OBDD(Join) Proofs Cannot Be Balanced

Sergei Ovcharov
St. Petersburg State University, Russia

Abstract

We study OBDD-based propositional proof systems introduced in 2004 by Atserias, Kolaitis, and Vardi that prove the unsatisfiability of a CNF formula by deduction of an identically false OBDD from OBDDs representing clauses of the initial formula. We consider a proof system OBDD($\wedge$) that uses only the conjunction (join) rule and a proof system OBDD($\wedge$, reordering) (introduced in 2017 by Itsykson, Knop, Romashchenko, and Sokolov) that uses the conjunction (join) rule and the rule that allows changing the order of variables in OBDD.

We study whether these systems can be balanced i.e. every refutation of size S can be reassembled into a refutation of depth $O(\log S)$ with at most a polynomial-size increase. We construct a family of unsatisfiable CNF formulas F_n such that F_n has a polynomial-size tree-like OBDD($\wedge$) refutation of depth poly(n) and for arbitrary OBDD($\wedge$, reordering) refutation Π of F_n for every $\alpha \in (0, 1)$ the following trade-off holds: either the size of Π is $2^{\Omega(n^\alpha)}$ or the depth of Π is $\Omega(n^{1-\alpha})$. As a corollary of the trade-offs, we get that OBDD($\wedge$) and OBDD($\wedge$, reordering) proofs cannot be balanced.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof complexity, OBDD, lower bounds, depth of proofs

Digital Object Identifier 10.4230/LIPIcs.MFCS.2023.72

Acknowledgements The author is grateful to his supervisor Dmitry Itsykson for a suggesting of the problem, for the many fruitful discussions of it and for his active help in the preparation of this paper.

1 Introduction

The paper devotes to propositional proof complexity theory. Propositional proof systems are used for certifying that a given CNF formula is unsatisfiable. Investigation of propositional proof systems is highly connected with the construction of solvers for the Boolean satisfiability problem (SAT-solvers). The execution protocol of a SAT solver running on an unsatisfiable formula may be considered as a certificate of unsatisfiability. Every SAT solver is based on some proof system. For example, CDCL solvers are based on Resolution [3], Pseudo Boolean solvers are based on Cutting Planes [8], OBDD-solvers are based on OBDD-based proof systems [2].

The minimal refutation size of a formula is a natural lower bound on the running time of the corresponding SAT-Solvers. In this paper, we also study the depth of refutations, i.e. the length of the longest path from a clause of a refuted formula to a contradiction. The depth is a very natural but not a much-studied measure of the proofs. The minimal depth of a refutation is a lower bound on the parallel running time of the corresponding solver.

Balancing proof systems

We consider only *refutational* proof systems. Each refutational proof system Π operates with *proof lines*, and each proof line is a Boolean predicate represented in some fixed way. Initially, all clauses of refuted formulas are represented by proof lines and new proof lines may be derived using a finite set of inference rules. The goal is to derive an identically false proof line. Every refutational proof system is defined by the type of predicates that may be

48th International Symposium on Mathematical Foundations of Computer Science (MFCS 2023).
Editors: Jérôme Leroux, Sylvain Lombardy, and David Peleg; Article No. 72; pp. 72:1–72:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

used as proof lines and by the list of inference rules. For example, in Resolution proof lines are clauses, in Cutting Planes [10] proof lines are linear inequalities with integer coefficients and Boolean variables, in Frege proof systems proof lines are propositional formulas, etc.

The *size* of a refutation is the total size of representations of all used proof lines.

Every refutation can be represented as a directed acyclic graph with one source corresponding to a contradiction and sinks corresponding to clauses of a refuted formula, and every proof line is obtained by the descendants using the inference rule. The *depth* of the refutation is the depth of the corresponding graph i.e. the length of the longest path from the source to a sink.

We say that proofs in some proof system can be balanced if it is always possible to reassemble each refutation in such a way that its depth becomes logarithmic in its size (perhaps with a polynomial-size increase).

The question of whether Resolution proofs can be balanced is trivial. Indeed, consider the formula $(x_1 \vee \ldots \vee x_n) \wedge (\neg x_1) \wedge \ldots \wedge (\neg x_n)$. It is easy to see that every refutation of this formula must have the depth at least n, therefore, Resolution refutations cannot be balanced in the general case. Urquhart [16] studied if refutations of $O(1)$-CNF formulas can be balanced for which the question is less trivial. It was proven that there exists a family of 3-CNF formulas F_{n_i} with n_i variables having a Resolution refutation of polynomial size but every its refutation must have depth $\Omega(n_i/\log n_i)$. Therefore Resolution refutations cannot be balanced even for $O(1)$-CNF formulas.

Atserias, Bonet, and Levy [1] proved that Cutting Planes proofs cannot be balanced either. However, it is known that refutations in Frege systems can be balanced (see, for instance, [15]).

OBDD-based proof systems

In 1986 Bryant [6] proposed an important way to represent a Boolean function. Every such function can be represented as a branching program with two sinks so that variables on every path from the source to a sink appear in the same order π. Such representation is called *Ordered Binary Decision Diagram* (OBDD or π-OBDD if we need to specify that the variables are ordered according to π). The restriction on a variables order allows us to perform many useful operations with OBDD efficiently e.g. check satisfiability, compute the conjunction of two OBDDs (given they use the same variable order), etc [13].

Atserias, Kolaitis, and Vardi [2] introduced an OBDD-based refutational proof system. Among them, we are most interested in OBDD($\wedge$). OBDD($\wedge$) represents clauses of an unsatisfiable formula as π-OBDDs for some order π and the only refutation rule allows deriving the conjunction of two OBDDs which were derived earlier. The size of the refutation is the total size of the OBDDs in it.

Itsykson, Knop, Romashchenko, Sokolov [12] proposed the OBDD($\wedge$, reordering) proof system. OBDD($\wedge$, reordering) is obtained from OBDD($\wedge$) by adding the reordering derivation rule that allows changing variables order of the derived OBDDs. While now OBDDs in the refutation may use different variable orders, the conjunction rule can be only applied to OBDDs that use the same variable order (otherwise it would be NP-hard to verify the correctness of such rule, see [14], Lemma 8.14).

Notice that the formula $(x_1 \vee \ldots \vee x_n) \wedge (\neg x_1) \wedge \ldots \wedge (\neg x_n)$ that we considered above has a tree-like OBDD($\wedge$) refutation of polynomial size and logarithmic depth.

For both the Resolution and the Cutting Planes proof systems there exist a family of formulas for which a refutation of small depth does not exist at all. We emphasize that it is not the case for OBDD-based proof systems. Indeed, every CNF formula with m clauses has a tree-like OBDD($\wedge$) refutation of the depth $\log(m)$; the graph of this refutation is a full binary tree with m leaves. Note that the size of this proof can differ dramatically from the size of the minimum refutation. Hence in the notion of balancing we require that the size of the balanced proof should be bounded by a polynomial from the size of the initial proof.

Our contribution

In Theorem 12 we construct a family of unsatisfiable formulas F_n having poly(n) size tree-like OBDD($\wedge$) refutations such that the following size vs depth trade-offs holds. For every $\alpha \in (0,1)$, any OBDD($\wedge$, reordering) refutation of F_n of depth $O(n^{1-\alpha})$ requires size at least $2^{\Omega(n^{\alpha})}$. Hence we prove that dag-like and tree-like OBDD($\wedge$), OBDD($\wedge$, reordering) proofs cannot be balanced.

Formulas for which the trade-offs hold are the Pebbling formulas based on the grid graphs $\mathrm{Peb}(\mathrm{Grid}_n)$. Pebbling formulas are a well-studied family of formulas ([5], [16], [4]). Moreover, they were used for proving Resolution depth lower bounds in [16]. However, usually, they are used together with Pebbling games and Pebbling numbers of graphs. This is not the case for our result since we rely significantly on the structure of the grid graphs (including self-similarity and expansion) by themselves and do not use Pebbling games.

In Section 2 we define the main notions. In Subsection 3.1 we prove the OBDD size lower bounds for some set of *hard* $\mathrm{Peb}(\mathrm{Grid}_n)$ subformulas. In Subsection 3.2 we prove the mentioned size vs depth trade-offs.

Open question

It would be interesting to study the similar questions for OBDD($\wedge$, weakening) proof system which is obtained from OBDD($\wedge$) by adding the *weakening* rule. The weakening rule allows deriving from an OBDD any its semantical implication represented by OBDD in the same order.

2 Preliminaries

▶ **Definition 1** (Branching Program). *Let $X = \{x_1, \ldots, x_n\}$ be a set of Boolean variables. A branching program is a directed acyclic graph with one node with indegree 0 (source) several inner nodes with outdegree 2 and two nodes with outdegree 0 (sinks). Every node except sinks is labeled with some variable from X, one of its outgoing edges is labeled with 0 and the other one is labeled with 1. One sink is labeled with 0 and the other one is labeled with 1.*

Every branching program represents some Boolean function of n variables. To compute a value of the function on input $x_1 = a_1, \ldots x_i = a_i, \ldots x_n = a_n$ we start a path from the source, and for every vertex labeled with variable x_i we continue the path along the edge labeled with a_i, such a path reaches a sink and the label of this sink is the value of the function.

▶ **Definition 2** (Ordered Binary Decision Diagram(OBDD)). *A branching program is called* OBDD *if variables on every path from the source to sinks appear according to some fixed order of variables.*

Sometimes we write π-OBDD instead of OBDD *to emphasize that variables appear according to the order of variables π.*

The order restriction in OBDDs allows to perform many useful operations on OBDDs efficiently e.g. minimize, check satisfiability, compute the conjunction of two OBDDs given they have a same order of variables, etc. [13].

Let us define a propositional proof system OBDD($\wedge$, reordering).

▶ **Definition 3** (OBDD($\wedge$, reordering)). *Let $\varphi = \bigwedge_i C_i$ be an unsatisfiable CNF formula. A refutation of φ is a sequence of* OBDD*s $D_1, D_2, \dots, D_t$ such that D_t is the constant false* OBDD *and for all $1 \le i \le t$ the diagram D_i either represents a clause of φ or obtained from the previous D_j's by one of the following derivation rules.*

- ***Conjunction (or join)*** *rule allows deriving an π-OBDD for $D_1 \wedge D_2$ from π-OBDDs D_1 and D_2. We emphasize here that the conjunction rule can be only applied to* OBDD*s with the same order of variables.*
- ***Reordering*** *rule allows deriving an* OBDD *B from an equivalent* OBDD *A (note that A and B may use different variable orders).*

The size of a refutation is the sum of the sizes of the OBDD*s from it.*

Every OBDD($\wedge$, *reordering*) *refutation can be represented as a directed acyclic graph (DAG) in which nodes are labeled with* OBDD*s from the refutation such that each sink is labeled with a* OBDD *for some clause of φ, the source is labeled with the constant false* OBDD*, and an* OBDD *in every inner node is the result of the application of some derivation rule to the* OBDD*s from the descendants.*

A refutation is called tree-like if every node except the source has indegree one.

The depth of a refutation is the length of the longest path from the source to a sink.

We call a refutation a π-OBDD($\wedge$) refutation if all OBDD*s have the same order (i.e. no reordering rule was applied).*

Note, that in order to call OBDD($\wedge$, reordering) a proof system (in the sense of Cook-Reckhow [9]) we need to be able to efficiently check if some OBDD is the result of an application of the derivation rules to some others OBDDs. Fortunately, the restriction on a variable order allows us to do that, as we mentioned before (for the details see [12]).

▶ **Definition 4** (Pebbling formulas (see for instance [16])). *Let $G = (E, V)$ be a directed acyclic graph. We associate with each node of G a distinct Boolean variable x; we will identify nodes and the associated variables. The* Peb(G) *formula is the conjunction of the following clauses:*

- *$(\neg u_1 \vee \ldots \vee \neg u_n \vee v)$, where $v \in V$ and $\{u_1, \dots, u_n\}$ is the set of all nodes such that edge $(u_i, v) \in E$. We denote this clause by $(u_1, \dots, u_n \to v)$. Note that if v is a source of the graph then $n = 0$. We call such clauses first type clauses.*
- *$(\neg v)$, where v is a sink. We call such clauses second type clauses.*

Note that for every directed acyclic graph G the formula Peb(G) is unsatisfiable.

Main goal of out work is to prove that OBDD($\wedge$) and OBDD($\wedge$, reordering) refutations cannot be balanced. In order to do it we construct a family of CNF formulas such that the formulas have small OBDD($\wedge$) refutations but they do not have refutations with small size and depth simultaneously.

▶ **Lemma 5** ([7]). *For every directed acyclic graph G and for every order of variables π formula* Peb(G) *has tree-like π-OBDD($\wedge$) refutation of size $O(|V|^2)$ and depth $O(|V|)$.*

Proof. See Appendix A. ◀

▶ **Definition 6** (Graph Grid_n). *Let* Grid_n *be a graph of the* $(n-1)\times(n-1)$ *grid, with edges directed top to bottom and left to right.*

In other words, the set of vertices is

$$V_n = \{(i,j),\ i,j \in [n]\}$$

and the set of edges is

$$E_n = \{((i,j),(i+1,j))|\ i \in [n-1],\ j \in [n]\} \cup \{((i,j),(i,j+1))|\ i \in [n],\ j \in [n-1]\}$$

▶ **Corollary 7.** *Formula* $\mathrm{Peb}(\mathrm{Grid}_n)$ *has* π*-OBDD*$(\wedge)$ *refutation of the size* $O(n^4)$ *and of the depth* $O(n^2)$ *for every variable ordering* π*.*

Proof. Follows from Lemma 5. ◀

Now, in order to prove that OBDD(join) proofs cannot be balanced, it is sufficient to prove size vs. depth trade-offs for refutations of $\mathrm{Peb}(\mathrm{Grid}_n)$. We prove such trade-offs in Theorem 12 but we still need several auxiliary lemmas.

▶ **Lemma 8** (Folklore). *Let* G *be a directed acyclic graph with only one sink. Then* $\mathrm{Peb}(G)$ *minimal unsatisfiable i.e. a conjunction of every proper subset of its set of clauses is satisfiable.*

Proof. See Appendix B. ◀

▶ **Notation 9.** *For a graph* $G(V,E)$ *(directed or undirected) and for two disjoint sets* $A, B \subset V$ *denote by* $E(A,B)$ *the set of edges with one end in* A *and the other one in* B*.*

Note that for directed graphs we include in $E(A,B)$ *both the edges directed from* A *to* B *and the edges directed from* B *to* A*.*

▶ **Definition 10** (Graph expansion [5]). *Expansion of the graph* $G(V,E)$ *is the minimum value of* $|E(U, V\setminus U)|$ *among all subsets* $U \subset V$ *such that* $\frac{|V|}{3} \le |U| \le \frac{2|V|}{3}$*.*

▶ **Lemma 11** (Folklore). $e(Grid_n) \ge \frac{1}{4}n$*.*

Proof. Consider an arbitrary subset $U \subset V$ such that $\frac{1}{3}|V| \le |U| \le \frac{2}{3}|V|$.

Assume that there are at least $\frac{n}{4}$ columns of the grid containing nodes from both U and $V\setminus U$. Then there is at least one pair of incident nodes in every such column with one node in U and the other one in $V\setminus U$. Then the edges between the vertices from such pairs lie in $E(U, V\setminus U)$. Thus, $|E(U, V\setminus U)| \ge \frac{n}{4}$.

Now assume that there are at least $\frac{3n}{4}$ columns lying completely in U or in $V\setminus U$. Since $\frac{|V|}{3} \le |U| \le \frac{2|V|}{3}$, there is at least one column completely lying in U and there is at least one column completely lying in $V\setminus U$. Therefore, in each row, there is at least one pair of incident nodes with one node in U and the other one in $V\setminus U$. In this case $|E(U, V\setminus U)| \ge n$. ◀

We want to point out that Grid_n graphs are not expanders in the conventional sense (see for example [11]) since Grid_n graph has n^2 nodes but $e(\mathrm{Grid}_n) = \Theta(n)$ (Lemma 11 shows only that $e(\mathrm{Grid}_n) = \Omega(n)$ but upper bounds for $e(\mathrm{Grid}_n)$ are trivial).

3 Depth vs size trade-offs

In this section we prove our main result.

▶ **Theorem 12.** *For every* $\alpha \in (0,1)$ *and for every* $\mathrm{OBDD}(\wedge, reordering)$ *refutation of* $\mathrm{Peb}(\mathrm{Grid}_n)$ *at least one of the following holds:*

- *the depth of the refutation is* $\Omega(n^\alpha)$*;*
- *the size of the refutation is* $2^{\Omega(n^{(1-\alpha)})}$*.*

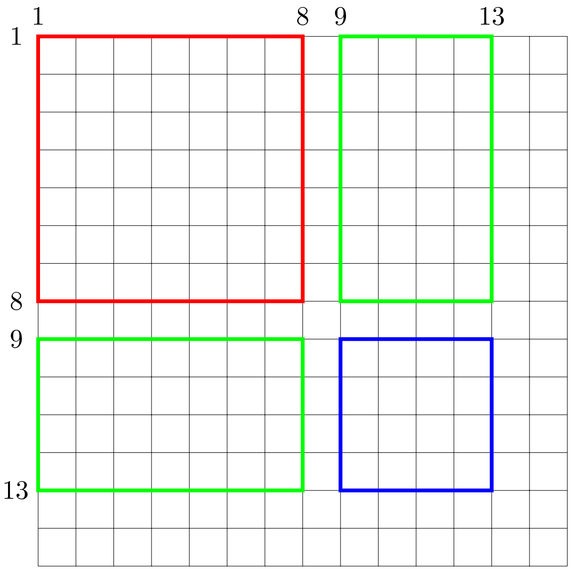

Figure 1 Possible partition for $n = 15,\ m = 13,\ k = 5$.

By a *configuration* we mean a conjunction of a subset of the set of Peb(Grid$_n$) clauses. In Subsection 3.1 we will show that some *hard* configurations cannot be represented by a small OBDD (see Lemma 13). Loosely speaking we are interested in configurations in which at least one clause from the top left part of the grid is missing and that contain many clauses from the bottom right part.

In Subsection 3.2 we finish the proof of Theorem 12 using Lemma 13. Namely, we show that either it is possible to find a hard configuration in the proof graph or the depth of the proof is large.

3.1 Configurations that are hard for OBDDs

Consider an arbitrary order π of variables of the formula Peb(Grid$_n$).

Let $\{x_0, y_0, x_1, y_1\} \subset [n]$. We denote by $[x_0, x_1] \times [y_0, y_1]$ the induced subgraph on the vertices set $\{(x, y) \mid x_0 \le x \le x_1\ y_0 \le y \le y_1\}$.

Let $m \in [n-1]$. We divide the nodes of the subgraph $[1, m] \times [1, m]$ into the four parts:

- $[1, m-k] \times [1, m-k]$,
- $[m-k+1, m] \times [m-k+1, m]$,
- $[1, m-k] \times [m-k+1, m]$,
- $[m-k+1, m] \times [1, m-k]$,

where $k \in [m]$ (see Fig. 1).

We divide variables associated with the vertices of $[m-k+1, m] \times [m-k+1, m]$ into two equal (or differing by at most 1) parts in such a way that each variable from the first part appears in the order π before each variable of the second part. We denote the first part by A_π and the second by B_π. Recall that we identify each variable with the associated node and assume that $A_\pi, B_\pi \subset V(\text{Grid}_n)$.

Subgraph $[m-k+1, m] \times [m-k+1, m]$ is isomorphic to the Grid$_k$ and $(|A_\pi| - |B_\pi|) \le 1$, $|A_\pi| + |B_\pi| = k^2$, hence $|E(A_\pi, B_\pi)| \ge \frac{k}{4}$ by Lemma 11.

Since the graph is directed, each edge in $E(A_\pi, B_\pi)$ is directed from A_π to B_π or from B_π to A_π. Let us consider the direction with the majority of the edges. We denote the set of the corresponding edges by $E_{0,\pi}$, $|E_{0,\pi}| \ge |E(A_\pi, B_\pi)|/2 \ge \frac{k}{8}$. All edges in $E_{0,\pi}$ are directed from A_π to B_π or B_π to A_π. This gives us two cases that we consider later.

Using the following procedure we remove some of the edges from $E_{0,\pi}$ to form a matching (that we will denote by $E_{1,\pi}$).

While $E_{0,\pi}$ is not empty:

- Choose an arbitrary edge $e \in E_{0,\pi}$, add it to $E_{1,\pi}$, remove it from $E_{0,\pi}$.
- If $E_{0,\pi}$ still contains edges adjacent to e then we remove them. Since every node in Grid_n has degree at most 4 then we remove at most 6 edges per step.

We obtain matching $E_{1,\pi} \subset E_{0,\pi}$ and $|E_{1,\pi}| \geq \frac{|E_{0,\pi}|}{7} \geq \frac{k}{56}$.

We call a node *special* if it is an head of some edge from $E_{1,\pi}$. There are $|E_{1,\pi}|$ special nodes since $E_{1,\pi}$ is a matching.

Using the following procedure we choose a subset W_π of the set of all special nodes such that the distance between any two nodes from W_π is at least 7.

While there are special nodes:

- Choose an arbitrary special node v that is not removed at the previous steps and add it to W_π.
- Remove all special nodes at the distance at most 6 from v. We removed at most $(6 + 1 + 6)^2 = 169$ nodes, since all removed nodes are in the square of the size 13×13 and with the center in v.

Since at each step we remove at most 169 special nodes and add one to W_π, $|W_\pi| \geq |E_{1,\pi}|/169 \geq k/9464$.

For a node $v = (i_0, j_0) \in V(\mathrm{Grid}_n)$ we denote by $B(v)$ the set $\{(i,j) \in V(\mathrm{Grid}_n) \mid |i - i_0| \leq 1, |j - j_0| \leq 1\}$ (i.e. the ball in the l_∞ metric). We refer to $B(v)$ as a *ball* although it is not a ball in the graph distance sense.

For every node $w \in W_\pi$ there is a unique edge in $E_{1,\pi}$ with an endpoint w. Let us denote the set of start-points of such edges by U_π.

▶ **Lemma 13.** *Let φ be a conjunction of some subset of* $\mathrm{Peb}(\mathrm{Grid}_n)$ *clauses. Suppose at least one clause of first type associated with a variable from the* $[1, m-k-1] \times [1, m-k-1]$ *is not included in φ. Also, suppose that φ contains exactly d clauses associated with variables from W_π. Then the minimum size of an π-OBDD for φ is at least 2^d.*

Proof. To prove that the size of any π-OBDD representation of φ is at least 2^d it is sufficient to split π into two consequent parts and define 2^d substitutions into the variables of the first part such that applications of them to φ lead to 2^d different Boolean functions. To show that two substitutions ρ_0 and ρ_1 lead to two different functions we just define a substitution ρ into the second part of variables such that $\varphi|_{\rho_0 \circ \rho} \neq \varphi|_{\rho_1 \circ \rho}$.

We have already divided the variables of $[m-k+1, m] \times [m-k+1, m]$ into two parts A_π and B_π according to π. Let us fix a partition of π into two parts such that A_π lies in the first and B_π in the second.

Let W'_π be the nodes from W_π whose associated clauses are in φ. Let U'_π be the nodes from U_π connected with W'_π. Let $W'_\pi = \{w_1, \ldots, w_d\}$ and $U'_\pi = \{u_1, \ldots, u_d\}$.

Substitutions we are defining differ only on a small set of nodes. To every variable outside this set each substitution assigns a fixed value (values may differ for different variables); we now define these values. Let $z = (x_0, y_0)$ be an arbitrary node from $[1, m-k-1] \times [1, m-k-1]$ whose first type clause is missing from φ (see Fig. 2).

- **Assignment to the variables in $[1, n] \times [1, n] \setminus [x_0, n] \times [y_0, n]$:**
  To the variables whose corresponding nodes are strictly to the left or strictly at the top from z all substitutions assign 1. Note that all first-type clauses corresponding to the nodes in which we substitute 1 are satisfied (follows trivially from the definition of the Grid_n clauses).

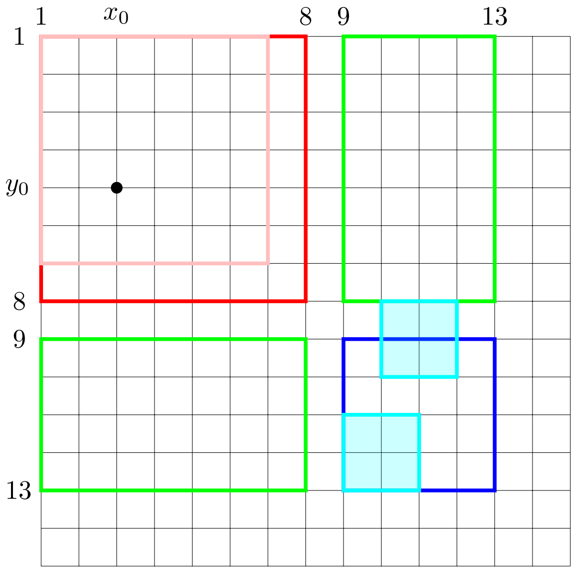

Figure 2 Balls $B(w_i)$ are shown with light blue; their centers lie in $[m-k+1, m] \times [m-k+1, m]$ (blue rectangle). Black bold point is (x_0, y_0), it lies in $[1, m-k-1] \times [1, m-k-1]$ (pink rectangle).

- **Assignment to the variables in $[x_0, n] \times [y_0, n] \setminus \bigcup_i B(w_i)$:**
  Consider the rectangle $[x_0, n] \times [y_0, n]$. It contains square $[m-k, n] \times [m-k, n]$ and, therefore, all balls $B(w_i)$ for $i \in [d]$ (their centers lie in $[m-k+1, m] \times [m-k+1, m]$). Let $L = [x_0, n] \times [y_0, n] \setminus \bigcup_i B(w_i)$.

  ▷ Claim 14. For every node $v \in L$ there exists a directed path from z to v that lies completely in L.

  Proof. Let us first note that for every $i \in [d]$ and for every $(x, y) \in B(w_i)$ it holds that $x > x_0, y > y_0$. Indeed, $w_i \in [m-k+1, m] \times [m-k+1, m]$ hence $(x, y) \in [m-k, m] \times [m-k, m]$ but $z = (x_0, y_0) \in [1, m-k-1] \times [1, m-k-1]$.
  It is sufficient to prove that for every $x \in L \setminus \{z\}$ at least one of its immediate predecessors lies in L. If we prove it we can build a desired path by induction. Suppose that there is a node $x \in L \setminus \{z\}$ such that its immediate predecessors lie outside of L. Hence each of them lies either in $\bigcup_i B(w_i)$ or in $V(\text{Grid}_n) \setminus [x_0, n] \times [y_0, n]$.
  Firstly, consider the case in which one of the predecessors lies outside the $[x_0, n] \times [y_0, n]$. Then coordinates of x look like $(x_0, *)$ or $(*, y_0)$. Without loss of generality assume that $x = (x_0, h)$ for some $h \geq y_0$. Then there is the following path: $z = (x_0, y_0)$, $(x_0, y_0 + 1)$, $\ldots, (x_0, h) = x$ between z and x. All its nodes lie in $[x_0, n] \times [y_0, n]$ and none of them lie in $\bigcup_i B(w_i)$ due to the restriction on the coordinates of vertices from the balls that we mentioned earlier. Hence, this case is impossible.
  Therefore its predecessors lie in $\bigcup_i B(w_i)$. But one of its predecessors is above x and the other is to the left of x. Hence they lie in different balls (otherwise x would also lie in the ball, and this contradicts the assumption $x \in L$). Hence the distance between the balls is at most 2 which contradicts the construction from the beginning of the subsection. This concludes the proof of the claim. ◁

All the substitutions assign 0 to the variables in L. Now we check that no first-type clause from φ is falsified after that. There is no z's clause in φ. For every other node $z' \in L$ there is a path from z to z' in L. We assign 0 to the nodes on the path hence we assign 0 to some immediate predecessor of z' hence its clause is satisfied (again, by the definition of the clause).

- We have already defined the substitutions into all variables except $\bigcup_i B(w_i)$. Fix $j \in [d]$. There are two nodes with edges going from them to w_j. One of them is u_j. We denote the other one by r_j. Note that $\{u_j, r_j\} \subset B(w_j)$. Every substitution will assign 1 to r_j. Hence its first-type clause is always satisfied. To the nodes $B(w_j) \setminus \{u_j, w_j, r_j\}$ we always substitute 0. We need to check that it will not falsify their clauses. It is easy to see that each node from $B(w_j) \setminus \{u_j, w_j, r_j\}$ has at least one immediate predecessor in $L \setminus \{w_i, u_i, r_i | i \in [d]\}$ (see Fig. 3). But we assign zeros to the variables from this set. Hence clauses corresponding to the $B(w_j) \setminus \{u_j, w_j, r_j\}$ are always satisfied (see Fig. 3).

Note that we substitute 0 to the sink. Indeed, if the sink does not lie in any ball, then it lies in L. Hence we substitute 0 to it. Otherwise, the sink lies in some ball; denote the center of this ball by (x_1, y_1). Then $x_1 \leq n-1$ and $y_1 \leq n-1$. But then the sink is the most right bottom variable of the ball, hence it is substituted with 0.

Therefore, the second-type clause corresponding to the sink is satisfied.

At this point, we have defined substitutions on the set where their values coincide. Now we define substitutions to the remaining nodes i.e. $\{w_i, u_i \mid i \in [d]\}$. We need to consider to cases: whether $\{u_j \mid j \in [d]\}$ or $\{w_j \mid j \in [d]\}$ lie in the first part of the variable order.

Case 1: Suppose that $\{u_j \mid j \in [d]\}$ lies in the first part of the variable order. Consider all possible substitutions of zeros and ones into the $\{u_j \mid j \in [d]\}$. There are 2^d such substitutions. Note that every node from $\{u_j \mid j \in [d]\}$ has an immediate predecessor from $L \setminus \{u_j, w_j, r_j \mid j \in [d]\}$ hence its clause is satisfied.

We show that we can separate any two such substitutions $\rho_0 \neq \rho_1$ by some substitution ρ with support W'. There exists $j_0 \in [d]$ such that $\rho_0(u_{j_0}) \neq \rho_1(u_{j_0})$. Without loss of generality suppose that $\rho_0(u_{j_0}) = 0$ then $\rho_1(u_{j_0}) = 1$. We define a substitution ρ as follows: $\rho(w_j) = \rho_0(u_j)$.

- On the one hand substitution $\rho_0 \circ \rho$ does not falsify φ since the only clauses that can be falsified are clauses associated with $\{w_j \mid j \in [d]\}$ (we have already checked that the other clauses are satisfied). But the clause corresponding to the node w_j for $\{j \in [d]\}$ is falsified iff 0 is substituted into w_j and 1 are substituted to all its immediate predecessors. But if $\rho(w_j) = 0$ then $\rho_0(u_j) = 0$. Hence $\varphi|_{\rho\circ\rho_0} = 1$.
- On the other hand, $\rho(w_{j_0}) = 0$. The node w_{j_0} has two immediate predecessors: the node r_{j_0}, into which we always assign 1, and u_{j_0} such that $\rho_1(u_{j_0}) = 1$. Hence $(u_{j_0}, r_{j_0} \to w_{j_0})|_{\rho\circ\rho_1} = 0$ and $\varphi|_{\rho\circ\rho_1} = 0$.

Case 2: Suppose that $\{u_j \mid j \in [d]\}$ lies in the second part of the variable order.

Similarly to the Case 1, we define 2^d substitutions into the variables $\{w_j \mid j \in [d]\}$. We need to show that substitutions ρ_0 and ρ_1 lead to different Boolean functions. Again we find $j_0 \in [d]$ such that $\rho_0(w_{j_0}) = 0$ and $\rho_1(w_{j_0}) = 1$. In this case we define $\rho(u_j) = \rho_1(w_j)$.

Similarly to the Case 1, ρ satisfies clauses for nodes from $\{u_j \mid j \in [d]\}$. Also $\rho \circ \rho_1$ satisfy clauses for nodes from $\{w_j \mid j \in [d]\}$ (we just copy ρ_1 from $\{w_j \mid j \in [d]\}$ to $\{u_j \mid j \in [d]\}$). At the same time $\rho(u_{j_0}) = 1$, r_{j_0} is always substituted with 1 and $\rho_0(w_{j_0}) = 0$. Hence $(u_{j_0}, r_{j_0} \to w_{j_0})|_{\rho\circ\rho_o} = 0$ and $\varphi|_{\rho\circ\rho_0} = 0$. ◀

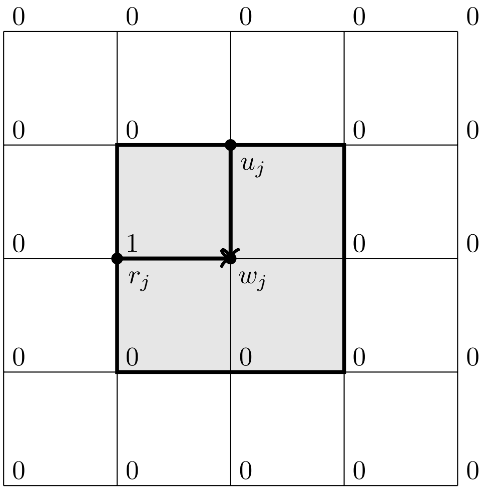

Figure 3 An example of a substitution to a ball's variables.

3.2 Proof of Theorem 12

Proof of Theorem 12. Let us fix $\alpha \in (0,1)$.

Now we show that for every OBDD($\wedge$, reordering) refutation of Peb(Grid$_n$) its depth is at least $n^{1-\alpha}/2$ or its size is at least $2^{n^\alpha/18928}$.

Suppose there exists a refutation with depth less than $n^{1-\alpha}/2$ and size less than $2^{n^\alpha/18928}$. Every refutation can be represented as directed acyclic graph such that:

- Each node is labeled with some OBDD from the refutation. Note that every such OBDD is equivalent to the conjunction of a subset of Peb(Grid$_n$) clauses. For every node, we add the conjunction to the label for clarity.
- Its only source is labeled with the constant false OBDD.
- Each sink is labeled with OBDD for some clause of Peb(Grid$_n$).
- If an OBDD in a node is obtained by the conjunction rule then the node has outdegree 2 and the OBDD in the node is a conjunction of the OBDDs in the descendants.
- If an OBDD in a node is obtained by the reordering rule then the node has outdegree 1 and the OBDD in the node is the result of the reordering rule applied to the OBDD in the node's descendant.

Let us divide the subgrid $[1, n-1] \times [1, n-1]$ into $[1, n-n^\alpha] \times [1, n-n^\alpha]$, $[n-n^\alpha+1, n-1] \times [n-n^\alpha+1, n-1]$, $[n-n^\alpha+1, n-1] \times [1, n-n^\alpha]$, $[1, n-n^\alpha] \times [n-n^\alpha+1, n-1]$ as in Subsection 3.1 (set $m = n-1$ and $k = n^\alpha$).

Let φ be a CNF formula and let S be a subset of its clauses. We denote $\varphi_S = \bigwedge_{C \in S} C$.

Consider an arbitrary OBDD($\wedge$, reordering) refutation of Peb(Grid$_n$). Consider its source. Since by Lemma 8, Peb(Grid$_n$) is minimal unsatisfiable, the source is labeled with the conjunction of all clauses (i.e. Peb(Grid$_n$) itself). In particular, all clauses for vertices from $[1, m] \times [1, m]$ lie there. We start a path at the source of the refutation. If the current node (initially the current node is the source) has only one immediate descendant then we move into the descendant until the current node has two of them. Assume that the current node is labeled with π-OBDD for some order of the variables π. By the definition of the conjunction rule the current node's descendants are also labeled with π-OBDDs. For this order π and parameters m and k find sets U_π, W_π as was described in Subsection 3.1. Suppose one of the descendants is labeled with formula φ_{S_1} and the other with φ_{S_2}. Then

$[1, m] \times [1, m] \subset S_1 \cup S_2$. Hence $|S_1 \cap W_\pi| \geq |W_\pi|/2$ or $|S_2 \cap W_\pi| \geq |W_\pi|/2$. Without loss of generality $|S_1 \cap W_\pi| \geq |W_\pi|/2$. Recall that $|W_\pi| \geq k/9464$ so $|S_1 \cap W_\pi| \geq k/18928$. Consider two cases: whether $[1, n-n^\alpha-1] \times [1, n-n^\alpha-1] \not\subset S_1$ or $[1, n-n^\alpha-1] \times [1, n-n^\alpha-1] \subset S_1$.

Case 1: $[1, n-n^\alpha-1] \times [1, n-n^\alpha-1] \not\subset S_1$. In this case we can apply Lemma 13 with $d = |S_1 \cap W_\pi| \geq k/18928 = n^\alpha/18928$ and variable order π. Hence the size of π-OBDD for φ_{S_1} from the current node is at least $2^{n^\alpha/18928}$. Therefore the size of the refutation is at least $2^{n^\alpha/18928}$. Hence this case is impossible.

Case 2: $[1, n-n^\alpha-1] \times [1, n-n^\alpha-1] \subset S_1$. All clauses from $[1, n-n^\alpha-1] \times [1, n-n^\alpha-1]$ are still in the conjunction.

We divide this square into 4 subsquares, same as we did with $[1, n-1] \times [1, n-1]$. Now we set $m = n - n^\alpha - 1$, $k = n^\alpha$ and repeat the actions for new values of m and k. Again we can move down at least one time in the refutation's graph so that at the current node there will be all clauses from the top left subsquare (this time it is $[1, n-2n^\alpha-2] \times [1, n-2n^\alpha-2]$).

Again, we divide this subsquare into 4 subsubsquares ($m = n - 2n^\alpha - 2$, $k = n^\alpha$) and so on. Case 1 is always impossible since k is always equal to n^α and we assumed that the size of the refutation is less than $2^{n^\alpha/18928}$. We can repeat the process $\frac{n}{n^\alpha+1} \geq n^{1-\alpha}/2$ times. Every time we move down in the refutation's graph at least once, therefore its depth is at least $n^{1-\alpha}/2$. ◀

▶ **Corollary 15.** *Dag-like and tree-like* OBDD($\wedge$) *and* OBDD($\wedge$, *reordering*) *proofs cannot be balanced i.e. there is no polynomial p such that for every unsatisfiable formula φ and for every its refutation w (dag-like or tree-like) of the size S, there exists a refutation w' (dag-like or tree-like respectively) of the size $p(S)$ and of the depth $O(\log(S))$.*

Proof. By Lemma 5 Formula $\mathrm{Peb}(\mathrm{Grid}_n)$ has tree-like OBDD($\wedge$) refutation of the size $O(n^4)$. But $\mathrm{Peb}(\mathrm{Grid}_n)$ cannot have OBDD($\wedge$, reordering) a refutation of the size $\mathrm{poly}(n)$ and of the depth $O(\log \mathrm{poly}(n)) = O(\log n)$ due to Theorem 12. Hence the proof systems are not balanced. ◀

References

1 Albert Atserias, Maria Luisa Bonet, and Jordi Levy. On Chvatal rank and cutting planes proofs. *Electron. Colloquium Comput. Complex.*, TR03-041, 2003. arXiv:TR03-041.

2 Albert Atserias, Phokion G. Kolaitis, and Moshe Y. Vardi. Constraint propagation as a proof system. 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 77–91. Springer, 2004. doi:10.1007/978-3-540-30201-8_9.

3 Paul Beame, Henry A. Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *J. Artif. Intell. Res.*, 22:319–351, 2004. doi:10.1613/jair.1410.

4 Eli Ben-Sasson and Jakob Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. In Bernard Chazelle, editor, *Innovations in Computer Science – ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 401–416. Tsinghua University Press, 2011. URL: http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/3.html.

5 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, March 2001. doi:10.1145/375827.375835.

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

7 Sam Buss, Dmitry Itsykson, Alexander Knop, and Dmitry Sokolov. Reordering rule makes OBDD proof systems stronger. *Electron. Colloquium Comput. Complex.*, TR18-041, 2018. arXiv:TR18-041.

8 Donald Chai and Andreas Kuehlmann. A fast pseudo-boolean constraint solver. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 24(3):305–317, 2005. doi:10.1109/TCAD.2004.842808.

9 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979. doi:10.2307/2273702.

10 William J. Cook, Collette R. Coullard, and György Turán. On the complexity of cutting-plane proofs. *Discret. Appl. Math.*, 18(1):25–38, 1987. doi:10.1016/0166-218X(87)90039-4.

11 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bull. Amer. Math. Soc.*, 43(04):439–562, August 2006. doi:10.1090/s0273-0979-06-01126-8.

12 Dmitry Itsykson, Alexander Knop, Andrei E. Romashchenko, and Dmitry Sokolov. On obdd-based algorithms and proof systems that dynamically change the order of variables. *J. Symb. Log.*, 85(2):632–670, 2020. doi:10.1017/jsl.2019.53.

13 Christoph Meinel and Anna Slobodová. On the complexity of constructing optimal ordered binary decision diagrams. In Igor Prívara, Branislav Rovan, and Peter Ruzicka, editors, *Mathematical Foundations of Computer Science 1994, 19th International Symposium, MFCS'94, Kosice, Slovakia, August 22–26, 1994, Proceedings*, volume 841 of *Lecture Notes in Computer Science*, pages 515–524. Springer, 1994. doi:10.1007/3-540-58338-6_98.

14 Christoph Meinel and Thorsten Theobald. *Algorithms and Data Structures in VLSI Design: OBDD – Foundations and Applications.* Springer, 1998. URL: http://www.informatik.uni-trier.de/%7Emeinel/books/obddbook.html.

15 Pavel Pudlák and Samuel R. Buss. How to lie without being (easily) convicted and the length of proofs in propositional calculus. In Leszek Pacholski and Jerzy Tiuryn, editors, *Computer Science Logic, 8th International Workshop, CSL '94, Kazimierz, Poland, September 25-30, 1994, Selected Papers*, volume 933 of *Lecture Notes in Computer Science*, pages 151–162. Springer, 1994. doi:10.1007/BFb0022253.

16 Alasdair Urquhart. The depth of resolution proofs. *Stud Logica*, 99(1-3):349–364, 2011. doi:10.1007/s11225-011-9356-9.

A Proof of Lemma 5

▶ **Proposition 16.** *Let $X = \{x_1, \dots, x_n\}$ be a set of Boolean variables and let $Y \subset X$. Let $\varphi = \bigwedge_{y \in Y} y$. Then there exists π-OBDD for φ of the size $O(|Y|)$ for every order of variables π.*

Proof. We enumerate Y according to the order π: $Y = \{y_1, \dots, y_{|Y|}\}$. We define π-OBDD for φ as follows: there is the unique node y_i for every $i \in [|Y|]$. We identify node and its label. The node y_1 is the source. For every $i \in [|Y|]$, y_i's outgoing edge labeled with 0 goes to the sink labeled with 0. If $i < |Y|$ then y_i's outgoing edge labeled with 1 goes to y_{i+1} otherwise it goes to the sink labeled with 1. It is easy to see that there is a unique path between the source and the sink, labeled with 1, and that all edges on the path are labeled with 1. Hence the π-OBDD we have defined represents φ. ◀

▶ **Lemma 5** ([7]). *For every directed acyclic graph G and for every order of variables π formula* Peb(G) *has tree-like π-OBDD($\wedge$) refutation of size $O(|V|^2)$ and depth $O(|V|)$.*

Proof. Let $\{A_1, \dots, A_{n_1}\}$ be the first-type clauses in topological sort order. Let B be an arbitrary second-type clause. Consider the following sequence of the CNF formulas: $A_1,\ A_1 \wedge A_2, \dots, \bigwedge_{i=1}^{n_1} A_i,\ (\bigwedge_{i=1}^{n_1} A_i) \wedge B$. Represent each of this formulas as π-OBDD. Then it is easy to see that the sequence of OBDDs is a tree-like π-OBDD($\wedge$) refutation. We now prove that the refutation has the size $O(|V|^2)$. It consists of $|V| + 1$ formulas. It is sufficient to prove that each formula has the size $O(|V|)$. We consider two cases:

Case 1: The formula is $A_1 \wedge \ldots \wedge A_i$ for some $i \in [n_1]$. We prove by the induction on i that $A_1 \wedge \ldots \wedge A_i \equiv v_1 \wedge \ldots \wedge v_i$, where $\equiv$ stands for logical equivalence. **Base:** $i = 1$ so v_1 is a source and the corresponding clause is (v_1). **Induction step:** Assume that $A_1 \wedge \ldots \wedge A_i \equiv v_1 \wedge \ldots \wedge v_i$ then $A_1 \wedge \ldots \wedge A_i \wedge A_{i+1} \equiv v_1 \wedge \ldots \wedge v_i \wedge A_{i+1}$. Let $A_{i+1} = (u_1, \ldots, u_k \to v_{i+1})$ where $u_1, \ldots, u_k$ is the set of all immediate predecessors of v_{i+1}. Since the clauses $\{A_i \mid i \in [n_1]\}$ appear in the topological sort order then the first-type clauses that correspond to the variables $\{u_1, \ldots, u_k\}$ lie in $\{A_1, \ldots, A_i\}$. Then it is easy to see that $v_1 \wedge \ldots \wedge v_i \wedge (u_1, \ldots, u_k \to v_{i+1}) \equiv v_1 \wedge \ldots \wedge v_i \wedge v_{i+1}$.

Proposition 16 implies that such formulas have OBDD representation of the size $O(|V|)$.

Case 2: The formula is $(\bigwedge_{i=1}^{n_1} A_i) \wedge B$. We already have proved that $(\bigwedge_{i=1}^{n_1} A_i) \equiv \bigwedge_j v_j$ is the conjunction of the all nodes. This conjunction implies that the values of variables of all nodes equal 1. At the same time the clause B implies that the variable of the corresponding sink equals 0. Hence the formula is unsatisfiable and the corresponding OBDD is constant false. ◀

B Proof of Lemma 8

▶ **Lemma 8** (Folklore). *Let G be a directed acyclic graph with only one sink. Then* Peb(G) *minimal unsatisfiable i.e. a conjunction of every proper subset of its set of clauses is satisfiable.*

Proof. Fix some proper subset S of the set of all clauses of Peb(G). We now prove that $\bigwedge_{C \in S} C$ is satisfiable.

Consider two cases:

Case 1: There is no second type clause in S. Then each clause from S contains literal without negation. Then the assigment of all 1 is satisfiable.

Case 2: There is no first-type clause in S. Denote the corresponding variable by v. Denote by t the unique sink of the graph. Note that there is a path from v to t (otherwise there are at least two sinks in the graph). Then the assignment of 0 to the path's variables and 1 to the other variables is satisfiable. Indeed, the first-type clauses corresponding to the variables substituted with 1 are always satisfied. The second-type clause is satisfied since the sink is substituted with 0. The other variables substituted with 0 have at least one immediate predecessor substituted with 0, hence their first type clauses are also satisfied. ◀