}(\Phi)] \to \{0,1\}$ such that $f_u(\tau) = f_u(\sigma)$ for any assignments τ and σ that agree on $D_\Phi(u)$. If $\vec{f}$ is a universal strategy and $\tau : \mathrm{var}_\exists(\Phi) \to \{0,1\}$ an assignment of existential variables, we write $\tau \cup \vec{f}(\tau)$ for the assignment of $\mathrm{var}(\Phi)$ such that $(\tau \cup \vec{f}(\tau))(x) = \tau(x)$ for existential variables $x \in \mathrm{var}_\exists(\Phi)$ and $(\tau \cup \vec{f}(\tau))(u) = f_u(\tau \cup \vec{f}(\tau))$ for universal variables $u \in \mathrm{var}_\forall(\Phi)$. A universal strategy $\vec{f}$ is a *universal winning strategy* for Φ if $\tau \cup \vec{f}(\tau)$ falsifies the matrix of Φ for every assignment τ of the existential variables. A QBF is *false* if it has a universal winning strategy, and *true* otherwise.

3 Rectangle decision lists with more than one order are exponentially shorter

In this section, we will see that even allowing one partition change in a rectangle decision list allows for exponentially shorter lists. We show this with the following function: let $G = (X, E)$ be a $(c, 3)$-expander graph where the vertex set X consists of the variables of

the function we will construct. Using Vizing's theorem, see [9, Section 5.3], we know that there is a valid edge coloring χ of G with at most 4 colors, say, $\{1, 2, 3, 4\}$. Now set

$$E_1 := \{e \in E \mid \chi(e) \in \{1, 2\}\} \text{ and } E_2 := \{e \in E \mid \chi(e) \in \{3, 4\}\}$$

and define the two graphs $G_1 := (X, E_1)$ and $G_2 := (X, E_2)$. The function we will consider is then

$$f_G := (z \wedge \mathsf{IP}_{G_1}(X)) \vee (\neg z \wedge \mathsf{IP}_{G_2}(X))$$

where z is a fresh variable not in X.

▶ **Proposition 8.** *f_G has a constant size rectangle decision list with one variable partition change.*

Proof. We will start with a simple observation:

▷ Claim 9. There is a variable partition Π_1 such that $\mathsf{IP}_{G_1}(X)$ has a constant length rectangle decision list.

Proof. All vertices in X are incident to at most 2 edges in G_1. So G_1 consists of a collection of cycles and paths. First, consider an order π' of X such that for each component the vertices appear in a contiguous sequence. Let (X'_1, X'_2) be the partition of X that we get by cutting X into two parts in the middle of π'. Then there is a most one component C of G_1 that has vertices in both X'_1 and X'_2. Sorting C in DFS order then yields an order π and a corresponding partition (X_1, X_2) such that there are at most two edges e_1, e_2 between X_1 and X_2 in G_1.

To complete the proof of the claim, we explain now how to compute $\mathsf{IP}_{G_1}(X)$ with the help of few rectangles. Let $e_1 = x_1y_1$ and $e_2 = x_2y_2$. Then, given an assignment τ to X, we can decide if τ satisfies $\mathsf{IP}_{G_1}(X)$ from the values $\mathsf{IP}_{G_1[X_1]}(\tau)$, $\mathsf{IP}_{G_1[X_2]}(\tau)$, $\tau(x_1), \tau(x_2), \tau(y_1), \tau(y_2)$. Note that the set of assignments τ' that coincides with τ on all these values is a rectangle with partition (X_1, X_2) and these rectangles are monochromatic with respect to f_G and partition the space of all assignments to X. Moreover, on half of the 32 resulting rectangles IP_{G_1} evaluate to 1. As a consequence, we can construct a rectangle decision list respecting (X_1, X_2) by iteratively testing for containment in one of the 16 rectangles on which IP_{G_1} evaluates to 1 and reject if the input is in none of them. ◀

With Claim 9, we get partitions Π_1 and Π_2 such that $\mathsf{IP}_{G_1}(X)$ has a constant length decision list with respect to the partition Π_1 and $\mathsf{IP}_{G_2}(X)$ has a constant length decision list with partition Π_2. By construction, in both of the constructed decision list, all outputs but that of the last (default) rectangle are 1. Thus, we get constant size decision lists of $z \wedge \mathsf{IP}_{G_1}(X)$ and $\neg z \wedge \mathsf{IP}_{G_2}(X)$ by adding the variable z and fixing it to 1, respectively 0, in all rectangles. We get a decision list for f_G by deleting the last default line from the list for $\mathsf{IP}_{G_1}(X)$ and concatenating $\mathsf{IP}_{G_2}(X)$ to it. ◀

We next show that if we allow no variable order changes, then rectangle decision lists for f_G are exponentially long.

▶ **Proposition 10.** *For every balanced partition (X_1, X_2), every rectangle decision list for f_G respecting the order (X_1, X_2) has size $2^{\Omega(|X|)}$.*

Proof. Fix a partition (X_1, X_2). Then, because G is an expander graph, there are $\Omega(|X|)$ edges between X_1 and X_2. Call this set of edges E'. Assume w.l.o.g. that $|E' \cap E_1| \geq |E' \cap E_2|$, so $|E' \cap E_1| = \Omega(|X|)$.

Now consider a (X_1, X_2)-rectangle decision list for f_G. By fixing in all rectangles the variable z to 1, we get a (X_1, X_2)-rectangle decision list for $\mathsf{IP}_{G_1}(X)$ without increasing the size of the list.

Now greedily extract from $E' \cap E_1$ an induced matching M. Since the degree of all vertices in G_1 is at most 2, we have $|M| = \Omega(|X|)$. Using Lemma 3, we get that any monochromatic rectangle of $\mathsf{IP}_{G_1}(X)$ respecting the partition has size at most $2^{|X|-\Omega(|X|)}$. Plugging this into Theorem 2, we get

$$2^{|X|-\Omega(|X|)} \geq \frac{1}{4es} 2^{|X|}$$

where s is the length of the rectangle decision list. It follows that $s = 2^{\Omega(|X|)}$ as claimed. ◀

4 Lower Bounds for Lists with Partition Changes

In this section, we will develop a lower bound technique for rectangle decision lists that also works when some partition changes are allowed. The main problem when trying to generalize the proof of Section 3 is that the argument of the proof of Theorem 2 breaks down when allowing even a single partition change. This is because a rectangle R for a partition Π_1 might not contain any big rectangles anymore when considered with respect to a different partition Π_2. To avoid this problem, we develop a new lower bound technique that can play the role of Theorem 2 when some partition changes can occur. Since bounds on rectangle sizes seem to be not quite strong enough to show such a lower bound, we base it on the more restrictive notion *discrepancy* which we introduce first.

4.1 Discrepancy

Discrepancy of Boolean functions is a well-known tool in communication complexity, in particular in randomized models, see e.g. [18, Chapter 3]. Here we will consider a variant of discrepancy with respect to different partitions of the variables. To this end, we make some definitions. Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. As usual, we define the discrepancy of a rectangle R with respect to the function f and a probability distribution μ of inputs to f as

$$\mathrm{Disc}_\mu(R, f) := \left| \Pr_\mu \left(f(x) = 1 \text{ and } x \in R\right) - \Pr_\mu \left(f(x) = 0 \text{ and } x \in R\right) \right|$$

where x is chosen randomly according to the distribution μ. Now we define the discrepancy of f for the partition Π and the distribution μ as

$$\mathrm{Disc}_\mu(f, \Pi) := \max_R \left(\mathrm{Disc}_\mu(R, f)\right)$$

where the maximum is over all rectangles respecting the partition Π. We will exclusively work with the uniform distribution so we leave out the subscript μ and get

$$\mathrm{Disc}(R, f) = \left| |f^{-1}(1) \cap R| - |f^{-1}(0) \cap R| \right| / 2^n$$

and

$$\mathrm{Disc}(f, \Pi) := \max_R \left(\mathrm{Disc}(R, f)\right).$$

We now give a bound on the discrepancy of the graph inner product function IP_G. It turns out that, as for the size of monochromatic rectangles in Lemma 3, the discrepancy is bounded exponentially in the size of an induced matching between the two sides of the considered partition.

▶ **Lemma 11.** *Let $G = (X, E)$ be a graph with n vertices. Let $\{e_1, \dots, e_m\}$ be an induced matching of G and let $\Pi = (X_1, X_2)$ be a partition of X such that for every e_i one of the end points is in X_1 and one is in X_2. Then*

$$\mathrm{Disc}(\mathsf{IP}_G, \Pi) \leq 2^{-m/2}.$$

The proof of Lemma 11 is very similar to that of Lemma 3, so we sketch it in Appendix A.

4.2 Rectangles in Partial Functions

It will be useful to consider partial functions which we model as functions $f : \{0,1\}^n \to \{0,1,*\}$. Here, as usual, 1 and 0 stand for true and false, respectively, while $*$ denotes inputs on which f is undefined. We say that a Boolean function f' is *consistent* with f if $f(a) = f'(a)$ whenever $f(a) \in \{0,1\}$. Essentially, the Boolean functions consistent with f are all functions we can get from f by defining all undefined values. As a special case, we say that a rectangle R is consistent with f if $f(R) \subseteq \{0,*\}$ or $f(R) \subseteq \{1,*\}$. Note that we assume in this that R is a Boolean function and in particular is *not* partial.

We will use the following simple observation.

▶ **Lemma 12.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function with discrepancy d. Let $\tilde{f}$ be a partial Boolean function with at most u undefined values such that f is consistent with $\tilde{f}$. Then any rectangle consistent with $\tilde{f}$ has size at most $2^n d + 2u$.*

Proof. Let R be a rectangle consistent with $\tilde{f}$. Assume w.l.o.g. that $f(R) \subseteq \{1,*\}$. Then we bound the size of R as follows:

$$\begin{aligned} |R| &= |f^{-1}(1) \cap R| + |f^{-1}(0) \cap R| \\ &\leq 2^n \mathrm{Disc}(R, f) + 2|f^{-1}(0) \cap R| \\ &\leq 2^n d + 2u \end{aligned}$$

The first step is true because by definition of discrepancy we have $|f^{-1}(1) \cap R| - |f^{-1}(0) \cap R| \leq 2^n \mathrm{Disc}(R, f)$. In the second step we use that, since R is consistent with $\tilde{f}$, all values in $|f^{-1}(0) \cap R|$ must be undefined in $\tilde{f}$. ◀

4.3 Lower Bounds for Functions with Small Discrepancy

We can now formulate and prove the main result of of this section which shows that discrepancy can be used to show lower bounds for decision lists with partition changes.

▶ **Proposition 13.** *Let $f : \{0,1\}^n \to \{0,1\}$ be computed by a rectangle decision list with $k-1$ partition changes using the k partitions $\Pi_1, \dots, \Pi_k$. Assume that for every $i \in [k]$ we have $\mathrm{Disc}(f, \Pi_i) \leq 2^{-cn}$. Then the length of the rectangle decision list is at least $\Omega(2^{\frac{cn}{(k-1)2^{k-1}}})$.*

Proof. Let f be computed by the decision list $(R_1, c_1), \dots, (R_t, c_t)$. Assume that the partitions $\Pi_1, \dots, \Pi_k$ are used in that order in the decision list.

We will iteratively construct a big rectangle $\bar{R}$ that is consistent with a partial function $\bar{f}$ that is consistent with f and has relatively few unknown values. The idea is similar to

that in [15] but more complicated due to the partition changes. We think of the rectangles $R_1, \ldots, R_t$ as organized in *phases*: rectangle R_i is in phase j if it is with respect to the partition Π_j.

We construct a rectangle $\bar{R}_i$ iteratively for all $i \in [t]$ such that

$$\left(\bigcup_{j \le i : R_i, R_j \text{ in same phase}} R_j \right) \cap \bar{R}_i = \emptyset. \tag{1}$$

Moreover, we construct for every phase $j \in [k]$ a partial function f^j that is consistent with f. We start by setting $f^1 := f$.

We now construct the $\bar{R}_i$. If R_i is the first rectangle in phase j, we check if $|R_i| > (2t+1)^{j-1} 2^{n - \frac{cn}{2^{j-1}}}$. If so, we set $\bar{R} = R_i$, $\bar{f} = \bar{f}_j$ and stop. Otherwise, if R_i has less than $(2t+1)^{j-1} 2^{\left(n - \frac{cn}{2^{j-1}}\right)/2}$ rows, we set $\bar{R}_i$ to be the rectangle we get from $\{0,1\}^n$ by deleting the rows of R_i. Otherwise, R_i has less than $(2t+1)^{j-1} 2^{\left(n - \frac{cn}{2^{j-1}}\right)/2}$ columns which we then delete from $\{0,1\}^n$ to get $\bar{R}_i$.

By construction, the property (1) is satisfied: the only rectangle R_j that is in the same phase as R_i and has $j \le i$ is in fact R_i. But since we deleted either all rows or all columns of R_i in the construction of $\bar{R}_i$, the intersection is empty.

If R_i is not the first rectangle in the phase, then we have already constructed $\bar{R}_{i-1}$ which has the same partition Π_j as R_i. Note that $R_i \cap \bar{R}_{i-1}$ is a rectangle. We proceed similarly to before but consider $R_i \cap \bar{R}_{i-1}$ instead of R_i: if $|R_i \cap \bar{R}_{i-1}| > (2t+1)^{j-1} 2^{n - \frac{cn}{2^{j-1}}}$, we set $\bar{R} = R_i \cap \bar{R}_{i-1}$ and $\bar{f} := f_j$. Otherwise, we delete the lines or columns of $R_i \cap \bar{R}_{i-1}$ from $\bar{R}_{i-1}$, whichever are less, to construct $\bar{R}_i$.

Finally, we define f_j for $j > 1$ inductively as the partial function we get from f_{j-1} by making all entries of all lines and columns that have ever been deleted in the construction of the $\bar{R}_i$ in an earlier phase take the value $*$. Obviously, f and f_j are consistent for all $j \in [k]$. Let us analyze how many inputs to $\bar{f}_j$ evaluate to $*$. In the worst case, we have deleted columns or rows in t steps of the construction. The undefined values in f_j are from the deletions of rows and columns in phases $1, \ldots, j-1$. The highest number of rows or columns deleted for one such R_i is in phase $j-1$ and is $(2t+1)^{j-2} 2^{\left(n - \frac{cn}{2^{j-2}}\right)/2}$ there, so in that step at most $2^{n/2}(2t+1)^{j-2} 2^{\left(n - \frac{cn}{2^{j-2}}\right)/2} = (2t+1)^{j-2} 2^{n - \frac{cn}{2^{j-1}}}$ values have been set to $*$. So f_j has at most $u := t(2t+1)^{j-2} 2^{n - \frac{cn}{2^{j-1}}}$ undefined values.

Now assume that $\bar{R}$ and $\bar{f}$ are assigned in phase j which happens by construction if we have that the first rectangle R_i in phase j satisfies $|R_i| > (2t+1)^{j-1} 2^{n - \frac{cn}{2^{j-1}}}$ or a later rectangle R_i in the phase j satisfies $|R_i \cap \bar{R}_{i-1}| > (2t+1)^{j-1} 2^{n - \frac{cn}{2^{j-1}}}$. Then in any case we get that

$$|\bar{R}| > (2t+1)^{j-1} 2^{n - \frac{cn}{2^{j-1}}} \ge 2^{n-cn} + 2u \ge 2^n \mathrm{Disc}(f, \Pi_j) + 2u. \tag{2}$$

Now assume w.l.o.g. that $c_i = 1$, i.e., the rectangle R_i assigns the value 1 to all inputs that end up at this test during the evaluation of the decision list and satisfy R_i. We claim that the function

$$f(x,y) := \begin{cases} 1, & \text{if } (x,y) \in \bar{R} \\ *, & \text{otherwise} \end{cases}$$

is consistent with $\bar{f}$. Assume this were not the case. Then there must be an element $(x,y) \in \bar{R}$ on which $\bar{f}(x,y) = 0$, so this value must be assigned in the test for a rectangle $R_{i'}$ that is tested before R_i. Assume first that this happens in the same phase j. Then $\bar{R} \subseteq \bar{R}_i \subseteq \bar{R}_{i'}$.

But by (1), the rectangle $R_{i'}$ can then not be responsible for assigning any value in $\bar{R}$. So the rectangle $R_{i'}$ must be in an earlier phase $j' < j$. But note that when constructing $\bar{R}_{i'}$, we have by (1) deleted all entries of $R_{i'}$. Thus, in $\bar{f}_i$ we have the value $*$ for the corresponding inputs which is a contradiction to $\bar{f}$ taking the value 0 there. But then we get a contradiction with Lemma 12 and (2), because the rectangle $\bar{f}$ is too big. It follows that $\bar{R}$ and $\bar{f}$ can never be assigned in the construction.

As a consequence, the construction of the $\bar{R}_i$ goes through to the end. Reasoning as before, all non-*-values in $\bar{R}_t$ must have the same value. As a consequence, we get with Lemma 12 that

$$|\bar{R}_t| \leq 2^n \mathrm{Disc}(f, \Pi_k) + 2u \leq 2^{n-nc} + 2t(2t+1)^{k-2}2^{n-\frac{cn}{2^{k-1}}}.$$

Reasoning as for the number of unknown values in f_j before, we also know that $\bar{R}_t$ is constructed from $\{0,1\}^n$ by deleting in at most t rounds at most $(2t+1)^{k-2}2^{n-\frac{cn}{2^{k-1}}}$ values each. From this we get

$$|\bar{R}_t| \geq 2^n - t(2t+1)^{k-2}2^{n-\frac{cn}{2^{k-1}}}.$$

Putting this together, it follows that

$$\begin{aligned}
&2^n - t(2t+1)^{k-2}2^{n-\frac{cn}{2^{k-1}}} \leq 2^{n-nc} + 2t(2t+1)^{k-2}2^{n-\frac{cn}{2^{k-1}}} \\
\Rightarrow &\frac{2^n - 2^{n-nc}}{2^{n-\frac{cn}{2^{k-1}}}} \leq (3t)^{k-1} \\
\Rightarrow &\frac{1}{3}2^{\frac{cn}{(k-1)2^{k-1}}} - \frac{1}{3} \leq t.
\end{aligned}$$

◀

5 Separating the Hierarchy for Partition Changes

In this section, we will use Proposition 13 to show that increasing the number k of partition changes allowed in a rectangle decision list makes them exponentially more succinct. To this end, we will need functions that have small discrepancy when considered for any set of k partitions but as soon as we allow $k+1$ partitions, they become easy. The functions that we will consider are constructed from such of the form IP_G for carefully chosen graphs.

To construct these graphs, let us make some additional definitions. Given two graphs G_1 and G_2 on the same vertex set V, denote by $G_1 + G_2$ the graph that we get by taking the union of the edge sets of G_1 and G_2.

▶ **Proposition 14.** *For every $k \in \mathbb{N}$ there is a constant c such that for every $n \in \mathbb{N}$ large enough, there are $k+1$ graphs $G_1, \ldots, G_{k+1}$ on a vertex set V of size n with the following properties:*

- *no graph G_i has any parallel edges,*
- *every vertex in every G_i has degree at most 2, and*
- *for every k partitions $(V_1^1, V_2^1), \ldots, (V_1^k, V_2^k)$ of V there is a graph G_i such that for every $j \in [k]$ the bipartite graph $G_i[V_1^j, V_2^j]$ has an induced matching of size cn.*

Proof. Let V be a vertex set that is big enough. We choose all graphs G_i randomly in the so-called configuration model which we briefly describe next. Let d be even. Then a random d-regular multi-graph in the configuration model is chosen by first considering the set $W := V \times [d]$ and choosing a random matching of W which we call the edges of the

configuration. We then get a multigraph by projecting the edges of the configuration to V. It is known that almost every 4-regular multi-graph chosen in the configuration model is an expander, see [4], i.e., with probability going to 1 for $n \to \infty$ a graph chosen in this model is an $(\alpha, 4)$-expander for some constant $\alpha > 0$.

An alternative way to construct a random 4-regular graph is to choose two random 2-regular graphs G_1, G_2 in the configuration model and take their sum $G_1 + G_2$; call this the *sum model*. It is known that the configuration model and the sum model are *contiguous*, which intuitively means that both of them have asymptotically the same properties almost surely, see [16, Chapter 9] for exact definitions and details on this. In particular, since graphs chosen in the configuration model are $(\alpha, 4)$-expanders for some constant $\alpha > 0$ with probability going to 1 for $n \to \infty$, the same is true for graphs chosen randomly in the sum model. From this convergence, it follows that there is a constant n_0 such that for a random graph with at least n_0 vertices chosen in the sum model the probability of not being an $(\alpha, 4)$-expander is bounded by the constant $\frac{1}{10(k+1)^2}$.

We now choose the graphs $G_1, \dots, G_{k+1}$ that were promised in the statement of the proposition as random 2-regular graphs in the configuration model with more than n_0 vertices. Applying the union bound, we get the following bound on the sum graphs $G_i + G_j$:

$$\begin{aligned}&\Pr(\exists i, j \in [k+1], i \neq j : G_i + G_j \text{ is not an } (\alpha, 4)\text{-expander})\\ \leq &\sum_{i,j \in [k+1], i \neq j} \Pr(G_i + G_j \text{ is not an } (\alpha, 4)\text{-expander}) \leq (k+1)^2 \frac{1}{10(k+1)^2} = \frac{1}{10}.\end{aligned}$$

So with probability at least .9 all sums $G_i + G_j$ with $i, j \in [k+1], i \neq j$ are $(\alpha, 4)$-expanders. We assume in the remainder that this is the case for our graphs G_i.

By definition, the degree bound of the G_i is clear. We next show the third item of the claim. We first show that there is a G_i such that for every $j \in [k]$ the induced bipartite graph $G_i[V_1^j, V_2^j]$ has at least $\alpha|V|/3$ edges. By way of contradiction, assume that this were not the case, so for every G_i there is a (V_1^j, V_2^j) such that $G_i[V_1^j, V_2^j]$ has less than $\alpha|V|/3$ edges. Then there is a j such that there are two graphs G_{i_1}, G_{i_2} such that $G_{i_1}[V_1^j, V_2^j]$ and $G_{i_2}[V_1^j, V_2^j]$ have at most $\alpha|V|/3$ edges which contradicts $G_{i_1} + G_{i_2}$ being an $(\alpha, 4)$-expander. So there is a graph G_i such that for every j we have that $G_i[V_1^j, V_2^j]$ has at least $\alpha|V|/3$ edges. It only remains to greedily extract an induced matching from each $G_i[V_1^j, V_2^j]$ which due to the fact that G_i has bounded degree is of size linear in $\alpha|V|/3$.

The only problem we still have to take care of is that the graphs G_i might have parallel edges. Because of the degree bound between each pair of vertices, there are at most two parallel edges. Deleting one of them reduces the number of edges in the induced matchings by at most half which completes the proof. ◀

▶ **Theorem 15.** *For every constant $k \in \mathbb{N}$ and $n \in \mathbb{N}$ sufficiently big, there is a Boolean function $f_{n,k}$ in $n + k + 1$ variables such that*

- *$f_{n,k}$ can be computed by a rectangle decision list of length $O(k)$ with k partition changes, but*
- *any rectangle decision list with $k - 1$ partition changes computing $f_{n,k}$ has length $2^{\Omega(n)}$.*

Proof. Let $G_1, \dots, G_{k+1}$ be graphs on a vertex set X with the properties guaranteed by Proposition 14. The function we consider is

$$f(X, Y) := \bigvee_{i \in [k+1]} y_i \wedge \mathsf{IP}_{G_i}(X)$$

where $Y := \{y_1, \dots, y_{k+1}\}$.

Reasoning as in Claim 9, we see that for all $i \in [k+1]$ there is a variable partition such that $\mathsf{IP}_{G_i}(X)$ and thus $y_i \wedge \mathsf{IP}_{G_i}(X)$ has a constant length rectangle decision list. Concatenating these decision lists yields one of length $O(k)$ computing f. This proves the first claim.

For the second claim, consider any rectangle decision list with at most $k-1$ partition changes computing f. Let $(X_1^1, X_2^1), \dots, (X_1^k, X_2^k)$ be the partitions used. Then, by Proposition 14, there is an $i \in [k+1]$ such that for every $j \in [k]$ the graph $G_i[X_1^j, X_2^j]$ has an induced matching of size cn for some constant c only depending on k. Fixing the variables $y_1, \dots, y_{k+1}$ in the right way, we get from the rectangle decision list for f one for IP_{G_i} of the same length and with the same partitions. By Lemma 11 we get that

$$\mathrm{Disc}(f, (X_1^j, X_2^j)) \leq 2^{-cn/2}.$$

Plugging this into Proposition 13, we get that the rectangle decision list for IP_{G_i} and thus that for f must be of length $2^{\Omega(n)}$, which completes the proof. ◀

6 Application to QBF Proof Complexity

In this section, we will present a consequence of the results developed above for certain QBF proof systems. We consider an extension of the proof systems introduced in [19] that models the behavior of certain OBDD-based QBF-solvers. We show that in this setting there is a similar hierarchy as in Theorem 15 for refutations in an extension of these proof systems that allows changing variable orders in derivations.

6.1 OBDD-Refutations with Reordering

In this section, we introduce the model that we will consider in the remainder of this paper. We work with the proof system introduced in [19] that uses OBDDs as lines in derivations as follows: let $\Phi = Q_1 x_1 \dots Q_n x_n . C_1 \wedge \dots \wedge C_m$ be a PCNF. A derivation of an OBDD L_k from Φ is a sequence $L_1, \dots, L_s$ of OBDDs such that for all $i \in [m]$ the OBDD L_i is equivalent to the clause C_i and for $i > m$ the OBDD L_i is derived by one of the following rules:

1. **conjunction** ($\wedge$): L_i represents $L_j \wedge L'_j$ for $j, j' < i$.
2. **projection** ($\exists$): L_i represents $\exists x . L_j$ for some $x \in \mathrm{var}(L_j)$ and $j < i$.
3. **entailment** ($\models$): L_i is entailed by $L_{i_1}, \dots, L_{i_r}$, for $i_1, \dots i_r < i$.
4. **universal reduction** ($\forall$): L_i represents $L_j[u/c]$, where $j < i$, u is a universally quantified variable that is rightmost among variables in L_j and $c \in \{0, 1\}$.

Here, $L_j[u/c]$ denotes the OBDD obtained from L_j by removing each node labeled with variable u and rerouting all incoming edges to its neighbor along the c-labeled edge (effectively substituting c for u).

In [19] it is explained how the above system corresponds to practical solvers like the QBDD-solver of [20] in the sense that lower bounds for the proof system give lower bounds for the runtime of the solver. In [19], it is assumed that the variable order of all OBDDs in a derivation is the same—which corresponds to the fact that the QBDD-solver uses a fixed variable order in each run. Since we want to model QBF-solvers that are allowed to change variable orders, we introduce a new rule:

5. **reordering** (r): L_i is equivalent to a line L_j where $j < i$ but has a different variable order.

We assume that, whenever applying the rules 1.–4., all OBDDs mentioned in those rules have the same variable order. As a consequence, whenever we want to change the variable order in a derivation, we have to do so by explicitly using rule 5. In the following, we assume only a bounded number of different variable orders are used in derivations. To this end, we make the following definition: Let $L_1, \ldots, L_s$ be a derivation. We say that it has k variable order changes if there are k indices $i \in [s-1]$ such that the variable order of L_i is different from that of L_{i+1}. We denote by $r_{\leq k}$ the reordering rule from above restricted to the case in which there are at most k variable changes allowed in a derivation.

The *size* of a derivation $L_1, \ldots, L_s$ is $\sum_{i \in s} |L_s|$, i.e., the sum of the sizes of all occurring OBDDs. The derivation is called a *refutation* if $L_s \equiv 0$. For every subset of rules $S \subseteq \{\wedge, \exists, \forall, r, r_k\}$, the proof system OBDD($S$) is the restriction of the proof system from above to the rules in S. Of particular interest in [19] were OBDD($\wedge, \exists, \forall$) and OBDD($\wedge, \exists, \forall, \models$): the former is a formalization of the practical solver QBDD [20] while the latter is the strengthening that allows "free" reasoning in NP and for which any lower bound is thus due to genuine hardness due to adding quantification. We here will study the fragments OBDD($\wedge, \exists, \forall, r_{\leq k}$) and OBDD($\wedge, \exists, \forall, \models, r_{\leq k}$) which add up to k variable order changes in derivations. Note that it was shown in [19] that the systems OBDD($\wedge, \exists, \forall$) and OBDD($\wedge, \exists, \forall, \models$) are sound, so in particular only false PCNF allow refutations, and it is easy to see that this remains true when allowing changes of variable orders.

6.2 Statement of the Hierarchy for OBDD-Refutations and the Separating Functions

The main aim of this section is the following hierarchy with respect to the number of variable order changes.

▶ **Theorem 16.** *For every constant $k \in \mathbb{N}$ and $n \in \mathbb{N}$ sufficiently big, there is a Boolean function Φ in $O(n)$ variables and size $O(kn)$ such that*

- *Φ has an* OBDD($\wedge, \exists, \forall, r_{\leq k}$)*-refutation of polynomial size, but*
- *every* OBDD($\wedge, \exists, \forall, \models, r_{\leq k-1}$)*-refutation of Φ has length $2^{\Omega(n)}$.*

We remark that in Theorem 16 the lower bound is for the stronger model with entailment ($\models$) while the upper bound does not use it. To prove Theorem 16, we again consider the graphs $G_1, \ldots, G_{k+1}$ from Proposition 14. We here use them to define a separating function for OBDD-refutations with an increasing number of variable order changes. Remember that X is the underlying vertex set of all the G_i and thus the variable set of the IP_{G_i}. Let $y_1, \ldots, y_{k+1}, z$ be additional variables not appearing in X. We use the following observation as a building block.

▶ **Observation 17.** *There is a constant w such that for every $i \in [k+1]$, there is a CNF-encoding ϕ_i of $y_i \vee (\mathsf{IP}_{G_i} \neq z)$ with auxiliary variables and an order π_i of the variables of ϕ_i such that every sub-formula of ϕ_i has an OBDD-representation of width at most w and with order π_i.*

The proof of Observation 17 is not hard but somewhat tedious, so we defer it to Appendix B.

For every $i \in [k+1]$, let Z_i denote the set of auxiliary variables used in the construction of ϕ_i and let $Z := Z_1 \cup \ldots \cup Z_{k+1}$. Assume that for all $i, i' \in [k+1]$ with $i \neq i'$, the sets Z_i and $Z_{i'}$ are disjoint. We now define the matrix of the PCNF we want to construct as

$$M(X, y_1, \ldots, y_{k+1}, z, Z) := (\neg y_1 \vee \ldots \vee \neg y_{k+1}) \wedge \bigwedge_{i \in [k+1]} \phi_i.$$

Then the separating formula Φ is defined as

$\exists y_1 \dots y_{k+1} \exists X \forall z \exists Z. M(X, y_1, \dots, y_{k+1}, z, Z)$.

We will use Φ to prove Theorem 16 in the next two sections.

6.3 The Upper Bound

In this section, we show the upper bound of Theorem 16. We start with the following Lemma.

▶ **Lemma 18.** *There is a constant w such that for every i, there exists a variable order π_i such that ϕ_i has a polynomial length* OBDD$(\wedge, \exists, \forall)$*-derivation with variable order π_i from the clauses of ϕ_i. Moreover, the width of all intermediate results of this refutations represented as OBDDs with order π_i is at most w.*

Proof. We use the variable order π_i from Observation 17. We first represent all clauses of ϕ_i by an OBDD with this variable order. Then we conjoin these OBDDs of all clauses in an arbitrary order. Since we know that all sub-formulas of ϕ_i have an OBDD-representation of width w, we know that all intermediate results we get have width at most w and thus polynomial size. The same is true for the end result, the representation of ϕ_i. ◀

▶ **Lemma 19.** *For every i, there is a variable order π_i such that there is an* OBDD$(\wedge, \exists, \forall)$*-derivation of y_i from Φ with the order π_i and of polynomial size.*

Proof. We first use Lemma 18 to derive an OBDD-representation of ϕ_i of width w. In the next step, we project away the variables in Z_i. The result of this is an OBDD B representing $y_i \vee (\mathsf{IP}_{G_i} \neq z)$. Since ϕ_i has width at most w, from Lemma 7, we get that B and all intermediate results have width at most 2^w and thus size polynomial in n.

In the next step, we apply universal reduction twice on B to get the OBDDs $B[z/0] \equiv y_i \vee \mathsf{IP}_{G_i}$ and $B[z/1] \equiv y_i \vee \neg\mathsf{IP}_{G_i}$. Since universal reduction does not increase the size of an OBDD, the overall derivation has polynomial size. We then use one conjoining step to compute $B[z/0] \wedge B[z/1] \equiv (y_i \vee \mathsf{IP}_{G_i}) \wedge (y_i \vee \neg\mathsf{IP}_{G_i}) \equiv y_i$. ◀

We are now ready to prove the upper bound of Theorem 16.

▶ **Lemma 20.** *There is a polynomial size* OBDD$(\wedge, \exists, \forall, r_{\leq k})$*-refutation of Φ.*

Proof. For every $i \in [k+1]$, we use a variable order π_i to derive y_i from Φ in polynomial size with the help of Lemma 19. Assume that y_{k+1} was the last of the y_i derived this way. Note every y_i has a constant size OBDD-representation with order π_{k+1}, so we can reorder the variables of the representations of y_i to π_{k+1}. We then conjoin all the representations of the y_i with that of $\neg y_1 \vee \ldots \vee \neg y_{k+1}$ to derive 0 as desired. Overall, we make a polynomial number of derivation steps and all intermediate results have polynomial size, so the overall size of the refutation is polynomial in n, as desired. Finally, remark that we have to change the variable order only k times. ◀

6.4 The Lower Bound

In this section, we will show that if we allow an OBDD-refutation of Φ to only make $k-1$ variable order changes, the refutation must be of exponential size. Our result uses a variant of strategy extraction from [19], see also [1, 2, 3] for more on this technique for lower bounds in the QBF-setting.

▶ **Proposition 21.** *Let Ψ be a PCNF. If Ψ has a* OBDD$(\wedge, \exists, \forall, r_{\leq k-1})$*-refutation of size s, then there is a universal winning strategy for Ψ whose functions can all be represented by rectangle decision lists of length s^2 with $k-1$ partition changes.*

We remark that Proposition 21 in [19] was only shown for the case without any partition changes. However, it is immediate from the proof that variable order changes translate directly into partition changes in the construction. For completeness, we show Proposition 21 in Appendix C.

▶ **Lemma 22.** *Every* OBDD$(\wedge, \exists, \forall, \models, r_{k-1})$*-refutation of Φ has size $2^{\Omega(n)}$.*

Proof. From an OBDD$(\wedge, \exists, \forall, r_{\leq k-1})$-refutation of size s, by Proposition 21 we get a rectangle decision list L of length s^2 that represents the winning strategy f_z for the only universal variable z in Φ. Note that f_z depends only on $D_\Phi(z) = \{y_1, \ldots, y_{k+1}\} \cup X$.

Since we assume that the refutation is a OBDD$(\wedge, \exists, \forall, \models, r_{k-1})$-refutation, by Proposition 21 there are at most $k-1$ partition changes in the rectangle decision list L for f_z; let $(V_1^1, V_2^1), \ldots, (V_1^k, V_2^k)$ denote the partitions. By Proposition 14, there is a graph G_i such that for every $j \in [k]$ we have that the graph $G[V_1^j, V_2^j]$ has an induced matching of size αn for some constant α.

Now fix some of the variables as follows: set $y_i = 0$ and all other $y_j = 1$. Denote the corresponding restriction of f_z by f'_z. By definition of winning strategies, we have that f'_z is such that for every assignment τ to $X \cup Z$, the formula we get from $M(X, y_1, \ldots, y_{k+1}, z, Z)$ by fixing the y_j as described evaluates to 0 under the assignment $\tau \cup f_z(\tau)$. But the assignment to the y_j satisfies the clause $\neg y_1 \wedge \ldots \wedge \neg y_{k+1}$ and all ϕ_j except ϕ_i, so what remains of the matrix is $\mathsf{IP}_{G_i} \neq z$. In order to not satisfy this, f'_z must thus compute IP_{G_i}.

Now from L we get a rectangle decision list L' by restricting in all rectangles the values of the y_j according to the way we fixed them above and then projecting away the corresponding variables. This does not increase the length of the decision list and keeps the partitions roughly balanced. Call the corresponding partitions $(\bar{V}_1^1, \bar{V}_2^1), \ldots, (\bar{V}_1^k, \bar{V}_2^k)$, then by Lemma 11,

$$\mathrm{Disc}(\mathsf{IP}_{G_i}) \leq 2^{-\alpha' n}$$

for some constant $\alpha' > 0$. By Proposition 13, we get that L' and thus also L has length $2^{\Omega(n)}$ as claimed. ◀

7 Conclusion

We have shown that rectangle decision lists in general become far more succinct when the underlying partition is allowed to change along the list. More precisely, whenever one more partition change is allowed, there are functions that can be expressed with exponentially shorter lists. We have also lifted this to OBDD-refutations for QBF. It follows that in principle one can construct OBDD-based QBF-solvers that are exponentially faster than the QBDD-solver from [20] by changing the variable order in a run. However, in practice there is of course the question of when the variable order should be changed and which order should be taken. A first natural idea in this direction might be to adapt the known reordering heuristics for OBDD construction, see e.g. [25, Chapter 5.8] for an overview.

It would be interesting to see if one can still show lower bounds for rectangle decision lists when no bound on the number k of partition changes is assumed. Note that the lower bound in Proposition 13 breaks down as soon as k is logarithmic in n, so the techniques presented here do not directly help in this setting. Considering that it is known how to prove lower bounds in the related model of multi-partition communication complexity [10], such bounds might be within reach of current lower bound techniques.

References

1 Valeriy Balabanov and Jie-Hong R. Jiang. Unified QBF certification and its applications. *Formal Methods Syst. Des.*, 41(1):45–65, 2012. doi:10.1007/s10703-012-0152-6.

2 Olaf Beyersdorff, Ilario Bonacina, Leroy Chew, and Ján Pich. Frege systems for quantified boolean logic. *J. ACM*, 67(2):9:1–9:36, 2020. doi:10.1145/3381881.

3 Olaf Beyersdorff, Luke Hinde, and Ján Pich. Reasons for hardness in QBF proof systems. *ACM Trans. Comput. Theory*, 12(2):10:1–10:27, 2020. doi:10.1145/3378665.

4 Béla Bollobás. The isoperimetric number of random regular graphs. *Eur. J. Comb.*, 9(3):241–244, 1988. doi:10.1016/S0195-6698(88)80014-3.

5 Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Computers*, 35(8):677–691, 1986. doi:10.1109/TC.1986.1676819.

6 Sam Buss, Dmitry Itsykson, Alexander Knop, and Dmitry Sokolov. Reordering rule makes OBDD proof systems stronger. In Rocco A. Servedio, editor, *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, volume 102 of *LIPIcs*, pages 16:1–16:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

7 Florent Capelli and Stefan Mengel. Tractable QBF by knowledge compilation. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, volume 126 of *LIPIcs*, pages 18:1–18:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.STACS.2019.18.

8 Arkadev Chattopadhyay, Meena Mahajan, Nikhil S. Mande, and Nitin Saurabh. Lower bounds for linear decision lists. *Chic. J. Theor. Comput. Sci.*, 2020, 2020. URL: http://cjtcs.cs.uchicago.edu/articles/2020/1/contents.html.

9 Reinhard Diestel. *Graph Theory, 5th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2016.

10 Pavol Duris, Juraj Hromkovic, Stasys Jukna, Martin Sauerhoff, and Georg Schnitger. On multi-partition communication complexity. *Inf. Comput.*, 194(1):49–75, 2004. doi:10.1016/j.ic.2004.05.002.

11 Andrea Ferrara, Guoqiang Pan, and Moshe Y. Vardi. Treewidth in verification: Local vs. global. In Geoff Sutcliffe and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, 12th International Conference, LPAR 2005, Montego Bay, Jamaica, December 2-6, 2005, Proceedings*, volume 3835 of *Lecture Notes in Computer Science*, pages 489–503. Springer, 2005. doi:10.1007/11591191_34.

12 Mika Göös, Pritish Kamath, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for P^NP. *Comput. Complex.*, 28(1):113–144, 2019. doi:10.1007/s00037-018-0175-5.

13 T.P. Hayes. Separating the k-party communication hierarchy: an application of the Zarankiewicz problem. Available at http://www.cs.unm.edu/ hayes/papers/, 2001.

14 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

15 Russell Impagliazzo and Ryan Williams. Communication complexity with synchronized clocks. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*, pages 259–269. IEEE Computer Society, 2010. doi:10.1109/CCC.2010.32.

16 Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random graphs*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 2000. doi:10.1002/9781118032718.

17 Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-24508-4.

18 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

19 Stefan Mengel and Friedrich Slivovsky. Proof complexity of symbolic QBF reasoning. In Chu-Min Li and Felip Manyà, editors, *Theory and Applications of Satisfiability Testing - SAT 2021 - 24th International Conference, Barcelona, Spain, July 5-9, 2021, Proceedings*,

volume 12831 of *Lecture Notes in Computer Science*, pages 399–416. Springer, 2021. doi:10.1007/978-3-030-80223-3_28.

20 Guoqiang Pan and Moshe Y. Vardi. Symbolic decision procedures for QBF. In Mark Wallace, editor, *Principles and Practice of Constraint Programming - CP 2004, 10th International Conference, CP 2004, Toronto, Canada, September 27 - October 1, 2004, Proceedings*, volume 3258 of *Lecture Notes in Computer Science*, pages 453–467. Springer, 2004. doi:10.1007/978-3-540-30201-8_34.

21 Periklis A. Papakonstantinou, Dominik Scheder, and Hao Song. Overlays and limited memory communication. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 298–308. IEEE Computer Society, 2014. doi:10.1109/CCC.2014.37.

22 Ronald L. Rivest. Learning decision lists. *Mach. Learn.*, 2(3):229–246, 1987. doi:10.1007/BF00058680.

23 Fabio Somenzi. CUDD: CU decision diagram package-release 2.4. 0. *University of Colorado at Boulder*, 2009.

24 György Turán and Farrokh Vatan. Linear decision lists and partitioning algorithms. In *Foundations of Computational Mathematics: Selected Papers of a Conference Held at Rio de Janeiro, January 1997*, page 414. Springer Science & Business Media, 2012.

25 Ingo Wegener. *Branching Programs and Binary Decision Diagrams*. SIAM, 2000. URL: http://ls2-www.cs.uni-dortmund.de/monographs/bdd/.

A Proof of Lemma 11

When fixing all variables not incident to any edge e_i according to a partial assignment τ, we get a function f_τ in the variables $\{x_1, \dots, x_m\} \cup \{y_1, \dots, y_m\}$ where we assume for every $i \in [m]$ that $e_i = x_i y_i$. As in the proof of Lemma 3 in [19], f_τ is essentially IP_m up to flipping the output and the sign of some inputs. Let R be the rectangle with partition (X_m, Y_m) that maximizes $\left||\mathsf{IP}_m^{-1}(1) \cap R| - |\mathsf{IP}_m^{-1}(0) \cap R|\right|$. Then it follows that for every τ and every rectangle $\bar{R}$ respecting Π we have

$$\left||f_\tau^{-1}(1) \cap \bar{R}| - |f_\tau^{-1}(0) \cap \bar{R}|\right| \leq \left||\mathsf{IP}_m^{-1}(1) \cap R| - |\mathsf{IP}_m^{-1}(0) \cap R|\right|.$$

We then get for every rectangle $\bar{R}$ respecting Π that

$$\begin{aligned}
\mathrm{Disc}(\bar{R}, \mathsf{IP}_G) &= \left||\mathsf{IP}_G^{-1}(1) \cap \bar{R}| - |\mathsf{IP}_G^{-1}(0) \cap \bar{R}|\right| / 2^n \\
&= \left|\sum_\tau |f_\tau^{-1}(1) \cap \bar{R}| - \sum_\tau |f_\tau^{-1}(0) \cap \bar{R}|\right| / 2^n \\
&\leq \sum_\tau \left||f_\tau^{-1}(1) \cap \bar{R}| - |f_\tau^{-1}(0) \cap \bar{R}|\right| / 2^n \\
&\leq 2^{n-2m} \left||\mathsf{IP}_m^{-1}(1) \cap R| - |\mathsf{IP}_m^{-1}(0) \cap R|\right| / 2^n \\
&= 2^{-2m} \left||\mathsf{IP}_m^{-1}(1) \cap R| - |\mathsf{IP}_m^{-1}(0) \cap R|\right| \\
&= \mathrm{Disc}(R, \mathsf{IP}_m) \\
&\leq 2^{-m/2}
\end{aligned}$$

where we use in the last line that the discrepancy of IP_m is known to be $2^{-m/2}$.

B Proof of Observation 17

Remember that we want to encode

$$y_i \vee (\mathsf{IP}_{G_i} \neq z)$$

where G_i is a graph such that all vertices have degree at most 2 and $\mathsf{IP}_{G_i}(X) = \bigoplus_{xy \in E} x \wedge y$. In our construction, it will be helpful to use the notion of pathwidth of a graph. So consider a graph $G = (V, E)$. A path decompostion of G is defined to be a sequence $B_1, \dots, B_s$ of so-called *bags* where $B_i \subseteq V$ for all $i \in [s]$ with the following properties:

- for every $v \in V$ there is an $i \in [s]$ such that $v \in B_i$,
- for every $e = uv \in E$ there is an $i \in [s]$ such that $\{u, v\} \subseteq B_i$, and
- for every $i, j \in [s]$ with $i \leq j$, if $v \in B_i \cap B_j$ then for all k with $i \leq k \leq j$ we have $v \in B_j$.

The width of a path decomposition is defined as $\max_{i \in [s]}(|B_i| - 1)$. The pathwidth of G is the minimal width of a path decomposition of G.

We will first encode IP_{G_i} as follows: because of the degree bound, we know that G_i consists of a union of paths and cycles. It follows that G_i has a path decomposition $B_1, \dots, B_s$ of width 2. Let $V^r := \bigcup_{i \in [r]} B_i$ and let G_i^r be the graph $G_i[V^r]$ induced by V_r in G_i. We inductively construct a CNF-formula F_r with auxiliary variables $z_1, \dots, z_r$ and of size $O(r)$ such that for every assignment τ to X, there is exactly one extension of τ to a model τ' of F_r and $\tau'(z_r)$ is the value of $\mathsf{IP}_{G_i^r}(\tau)$. For $r = 1$, this is easy to do, since the number of variables and edges is constant. For the induction step, observe that the additional clauses for F_r only need to contain z_{r-1}, z_r and the variables in B_{r-1} and B_r. In particular, it follows that there is a linear size CNF-encoding F of IP_{G_i} whose models compute the value of that function in a variable z_n.

Define the primal graph of a CNF-formula F to be the graph whose vertices are the variables in F and whose edge set contains an edge xy if and only if x and y appear in a common clause of F. Define for every $i \in [s-1]$ new bags $B_i' := B_i \cup B_{i+1} \cup \{z_{r-1}, z_r\}$. Then $B_1', \dots, B_{s-1}'$ is a path decomposition of width at most 7 of the primal graph of F.

Next, add two clauses that force $z \neq z_n$. Finally, add y_i to all clauses of the resulting formula. Call the result ϕ_i. Note that ϕ_i encodes $y_i \vee (\mathsf{IP}_{G_i} \neq z)$ as required. Moreover, since adding two variables increases the pathwidth by at most two, the primal graph of ϕ_i has pathwidth at most 9. Now using the fact that taking subgraphs does not increase the pathwidth of a graph and that all CNF-formulas whose primal graph has pathwidth k have OBDD-representations of width $O(2^k)$, see [11], completes the proof.

C Strategy extraction

In this section, we show how to perform strategy extraction for OBDD-proof systems with changes of variable orders, proving Proposition 21. Instead of directly extracting rectangle decision lists, we make an intermediate step using OBDD-decision lists which we define next.

▶ **Definition 23.** *An OBDD-decision list of length s is a sequence $(L_1, c_1), \dots, (L_s, c_s)$ where the $c_i \in \{0, 1\}$ are truth values and the L_i are OBDDs, and L_s computes the constant function 1. Let V be the set of variables occurring in the circuits L_i. The decision list computes a function $f : \{0,1\}^V \to \{0,1\}$ as follows. Given an assignment $\tau : V \to \{0,1\}$, let $i = \min\{1 \leq j \leq s \mid L_j(\tau) = 1\}$. Then we have $f(\tau) = c_i$. The* width *of the OBDD-decision list is the maximal width of the L_i. We say that there are k variable order changes in the decision list if there are k indices $i \in [s-1]$ for which the variable order of L_i and L_{i+1} differ.*

The next result states that OBDD-decision lists can be efficiently extracted from OBDD-refutations.

▶ **Proposition 24.** *There is a linear-time algorithm that takes an* OBDD$(\wedge, \exists, \forall, \models, r)$*-refutation of length s of a PCNF formula Φ and outputs a family of OBDD-decision lists of*

length at most s and width at most w computing a universal winning strategy for Φ, where w is the width of the refutation. Moreover, if there are at most k variable order changes in the refutation, then the same is true in each of the OBDD-decision lists.

Proof. Let $R = L_1, \ldots, L_s$ be an OBDD-refutation of a PCNF formula Φ. For each universal variable u, the algorithm is going to compute an OBDD decision list L_u as follows. Let $L_{i_1} = L_{j_1}[u/c_1], L_{i_2} = L_{j_2}[u/c_2], \ldots, L_{i_\ell} = L_{j_\ell}[u/c_\ell]$ be the lines of R obtained by universal reduction of variable u in their order of appearance in R, that is, $i_1 < i_2 < \ldots < i_\ell$ and $1 \leq j_r < i_r \leq k$ for each $r \in [\ell]$. The decision list is $\mathsf{L}_u = (\neg L_{i_1}, c_1), (\neg L_{i_2}, c_2), \ldots, (\neg L_{i_\ell}, c_\ell), (1, 1)$. These lists can be constructed in linear time by scanning the proof line by line and adding the pair $(\neg L_i, c)$ to the decision list L_u whenever $L_i = L_j[u/c]$ is derived from L_j by universal reduction (recall that OBDDs can be negated simply by swapping the 0 and 1 sinks).

We claim that the Boolean functions $\vec{f} = \{f_u\}_{u \in \mathrm{var}_\forall(\Phi)}$ computed by the decision lists L_u represent a winning universal strategy for Φ. We begin by observing that, for every assignment τ of the existential variables, there is a unique assignment $\vec{f}(\tau)$ of the universal variables such that $\tau \cup \vec{f}(\tau)$ is consistent with $\vec{f}$: the OBDDs in each decision list L_u only contain variables that precede u in the quantifier prefix, so that each function f_u only depends on these variables and no circular dependencies can arise. The assignment $\vec{f}(\tau)$ can be computed simply by following the order of universal variables in the quantifier prefix.

Let m denote the number of clauses of Φ. We now prove, by downward induction on i for $m \leq i \leq s$, that φ_i is falsified by any assignment τ that is consistent with $\vec{f}$, where $\varphi_i = \bigwedge_{j=1}^{i} L_j$ denotes the conjunction of proof lines up to i. Since φ_m is logically equivalent to the matrix of Φ, this implies that $\vec{f}$ is a universal winning strategy. The base case $i = s$ is trivial as $L_s \equiv 0$ is falsified under any assignment $\tau \cup \vec{f}(\tau)$. For the induction step, assume that the assignment $\tau \cup \vec{f}(\tau)$ falsifies φ_i for each assignment τ of the existential variables. We consider two cases:

1. If L_i is derived using conjunction, projection, entailment, or reordering, then $\varphi_{i-1} \models L_i$, so any assignment that falsifies L_i must falsify φ_{i-1} as well. In combination with the induction hypothesis, this tells us that the assignment $\tau \cup \vec{f}(\tau)$ falsifies φ_{i-1} for each assignment τ of the existential variables.
2. Otherwise, the line $L_i = L_j[u/c]$ is derived from L_j with $j < i$ by universal reduction. Towards a contradiction, assume that there is an assignment τ of the universal variables so that $\tau \cup \vec{f}(\tau)$ satisfies φ_{i-1} but falsifies L_i. Consider the decision list L_u. By construction, it contains the pair $(\neg L_i, c)$, and since φ_{i-1} is satisfied by $\tau \cup \vec{f}(\tau)$, the OBDD $\neg L_{i_r}$ is falsified for each pair $(\neg L_{i_r}, c_{i_r})$ that precedes $(\neg L_i, c)$ in L_u. Since L_i is falsified by $\tau \cup \vec{f}(\tau)$, the OBDD $\neg L_i$ is satisfied and thus $f_u(\tau \cup \vec{f}(\tau)) = c$. But $L_i = L_j[u/c]$ is falsified, so L_j must be falsified as well, a contradiction.

Observing that the variable orders appearing in L_u also appear in R and the order in which they appear is the same, completes the proof. ◀

▶ **Lemma 25.** *If there is an OBDD-decision list of length s and width w computing a function $f : \{0,1\}^V \to \{0,1\}$. Let $\pi_1, \ldots, \pi_k$ be the variable orders of the OBDDs used in that order, and let for every $i \in [k]$ the pair (X_i, Y_i) of variable sets be a partition of V such that X_i is the set of variables appearing in a prefix of π_i. Then there is a rectangle decision list of length $w(s-1)+1$ computing f with partitions $\pi_1, \ldots, \pi_k$ appearing also in that order.*

The proof of Lemma 25 makes use of the following well-known connection between OBDD and rectangles, see e.g. [18].

▶ **Theorem 26.** *Let g be a function in variables X computed by a π-OBDD of width w. Let X_1 be a prefix of the variable order π and let $X_2 := X \setminus X_1$. Then $g(X) = \bigvee_{i=1}^{w} R_i(X)$, where every R_i is rectangle with partition (X_1, X_2).*

Proof of Lemma 25. Let $(L_1, c_1), \ldots, (L_s, c_s)$ be an OBDD-decision list of length s and width w computing function $f : \{0,1\}^V \to \{0,1\}$, and let (X_i, Y_i) be partitions of V as in the statement of the lemma. By Theorem 26, each OBDD L_j for $1 \leq j < s$ with variable order π_i is equivalent to a disjunction $\bigvee_{j=k}^{w} R_{jk}(V)$ of rectangles with respect to (X_i, Y_i). We construct a rectangle decision list by replacing each pair (L_j, c_j) for $1 \leq j < s$ by the sequence $(R_{j1}, c_j), \ldots, (R_{jw}, c_j)$. We can simply append (L_s, c_s) to this sequence since the constant L_s trivially is a rectangle. The resulting rectangle decision list computes f and has length $w(s-1)+1$. Moreover, it makes $k-1$ partition changes and the partitions it uses are $(X_1, Y_1), \ldots (X_k, Y_k)$ in that order. ◀

Combining Proposition 24 and Lemma 25 directly yields Proposition 21.