Resolution Simulates Ordered Binary Decision Diagrams
for Formulas in Conjunctive Normal Form

Olga Tveretina
School of Computer Science
University of Hertfordshire
United Kingdom
o.tveretina@herts.ac.uk

Abstract. A classical question of propositional logic is one of the shortest proof of a tautology. A related fundamental problem is to determine the relative efficiency of standard proof systems, where the relative complexity is measured using the notion of polynomial simulation.

Presently, the state-of-the-art satisfiability algorithms are based on resolution in combination with search. An Ordered Binary Decision Diagram (OBDD) is a data structure that is used to represent Boolean functions.

Groote and Zantema have proved that there is exponential separation between resolution and a proof system based on limited OBDD derivations. However, formal comparison of these methods is not straightforward because OBDDs work on arbitrary formulas, whereas resolution can only be applied to formulas in Conjunctive Normal Form (CNFs).

Contrary to popular belief, we argue that resolution simulates OBDDs polynomially if we limit both to CNFs and thus answer negatively the open question of Groote and Zantema whether there exist unsatisfiable CNFs having polynomial OBDD refutations and requiring exponentially long resolution refutations.

1 Introduction

Propositional proof complexity is the study of the lengths of proofs of statements expressed as propositional formulas. It is tightly connected in many ways to computational complexity, classical proof theory and practical questions of automated deduction.

A classical question of propositional proof complexity is one of the shortest proof of a tautology. A related fundamental problem is to determine the relative efficiency of standard proof systems, where the relative complexity is measured using the notion of polynomial simulation.

Proof systems. Propositional proof systems were defined by Cook and Reckhow as polynomial-time functions which have as their range the set of all tautologies [1]. They also noticed that if there is no propositional proof system that admits proofs polynomial in size of the input formula then the complexity classes NP and co-NP are different, and hence P ≠ NP.

In [2], Atserias, Kolaitis and Vardi generalised the notion of a refutational propositional proof system, viewing it as a special case of constraint propagation. Their proof system consists of the following four rules: (1) Axiom defines the initial set of constraints; (2) Join combines two constraints by intersecting two relations and extending them to all variables occurring in either one of them; (3) Projection computes the projection of a constraint which is the existential quantification; (4) Weakening relaxes the constraint by enlargeing its relation.
This generalisation brings the methods of constraint propagation to the area of proof complexity. On the other hand it introduces new classes of proof systems and on the other hand the existing refutational proof systems can be viewed as a special case of this constraint propagation system.

Efficiency of proof systems. One of the most fundamental problems in the area of propositional proof complexity is to determine the relative efficiency of standard proof systems as it has been introduced by Cook and Reckhow in [6] who found it useful to separate the idea of providing a proof from that being efficient.

Proof systems are compared according to their strength using the notion of polynomial simulation. A proof system $S_1$ simulates polynomially a proof system $S_2$ if every tautology has proofs in $S_1$ of size at most polynomially larger than in $S_2$. Proof systems $S_1$ and $S_2$ are equivalent if they simulate each other polynomially.

Although substantial progress has been made in determining the relative complexity of proof systems and in proving strong lower bounds for some relatively weak proof systems, some major problems still remain unsolved.

As it is defined by Razborov in [13], the question of existence of an efficient proof has to be separated from another important question how to find such a proof efficiently and whether this search adds substantially to the inherent complexity of finding the shortest proof in a specific proof system. It is formalised by the notion of automatizability: a proof system is automatizable if it produces a proof of a tautology in time polynomial in the size of its smallest proof [3].

The current proof systems of practical use are not automatizable (or just weakly automatizable). That is why in addition to lower bounds, there is practical interest in understanding relative efficiency of somewhat weaker proof systems. Interesting examples of such systems are those based either on general resolution or on classical OBDDs not utilising existential quantification.

Resolution versus OBDDs. In the automated reasoning community resolution and OBDDs are popular techniques for solving the propositional satisfiability problem abbreviated as SAT. In fact, both resolution and OBDDs are families of algorithms, where each corresponds to a specific way of making choices.

Resolution underlies the vast majority of all proof search techniques in this area. For example, the DPLL algorithm [7], as well as the clause learning methods are highly optimised implementations of resolution [12]. It has been shown in [13] that modern SAT solvers simulate resolution polynomially.

An OBDD is a canonical data structure that is used for the symbolic representation of Boolean functions [5,19]. Aserias et al. introduced and studied a proof system operating with OBDDs as a special case of constraint propagation [2]. They proved that OBDD based refutations polynomially simulate resolution if they utilise existential quantification. That is OBDD based proof systems containing all four rules Axiom, Join, Weakening and Projection are strictly stronger than resolution but they are still exponential [9].

In the following we consider the OBDD proof system which contains just two rules, Axiom and Join. These rules are equivalent to the Apply operator as it is defined in [5].

Benchmark studies show incomparable behaviour of resolution and such OBDD based systems [13]. Groote and Zantema proved that resolution and OBDDs do not simulate each other polynomially on arbitrary inputs for limited OBDD derivations [5]. Tveretina, Sinz and Zantema strengthened the above result and presented a class of CNFs hard for an arbitrary OBDD derivation and easy for resolution [17].

In general, formal comparison of resolution and OBDDs is not straightforward because the later work on arbitrary formulas, whereas resolution can take as an input only CNFs.
We argue that resolution simulates OBDDs polynomially if we limit both to CNFs. Thus we answer negatively the open question of Groote and Zantema posed in [8] whether there exist unsatisfiable CNFs having polynomial OBDD refutations and requiring exponentially long resolution refutations.

Previous work. There are several works which study the relative efficiency of resolution based and OBDD based proof systems. The most relevant studies to our setting are the following ones.

Peltier shows in [11] that resolution augmented with the extension rule polynomially simulates OBDDs in the following sense: for any unsatisfiable formula \( \varphi \) there exists a refutation of \( \varphi \) with the size polynomially bounded by the maximal size of the reduced OBDDs corresponding to the subformulas occurring in \( \varphi \). As mentioned before, Atserias et al prove in [2] that OBDD based refutations utilising existential quantification polynomially simulate resolution; moreover they are exponentially stronger. Groote and Zantema construct in [8] biconditional formulas that have short OBDD refutations and after transforming them into CNFs they require exponentially long resolution proofs.

Main result. We show that for any unsatisfiable CNF \( \varphi \) there exists a resolution refutation of \( \varphi \) with the size polynomially bounded by the size of an OBDD based refutation of \( \varphi \) if it consists of two rules Axiom and Join. We now formally state the theorem.

**Theorem 1.** Assume an unsatisfiable CNF \( \varphi \). If there is an OBDD refutation of \( \varphi \) with two rules Axiom and Join of size \( n \) then there is a resolution refutation of \( \varphi \) of size \( O(n^2) \).

Our main argument is based on the idea that the elimination rule can be simulated by applying the resolution rule on the variable corresponding to the eliminated node [11]. But we use it differently.

We strengthen this idea and prove that the number of resolution steps corresponding to the elimination of a node is bounded by the number of clauses in the input CNF \( \varphi \). Moreover, we show that it is an invariant property: although resolution steps generate new clauses, the number of resolution steps needed to simulate elimination of a node in an intermediate OBDD remains bounded by the number of clauses of \( \varphi \) encoded by this OBDD. Furthermore, we show that it is sufficient to simulate only the elimination rule, that is the merging rule plays no role in the context.

The remainder of the paper. We give the necessary background in Section 2. In Section 3 we introduce two proof systems of interest: one is based on resolution and the other corresponds to OBDD derivations based on the Axiom and Join rules. In Section 4 we show how to simulate the elimination rule using resolution and in Section 5 we prove our main result. Section 6 contains concluding remarks.

2 Definitions

2.1 Propositional Logic and Conjunctive Normal Forms

In this section we recall some basic notations about propositional logic and only provide a short overview of the main definitions.

In the following we consider propositional formulas in Conjunctive Normal Form (CNF) built using variables from a set \( \text{var} \). A literal \( l \) is either a variable \( x \) or its negation \( \neg x \) with \( \text{var}(l) = x \). A clause \( C \) is a disjunction of literals, and a CNF \( \varphi \) is a conjunction of clauses. By \( \text{Cnf} \) we denote the set of all CNFs.

We define \( \text{Cls}(\varphi) \) to be the set of clauses, \( \text{Lit}(\varphi) \) the set of literals, and \( \text{var}(\varphi) \) the set of variables contained in the CNF \( \varphi \).
A truth assignment is a function $A : \text{var} \to \{\text{true}, \text{false}\}$. We denote by $\mathcal{A}$ the set of all possible assignments. The truth values of literals, clauses and CNFs are defined in a standard way.

We write $A \models \varphi$ if $\varphi$ evaluates to true for the assignment $A$, otherwise we write $A \not\models \varphi$. We say that $\varphi$ is unsatisfiable if $A \not\models \varphi$ for any $A \in \mathcal{A}$, otherwise it is satisfiable; $\varphi$ is a tautology if $A \models \varphi$ for any $A \in \mathcal{A}$.

We say that two CNFs $\varphi$ and $\psi$ are logically equivalent, denoted $\varphi \equiv \psi$, if $A \models \varphi$ if and only if $A \models \psi$ for any $A \in \mathcal{A}$.

We use $\top$ for the empty set of clauses and $\bot$ for the CNF consisting of the empty clause. By definition the empty clause is unsatisfiable that is it is equivalent to false, and the the empty set of clauses is equivalent to true.

### 2.2 Ordered Binary Decision Diagrams

The concept of ordered binary decision diagrams (OBDDs) was first proposed by Lee in [10] as a means to represent propositional formulas (Boolean functions) compactly as directed acyclic graphs (DAGs). Then it was further developed to a data structure by Acers [1] and Boute [4], and subsequently by Bryant [5].

**Definition 1 (An OBDD).** An OBDD $B$ is a directed acyclic graph satisfying the following:

1. it has a unique node called the root and denoted by $\text{root}(B)$;
2. each inner node $p$ is labeled by the propositional variable $\text{var}(p)$ and has exactly two successors, a false-successor and a true-successor;
3. the inner nodes build the set $\text{Node}(B)$, and the labels build the set $\text{var}(B)$;
4. each leaf node is labeled by either true or false;
5. there is a total variable order $\prec$ such that for each transition from the inner node with label $x$ to the inner node with label $y$ we have that $x \prec y$.

We use $\text{high}(p)$ and $\text{low}(p)$ to denote the OBDDs rooted at the true-successor and the false-successor of $p$; $|B|$ is the size of $B$, that is the number of its inner nodes.

We use $B_1 \equiv B_2$ to denote that $B_1$ and $B_2$ are isomorphic OBDDs defined as follows:

- both $B_1$ and $B_2$ consist either of the node true or of the node false;
- $\text{var}(\text{root}(B_1)) = \text{var}(\text{root}(B_2))$, $\text{high}(B_1) \equiv \text{high}(B_2)$ and $\text{low}(B_1) \equiv \text{low}(B_2)$.

OBDD operations are applicable only to OBDDs that respect the same variable ordering. To shorten the notations in the rest of the paper, we assume without explicitly stating that all the variables agree on the common variable order $x_1 \prec x_2 \prec x_3 \prec \ldots$ when considering different OBDDs in the same context.

**Definition 2 (Path).** A path of an OBDD $B$ is a sequence $\alpha = l_1 \ldots l_k$ of literals with $k \geq 1$ such that there are $p_1, \ldots, p_k \in \text{Node}(B)$, where

- $p_1 = \text{root}(B)$;
- for $1 \leq i < k$, either $p_{i+1} = \text{root}(\text{high}(p_i))$ and $l_i = \text{var}(p_i)$, or $p_{i+1} = \text{root}(\text{low}(p_i))$ and $l_i = \neg \text{var}(p_i)$;
- $\text{high}(p_k) \in \{\text{false}, \text{true}\}$ if $l_k = \text{var}(p_k)$;
- $\text{low}(p_k) \in \{\text{false}, \text{true}\}$ if $l_k = \neg \text{var}(p_k)$. 
We use $\text{Path}(B)$ to denote the set of all paths of $B$, and $\text{Path}^f(B)$ to denote the set of all paths that go to the false node. By $\text{Path}(p)$ we mean the set of all paths that go through the inner node $p$, and $\text{Path}^f(p) = \text{Path}(p) \cap \text{Path}^f(B)$.

A path can be seen as a conjunction of literals. In this way, each path is a conjunction of literals. In this way, each path $\alpha \notin C$ for a clause $C$ if $A \models C$ for any $A \in \mathcal{A}(\alpha)$. Where it is convenient, we see a path as a set of literals and use the set notations.

We say that a CNF $\varphi$ and an OBDD $B$ are logically equivalent if for every assignment $A \in \mathcal{A}$, $\alpha \models \varphi$ if and only if there is a path $\alpha \in \text{Path}^f(B)$ such that $A \in \mathcal{A}(\alpha)$.

For any CNF $\varphi$ and OBDD $B$, we use $\varphi \leq B$ to denote that $\varphi$ and $B$ are logically equivalent and for each path $\alpha \in \text{Path}^f(B)$ there is a clause $C \in \text{Cls}(\varphi)$ such that $\alpha \models C$.

### 2.3 OBDD Construction

The straightforward way to construct an OBDD is to start with a binary decision tree and then incrementally eliminate redundancies and identify identical subtrees. The other more efficient way follows the structure of the propositional formula. Such algorithms start with building OBDDs for variables or literals, and then construct more complex OBDDs by using OBDD operations for logical connectives.

Algorithm 1 presented below (from [16]) takes as an input two OBDDs $B_1$ and $B_2$ and returns their conjunction denoted by $B_1 \land B_2$. It proceeds from the root downward creating vertices in the resulting graph as follows:

1. the function $\text{Dec}$ decomposes a non-terminal OBDD node into its constituent components, that is its variable and cofactors;
2. the function $\text{Node}$ constructs a new OBDD node if it is not already present, and otherwise returns the already existent node.

Lemma 1 is a technical lemma which will be used to prove Lemma 1. It follows relatively straightforwardly from the definition of Algorithm 1.

**Lemma 1.** Assume OBDDs $B_1$ and $B_2$ such that $\text{var}(B_1) \neq \emptyset$ and $\text{var}(B_2) \neq \emptyset$. Then Algorithm 1 returns the OBDD $B_1 \land B_2$ such that

- for each $\alpha \in \text{Path}^f(B_1 \land B_2)$ there is $\beta \in \text{Path}^f(B_1) \cup \text{Path}^f(B_2)$ such that $\beta \subseteq \alpha$;
- for each $\beta \in \text{Path}^f(B_1) \cup \text{Path}^f(B_2)$ there is $\alpha \in \text{Path}^f(B_1 \land B_2)$ such that $\beta \subseteq \alpha$.

**Proof.** We give a proof by induction on $k = |\text{var}(B_1) \cup \text{var}(B_2)|$. For the basis step we choose $k = 0$ and the lemma trivially holds.

We assume that the lemma holds for any $B'_1$ and $B'_2$ such that $|\text{var}(B'_1) \cup \text{var}(B'_2)| \leq k - 1$. Let $|\text{var}(B_1) \cup \text{var}(B_2)| = k$ and $x \in \text{var}(B_1) \cup \text{var}(B_2)$ be the smallest variable. Then by the definition of an OBDD $|\text{var}(\text{high}(B_1 \land B_2))| \leq k - 1$ and $|\text{var}(\text{low}(B_1 \land B_2))| \leq k - 1$.

If $x \in \text{var}(B_1)$ and $x \notin \text{var}(B_2)$ then by the definition of Algorithm 1 it returns the OBDD $B_1 \land B_2$ such that

- $\text{var(root)} = x$;
- $\text{high}(B_1 \land B_2) = \text{high}(B_1) \land \text{high}(B_2)$ and $\text{low}(B_1 \land B_2) = \text{low}(B_1) \land \text{low}(B_2)$.

If $x \in \text{var}(B_1)$ and $x \notin \text{var}(B_2)$ then by the definition of Algorithm 1 it returns the OBDD $B_1 \land B_2$ such that

- $\text{var(root)} = x$;
Algorithm 1: The algorithm for constructing $B_1 \land B_2$

We use the induction hypothesis and conclude that the lemma holds.

The following corollary is a direct consequence of Lemma 1 and it will be used later to prove the main result.

**Corollary 1.** Assume CNFs $\varphi_1$ and $\varphi_2$ and OBDDs $B_1$ and $B_2$ such that $\varphi_1 \preceq B_1$ and $\varphi_2 \preceq B_2$. Then Algorithm 1 returns the OBDD $B_1 \land B_2$ such that $\varphi_1 \land \varphi_2 \preceq B_1 \land B_2$.

### 2.4 Reduction Rules

There are two reduction rules not affecting the semantics of OBDDs that can be used to reduce the size of the OBDDs constructed by Algorithm 1 defined as follows.

- **Merging:** If $\text{low}(p) \equiv \text{low}(q)$ and $\text{high}(p) \equiv \text{high}(q)$ for $p, q \in \text{Node}(B)$ then the node $p$ can be removed. Any link to the node $p$ is replaced by a link to the node $q$.

- **Elimination:** If $\text{low}(p) \equiv \text{high}(p)$ for $p \in \text{Node}(B)$ then the node $p$ can be removed. Any link to the node $p$ is replaced by a link to the root of $\text{high}(p)$. We write $B \rightarrow_p B'$ if $B'$ is obtained from $B$ by eliminating the node $p$.

Let $B \rightarrow_p B'$. For simplicity, we define informally what we mean by $q \rightarrow_p q'$. The OBDD $B'$ contains in fact the same nodes as $B$ except the node $p$. We write $q \rightarrow_p q'$ to denote that the node $q' \in \text{Node}(B')$ is in fact the node $q \in \text{Node}(B)$ after renaming as $q'$.

We use $B^+$ to denote the reduced OBDD obtained from $B$, that is no reduction rule can be applied to $B^+$ any more.
Lemma 2 (Bryant [5]). If $B_1$ and $B_2$ are logically equivalent OBDDs then $B_1^i \equiv B_2^i$.

The total time complexity of the algorithm is $O(|B_1| \times |B_2|)$. In the worst case, the upper bound is achieved and $B_1 \land B_2$ can contain $O(|B_1| \times |B_2|)$ nodes.

Theorem 2 (Bryant [5]). Let $B_1$ and $B_2$ be two reduced OBDDs, that is $B_1 \equiv B_1^1$ and $B_2 \equiv B_2^j$. Then the size of $B_1 \land B_2$ is $O(|B_1| \times |B_2|)$, and the number of merging and elimination steps to compute the reduced OBDD corresponding to $B_1 \land B_2$ is at most $|B_1 \land B_2|$.

Proof. 1. By definition, any subOBDD of $B_1 \land B_2$ is of the form $B_1' \land B_2'$, where $B_1'$ and $B_2'$ are subOBDDs of $B_1$ and $B_2$ respectively. Hence, the size of $B_1 \land B_2$ is bounded by $O(|B_1| \times |B_2|)$.
2. This is immediate since the elimination and merging rules strictly decrease the number of nodes.

\[ \square \]

3 OBDDs and Resolution as Proof Systems

Proof systems based on resolution and OBDDs are so-called refutational proof systems. A refutation of an unsatisfiable CNF $\varphi$ starts with the clauses of $\varphi$ and derives a contradiction represented by the empty clause $\bot$ for resolution and by the $\text{false}$ node for OBDDs.

Any proof system operating with OBDDs can be seen as an instance of the constraints based proof system. In the following we consider the OBDD proof system which uses two rules, Axiom and Join.

Definition 3 (OBDD refutation). An OBDD refutation of a CNF $\varphi$ is a sequence of OBDDs $B_1, \ldots, B_k$ such that the following holds:

1. Axiom : $B_i \equiv C_i$ with $1 \leq i \leq |\text{Cls}(\varphi)|$;
2. Join : $B_i \equiv B_i^{j'} \land B_i^{j''}$ with $1 \leq j' < j'' < i$ and $|\text{Cls}(\varphi)| < i \leq k$;
3. $B_k^1 \equiv \text{false}$.

The size of the OBDD refutation is defined as $\sum_{i=1}^{k} |B_i|$.

Without loss of generality we can assume that each OBDD is used exactly once, that is if a CNF $\varphi$ consists of $m$ clauses then the number of OBDDs in the OBDD refutation of $\varphi$ is exactly $2m - 1$.

The resolution proof system goes back to Robinson [12] and consists of a single rule. It derives from two clauses $l \lor C$ and $\neg l \lor D$, such that $C$ and $D$ do not contain a complementary literal, the new clause $C \lor D$ called the resolvent of $l \lor C$ and $\neg l \lor D$, and denoted in the following by $\text{res}(l \lor C, \neg l \lor D)$.

When we write $\text{res}(C, D)$, we assume that there is a literal $l$ such that $l \in \text{Lit}(C)$ and $\neg l \in \text{Lit}(D)$; moreover, the clauses $C$ and $D$ contain no other complimentary literals.

Definition 4 (Resolution refutation). A resolution refutation of a CNF $\varphi$ of size $k$ is a sequence of clauses $C_1, \ldots, C_k$ such that

1. Axiom : $C_i \in \text{Cls}(\varphi)$ with $1 \leq i \leq |\text{Cls}(\varphi)|$;
2. Join : $C_i = \text{res}(C_{j'}, C_{j''})$ with $1 \leq j' < j'' < i$ and $|\text{Cls}(\varphi)| < i \leq k$;
3. $C_k = \bot$.

We say that $k$ is the size of the resolution refutation. We write $\varphi \vdash_{\text{res}} \psi$ for any CNF $\psi$ such that $\psi = \bigwedge_{i=|\text{Cls}(\varphi)|+1}^{k'} C_i$ with $k' \leq k$. 
4 Simulating OBDDs by Resolution

In the rest of the paper we will show formally that if there is an OBDD refutation of a CNF \( \varphi \) of size \( n \) then there is a resolution refutation of \( \varphi \) of size at most \( n^2 \). The existence of such resolution refutation is based on the following observations:

1. elimination of a node can be simulated by at most \(|\text{Cls}(\varphi)|\) resolution steps;
2. \(|\text{Cls}(\varphi)| \leq n\);
3. the number of nodes which can be removed by the elimination rule is at most \( n \).

Our main argument is based on the idea that the elimination rule can be simulated by applying the resolution rule on the variable corresponding to the eliminated node \([11]\). But we use it differently. We strengthen this idea and prove that the number of resolution steps corresponding to the elimination of a node is bounded by the number of clauses in the input CNF \( \varphi \).

Moreover, we show that it is an invariant property: although resolution steps generate new clauses, the number of resolution steps needed to simulate elimination of a node in an intermediate OBDD remains bounded by the number of clauses of \( \varphi \) encoded by this OBDD. As we will see it later, it is sufficient to simulate only the elimination rule. As the merging rule plays no role in the context, we assume for simplicity that all intermediate OBDDs are merged.

**Example 1.** Before giving technical details, we provide a simple illustrating example. We consider the CNF

\[
\varphi = (x \lor \neg y) \land (y \lor z) \land (y \lor \neg z) \land \neg x
\]

An OBDD refutation of \( \varphi \) is depicted in Figure 1. The OBDDs \( B_6 \) and \( B_7 \) are obtained from \( B_5 \) and \( B_7 \) correspondingly by applying the elimination rule. Consider the following resolution refutation of \( \varphi \):

\[
C_1 = x \lor \neg y, C_2 = y \lor z, C_3 = y \lor \neg z, C_4 = \neg x,
\]
\[C_5 = \text{res}(C_2, C_3) = y, C_6 = \text{res}(C_1, C_5) = x, C_7 = \text{res}(C_4, C_6) = \bot\]

It simulates the OBDD refutation as follows:

– By resolving the clauses \(C_2\) and \(C_3\) we eliminate the occurrences of \(z\). By resolving the clauses \(C_1\) and \(C_5\) we eliminate the occurrences of \(y\). The new set of clauses decodes the OBDD \(B_6\).
– By resolving the clauses \(C_4\) and \(C_6\) we eliminate the occurrences of \(x\).

### 4.1 Notations

Now we define notations that will be used in the rest of the paper. Let \(\varphi\) be a CNF and \(B\) an OBDD such that \(\varphi \preceq B\). We tacitly assume a function

\[F : \text{Path}^f(B) \rightarrow \text{Cls}(\varphi)\]

such that \(\alpha \not\models F(\alpha)\) for an \(\alpha \in \text{Path}^f(B)\). For each node \(p \in \text{Node}(B)\), we define the set

\[\text{Cls}(p, \varphi) = \{C \in \text{Cls}(\varphi) \mid \exists \alpha \in \text{Path}^f(p) : C = F(\alpha)\}\]

For each \(C \in \text{Cls}(\varphi)\) and \(p \in \text{Node}(B)\), we define

\[\gamma(p, C) = |\{\alpha \in \text{Path}^f(p) \mid F(\alpha) = C\}| - 1\]

and

\[\tau(p, \varphi) = \sum_{C \in \text{Cls}(p, \varphi)} \gamma(p, C)\]

Suppose \(B \rightarrow_p B'\) for a node \(p \in \text{Node}(B)\) and an OBDD \(B'\). The set \(\text{Res}(p, \varphi)\) is defined as follows:

\[\text{Res}(p, \varphi) = \{\psi \in \text{Cnf} \mid \varphi \vdash_{\text{res}} \psi \text{ and } \varphi \land \psi \preceq B'\}\]

### 4.2 Technical Background

Lemma 3 and Corollary 2 present technical results which will be used to prove Theorem 4.

**Lemma 3.** Let \(S\) be a finite set such that \(|S| > 0\), and \(B_1, \ldots, B_l \subseteq S\) be a sequence with:

- \(B_l = S\)
- For each \(B_i\), \(1 \leq i < l\), one of the following holds:
  - \(B_i = \{s\}\) for \(s \in S\)
  - \(B_i = B_j \cup B_k\) for some \(j, k\) with \(j < k < i\)

Then \(l = 2|S| - 1\).

**Proof.** We give a proof by induction on \(|S|\). As the basis step we choose \(|S| = 1\). Then the lemma hold as trivially \(l = 1\). Let the lemma hold for any \(|S|\). Assume a set \(S'\) such that \(|S'| = |S| + 1\). Then \(l' = l + 2 = (2|S| - 1) + 2 = 2|S'| - 1\). \(\Box\)

The following corollary follows directly from Lemma 3.

**Corollary 2.** Let the conditions of Lemma 3 hold. Moreover, let \(k = l - 2|S| + 1 > 0\). Then there are \(k\) sets \(B'_1, \ldots, B'_k \in \{B_1, \ldots, B_l\}\) such that for each \(B' \in \{B'_1, \ldots, B'_k\}\) there is \(B \in \{B_1, \ldots, B_l\} \setminus \{B'_1, \ldots, B'_k\}\) such that \(B' = B\).
4.3 Simulation of the Elimination Rule

In this section we show that elimination of a node can be simulated by at most $|\text{Cls}(\varphi)|$ resolution steps, where $\varphi$ is the input unsatisfiable CNF.

**Example 2.** We provide another illustrating example. Consider the CNF $\varphi$ consisting of the following eight clauses:

$$
C_1 = \neg x \lor \neg y \lor \neg v \quad C_2 = \neg x \lor \neg z \lor \neg w \quad C_3 = \neg x \lor y \lor \neg v \quad C_4 = \neg x \lor z \lor \neg w \\
D_1 = x \lor \neg z \lor \neg v \quad D_2 = x \lor \neg y \lor \neg w \quad D_3 = x \lor z \lor \neg v \quad D_4 = x \lor y \lor \neg w
$$

Figure 2 represents the OBDD encoding of $\varphi$, and the mapping of the false-paths onto the clauses of $\varphi$. Elimination of the node labelled with $x$ can be simulated by the following resolution steps:

$$
\text{res}(C_1, D_1) = \neg y \lor \neg z \lor \neg w \quad \text{res}(C_2, D_2) = \neg y \lor z \lor \neg w \quad \text{res}(C_1, D_3) = \neg y \lor \neg z \lor \neg w \\
\text{res}(C_4, D_2) = \neg y \lor z \lor \neg w \quad \text{res}(C_3, D_1) = y \lor \neg z \lor \neg w \quad \text{res}(C_2, D_4) = \neg y \lor z \lor \neg w \\
\text{res}(C_3, D_3) = y \lor \neg z \lor \neg w \quad \text{res}(C_4, D_4) = y \lor z \lor \neg w
$$

Theorem 3 demonstrates that elimination of a node $p$ can be simulated by at most $k/2$ resolution steps, where $k$ is the number of paths going through $p$ to the false-node. This is a variant of Lemma 2 from [11] which serves our needs better.

**Theorem 3.** Assume a CNF $\varphi$ and an OBDD $B$ such that $\varphi \leq B$. Let $B \rightarrow_p B'$ for a node $p \in \text{Node}(B)$ and an OBDD $B'$. Then there is a CNF $\psi \in \text{Res}(p, \varphi)$ such that

$$
|\text{Cls}(\psi)| \leq |\text{Path}^f(p)|/2
$$

**Proof.** Assume that $x = \text{var}(p)$. Let $\psi$ be the smallest CNF satisfying the following: for all $\alpha$ and $\beta$ such that $\alpha.x.\beta, \alpha.\neg x.\beta \in \text{Path}^f(p)$,

$$
\neg(\exists C \in \text{Cls}(\varphi) : \alpha.\beta \not\models C) \implies \text{res}(F(\alpha.x.\beta), F(\alpha.\neg x.\beta)) \in \text{Cls}(\psi)
$$

By construction, $\psi \in \text{Res}(p, \varphi)$ and $|\text{Cls}(\psi)| \leq |\text{Path}^f(p)|/2$. \qed
The subsequent statements improve the upper bound on the number of resolution steps needed to simulate elimination of an arbitrary node. Namely, we show that the number of resolution steps sufficient to simulate elimination of a node \( p \) is bounded by \(|\text{Cls}(\varphi)|\).

**Theorem 4.** Let \( \varphi \) be a CNF and \( B \) be an OBDD such that \( \varphi \leq B \). Suppose \( B \rightarrow_p B' \) for some \( p \in \text{Node}(B) \) and \( B' \). Then there is a CNF \( \psi \in \text{Res}(p, \varphi) \) such that

\[ |\text{Cls}(\psi)| \leq |\text{Cls}(\varphi)| \]

**Proof.** If \(|\text{Cls}(\varphi)| \geq |\text{Path}^f(p)|/2 \) then the theorem holds by Theorem 3. We assume that

\[ |\text{Cls}(\varphi)| < |\text{Path}^f(p)|/2 \]

Let \( m = |\text{Path}^f(p)|/2 \), \( m' = |\text{Cls}(\varphi)| \), \( p^h = \text{root}(\text{high}(p)) \), \( p^l = \text{root}(\text{low}(p)) \) and \( x = \text{var}(p) \).

Let \( \overline{B} \equiv \text{high}(p) \equiv \text{low}(p) \) with \( \overline{p} = \text{root}(\overline{B}) \) be a subOBDD of \( B' \) and let \( \overline{N} = \text{Node}(\overline{B}). \)

In the rest of the proof we treat the roots of isomorphic subOBDDs and the false-nodes as distinct nodes if they are reachable by different paths (see Figure 2(b)). That is by Lemma 3 \(|\overline{N}| = m - 1 \). Let \( \overline{P} = \text{Path}^f(\overline{p}) \). By construction,

- \( \alpha, \beta \in \overline{P} \) if and only if there are \( \alpha, x, \beta, \alpha, \neg x, \beta \in \text{Path}^f(p) \) for some \( \alpha \) and \( \beta \);
- \(|\overline{P}| = m \).

We construct now three sequences of sets which correspond to the sets of Lemma 3 and Corollary 2.

1. We define the sets \( S, S_1, \ldots, S_l \) with \( l = 2m - 1 \) as follows:
   - \( S = \overline{P} \);
   - \( S_i = \{ s_i \} \) for \( s_i \in S \) with \( 1 \leq i \leq |S| \);
   - \( S_i = S_j \cup S_k \) with \( i > j > k \) and \( i > |S| \);
   - \( S_1 = S \).

Moreover, we choose the \( m - 1 \) sets \( S_{m+1}, \ldots, S_{2m-1} \) as follows: if \( S_i = S_j \cup S_k \) then there are \( q_i, q_j, q_k \in \overline{N} \) such that
   - \( S_i = \text{Path}^f(q_i) \);
   - \( q_j = \text{high}(q_i) \) and \( q_k = \text{low}(q_i) \).

2. We define the sets \( S', S'_1, \ldots, S'_l \) with \( l = 2m - 1 \) as follows:
   - \( S' = \text{Cls}(p^h, \varphi) \);
   - \( S'_i = \{ s'_i \} \) for \( s'_i \in S' \) with \( 1 \leq i \leq m \);
   - \( S'_i = S'_j \cup S'_k \) with \( i > j > k \) and \( m + 1 \leq i \leq 2m - 1 \).

We choose the \( m - 1 \) sets \( S'_{m+1}, \ldots, S'_{2m-1} \) as follows: if \( S'_i = S'_j \cup S'_k \) then there are \( q'_i, q'_j, q'_k \in \text{Node}(\text{high}(p)) \) such that
   - \( S'_i = \text{Cls}(q'_i, \varphi) \);
   - \( q'_j = \text{high}(q'_i) \) and \( q'_k = \text{low}(q'_i) \).

3. We define the sets \( S'', S''_1, \ldots, S''_l \) with \( l = 2m - 1 \) as follows:
   - \( S'' = \text{Cls}(p^l, \varphi) \);
   - \( S''_i = \{ s''_i \} \) for \( s''_i \in S'' \) with \( 1 \leq i \leq m \);
   - \( S''_i = S''_j \cup S''_k \) with \( i > j > k \) and \( m + 1 \leq i \leq 2m - 1 \).

Now we choose the \( m - 1 \) sets \( S''_{m+1}, \ldots, S''_{2m-1} \) as follows: if \( S''_i = S''_j \cup S''_k \) then there are \( q''_i, q''_j, q''_k \in \text{Node}(\text{low}(p)) \) such that
   - \( S''_i = \text{Cls}(q''_i, \varphi) \);
\begin{itemize}
  \item $q''_j = \text{high}(q''_j)$ and $q''_k = \text{low}(q''_k)$.
\end{itemize}

It follows from $\overline{B} \equiv \text{high}(p) \equiv \text{low}(p)$ and Corollary \ref{corollary} that there are $m - m'$ paths
\[
\alpha_i, \ldots, \alpha_{i_{m-m'}} \in \overline{P}
\]
such that for each $\alpha \in \{\alpha_i, \ldots, \alpha_{i_{m-m'}}\}$ there is $\alpha' \in P \setminus \{\alpha_i, \ldots, \alpha_{i_{m-m'}}\}$
such that $\alpha, \alpha' \not\equiv \text{res}(C, D)$ for some $C, D \in \text{Cls}(\varphi)$. It implies that there is a CNF $\psi \in \text{Res}(p, \varphi)$ such that
\[
|\text{Cls}(\psi)| \leq |\text{Path}^f(p)|/2 - (m - m')
\]
Hence, we obtain that there is a CNF $\psi \in \text{Res}(p, \varphi)$ such that $|\text{Cls}(\psi)| \leq |\text{Cls}(\varphi)|$. \qed

Lemma \ref{lemma4} follows straightforwardly from Theorem \ref{theorem}.

**Lemma 4.** Let $\varphi$ be a CNF and $B$ be an OBDD such that $\varphi \subseteq B$. Suppose $B \rightarrow_p B'$ for some $p \in \text{Node}(B)$ and $B'$. Then there is a $\psi \in \text{Res}(p, \varphi)$ such that
\[
|\text{Cls}(\psi)| \leq |\text{Path}^f(p)| - \tau(p, \varphi)
\]

**Proof.** By the definition of $\tau(p, \varphi)$, we obtain that $|\text{Cls}(p, \varphi)| = |\text{Path}^f(p)| - \tau(p, \varphi)$. As $|\text{Cls}(p, \varphi)| \leq |\text{Cls}(\varphi)|$, we conclude that $|\text{Cls}(\psi)| \leq |\text{Path}^f(p)| - \tau(p, \varphi)$. \qed

\subsection{4.4 Invariant}

In this section we show that although resolution steps generate new clauses, the number of resolution steps needed to simulate elimination of a node in an intermediate OBDD remains bounded by the number of clauses of $\varphi$ encoded by this OBDD.

**Theorem 5.** Assume a CNF $\varphi$ and an OBDD $B$ with $\varphi \subseteq B$. Suppose $B \rightarrow_p B'$ and $q \rightarrow_p q'$ for $p, q \in \text{Node}(B)$ and $q' \in \text{Node}(B')$. Then there is a $\varphi' \in \text{Res}(p, \varphi)$ such that
\[
|\text{Path}^f(q')| - \tau(q', \varphi \land \varphi') \leq |\text{Path}^f(q)| - \tau(q, \varphi)
\]

**Proof.** We consider the following cases:

\begin{itemize}
  \item Suppose the nodes $p$ and $q$ are not connected by a path. Then removing the node $p$ does not affect the false-paths that go through the node $q$. That is, $\text{Path}^f(q') = \text{Path}^f(q)$ and $\tau(q', \varphi \land \varphi') = \tau(q, \varphi)$. Hence,

\[
|\text{Path}^f(q')| - \tau(q', \varphi \land \varphi') = |\text{Path}^f(q')| - \tau(q', \varphi) = |\text{Path}^f(q)| - \tau(q, \varphi)
\]

For both cases below we assume that the nodes $p$ and $q$ are connected by a path.

\item Let $\text{var}(p) \prec \text{var}(q)$.

From $\text{high}(p) \equiv \text{low}(p)$ we obtain that there is a path $x.\alpha$ from $p$ to $q$ if and only if there is a path $\neg x.\alpha$ from $p$ to $q$. After eliminating the node $p$, $x.\alpha$ and $\neg x.\alpha$ will be replaced with $\alpha$. Hence,
\[
|\text{Path}^f(q')| = |\text{Path}^f(q)|/2
\]
We observe that
\[
\tau(q, \varphi) - \tau(q', \varphi \land \varphi') \leq |\text{Path}^f(q)|/2
\]
and obtain
\[
|\text{Path}^f(q')| - \tau(q', \varphi \land \varphi') \leq |\text{Path}^f(q)| - \tau(q, \varphi)
\]
Let \( \text{var}(q) \prec \text{var}(p) \). then the OBDD rooted at the \( p \) node is a subOBDD of the OBDD rooted at the \( q \) node.

From \( \text{high}(p) \equiv \text{low}(p) \) we obtain that there is a path \( \alpha.x \) from \( q \) to \( p \) if and only if there is a path \( \alpha.\neg x \) from \( q \) to \( p \). After eliminating the node \( p \), \( \alpha.x \) and \( \alpha.\neg x \) will be replaced with \( \alpha \). Hence,

\[
|\text{Path}_i^f(q')| = |\text{Path}_i^f(q)| - |\text{Path}_i^f(p)|/2
\]

We observe that

\[
\tau(q, \varphi) - \tau(q', \varphi \land \varphi') \leq |\text{Path}_i^f(p)|/2
\]

and obtain

\[
|\text{Path}_i^f(q')| - \tau(q', \varphi \land \varphi') \leq |\text{Path}_i^f(q)| - \tau(q, \varphi)
\]

\( \square \)

**Theorem 6.** Assume CNFs \( \varphi_1 \) and \( \varphi_2 \), and OBDDs \( B_1 \) and \( B_2 \) with \( \varphi_1 \preceq B_1 \) and \( \varphi_2 \preceq B_2 \). Let for any \( q_1 \in \text{Node}(B_1) \) and \( q_2 \in \text{Node}(B_2) \), and some \( k_1, k_2 \geq 0 \)

- \( |\text{Path}_i^f(q_1)| - \tau(q_1, \varphi_1) \leq k_1 \)
- \( |\text{Path}_i^f(q_2)| - \tau(q_2, \varphi_2) \leq k_2 \)

Then Algorithm \( \square \) returns the OBDD \( B_1 \land B_2 \) such that for any \( q \in \text{Node}(B_1 \land B_2) \)

\[
|\text{Path}_i^f(q)| - \tau(q, \varphi_1 \land \varphi_2) \leq k_1 + k_2
\]

**Proof.** Let \( q \), \( q_1 \) and \( q_2 \) be the nodes in the OBDDs \( B_1 \land B_2 \), \( B_1 \) and \( B_2 \) correspondingly reachable by the same partial path.

We recall that by Lemma \( \square \) \( \varphi_1 \land \varphi_2 \preceq B_1 \land B_2 \) and define the sets \( S_1 \) and \( S_2 \) as follows:

- \( S_1 = \{ \alpha \in \text{Path}_i^f(B_1 \land B_2) \mid F(\alpha) \in \text{Cls}(\varphi_1) \} \);
- \( S_2 = \{ \alpha \in \text{Path}_i^f(B_1 \land B_2) \mid F(\alpha) \in \text{Cls}(\varphi_2) \} \).

That is, the set \( S_1 \) contains the false-paths of \( B_1 \land B_2 \) falsified by the clauses of \( \varphi_1 \) and \( S_2 \) contains the false-paths of \( B_1 \land B_2 \) falsified by the clauses of \( \varphi_2 \). Suppose

- \( m_1 = |S_1| - |\text{Path}_i^f(q_1)| \);
- \( m_2 = |S_2| - |\text{Path}_i^f(q_2)| \).

It follows from Lemma \( \square \) and Corollary \( \square \) that

\[
\tau(q, \varphi_1 \land \varphi_2) = \tau(q_1, \varphi_1) + \tau(q_2, \varphi_2) + m_1 + m_2
\]

and therefore

\[
|\text{Path}_i^f(q)| - \tau(q, \varphi_1 \land \varphi_2) = (|\text{Path}_i^f(q_1)| + |\text{Path}_i^f(q_2)| + m_1 + m_2) - \\
(\tau(q_1, \varphi_1) + \tau(q_2, \varphi_2) + m_1 + m_2) \leq k_1 + k_2
\]

\( \square \)

Now we combine Theorems \( \square \) \( \square \) and \( \square \) and Lemma \( \square \) and obtain the following result.

**Corollary 3.** Assume an unsatisfiable CNF \( \varphi \). Let \( B_1, \ldots, B_k \) be an OBDD refutation of \( \varphi \). Then elimination of a node in any OBDD \( B_i \), \( 1 \leq i \leq k \), can be simulated by at most \( |\text{Cls}(\varphi)| \) resolution steps.
5 The Main Result

Now we establish the main result that any OBDD refutation of an unsatisfiable CNF \( \varphi \) of size \( n \) can be simulated by a resolution refutation of \( \varphi \) of size at most \( O(n^2) \).

**Theorem 7.** Assume an unsatisfiable CNF \( \varphi \). If there is an OBDD refutation of \( \varphi \) of size \( n \) then there is a resolution refutation of \( \varphi \) of size \( O(|\text{Cls}(\varphi)| \cdot n) \).

**Proof.** By Corollary 3 elimination of a node can be simulated by at most \( |\text{Cls}(\varphi)| \) steps. Since the OBDD refutation has size \( n \), we obtain that there is a resolution refutation of \( \varphi \) of size \( O(|\text{Cls}(\varphi)| \cdot n) \). \( \square \)

Now, Theorem 1 stating that if there is an OBDD refutation of \( \varphi \) of size \( n \) then there is a resolution refutation of \( \varphi \) of size \( O(n^2) \) follows straightforwardly from Theorem 7.

**Proof.** (Proof of Theorem 1) We can assume without loss of generality that \( \varphi \) is a minimally unsatisfiable CNF, that is removing any clause from \( \varphi \) will result in a satisfiable CNF.

Then the size of any OBDD refutation of \( \varphi \) is at least \( |\text{Cls}(\varphi)| \), that is \( n \geq |\text{Cls}(\varphi)| \). By Theorem 7 there is a resolution refutation of \( \varphi \) of size \( O(k) \), where

\[
k \leq n \cdot |\text{Cls}(\varphi)| \leq n^2
\]

\( \square \)

6 Conclusions

In this paper we proved that that resolution simulates OBDDs polynomially if we limit both to CNFs and thus answered the open question of Groote and Zantema posed in [8] whether there exists unsatisfiable CNFs having polynomial OBDD refutations and exponentially long resolution refutations. A logical next step would be to answer the question whether OBDD Apply method can be simulated by resolution linearly.

Acknowledgments

The author thanks Erika Ábrahám for helpful discussions at the early stage of this study.

References

1. S. B. Akers. Binary decision diagrams. *IEEE Transactions on Computers*, C-27(6):509–516, June 1978.
2. A. Atserias, P. Kolaitis, and M. Vardi. Constraint propagation as a proof system. In *Principles and Practice of Constraint Programming (CP 2004)*, volume 3258 of *LNCS*, pages 77–91, 2004.
3. M. Bonet, T. Pitassi, and R. Raz. On interpolation and automatization for Frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000.
4. R. T. Boute. The binary decision machine as a programmable controller. *EUROMICRO Newsletter*, 1(2):16–22, January 1976.
5. R. Bryant. Graph-based algorithms for boolean function manipulation. 8(C-35):677–691, 1986.
6. S. Cook and R. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.
7. M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. Communica-
tions of the ACM, 5(7):394–397, 1962.
8. J. F. Groote and H. Zantema. Resolution and binary decision diagrams cannot simulate each
other polynomially. Discrete Applied Mathematics, 130(2):157–171, 2003.
9. J. Krajíček. An exponential lower bound for a constraint propagation proof system based on
ordered binary decision diagrams. The Journal of Symbolic Logic, 73(1):227–237, 2008.
10. C. Y. Lee. Representation of switching circuits by binary-decision programs. Bell Systems
Technical Journal, 38:985–999, 1959.
11. N. Peltier. Extended resolution simulates binary decision diagrams. Discrete Applied Mathe-
ematics, 156(6):825–837, 2008.
12. K. Pipatsrisawat and A. Darwiche. On the power of clause-learning SAT solvers with restarts.
In Principles and Practice of Constraint Programming, 2009.
13. K. Pipatsrisawat and A. Darwiche. On modern clause-learning satisfiability solvers. J. Autom.
Reasoning, 44(3):277–301, 2010.
14. A. Razborov. Propositional proof complexity. J. ACM, 50(1):80–82, 2003.
15. J. A. Robinson. A machine-oriented logic based on the resolution principle. Journal of the ACM
(JACM), 12(1):23–41, 1965.
16. C. Sinz and A. Biere. Extended resolution proofs for conjoining BDDs. In Computer Science -
Theory and Applications, First International Computer Science Symposium in Russia, volume
3967 of LNCS. Springer, 2006.
17. O. Tveretina, C. Sinz, and H. Zantema. Ordered binary decision diagrams, pigeonhole formulas
and beyond. Journal on Satisfiability, Boolean Modeling and Computation, 7:35–58, 2010.
18. T. E. Uribe and M. E. Stickel. Ordered binary decision diagrams and the davis-putnam proce-
dure. In First International Conference on Constraints in Computational Logics, volume 845,
pages 34–49. Lecture Notes in Computer Science, 1994.
19. I. Wegener. Branching programs and binary decision diagrams: theory and applications. Society
for Industrial and Applied Mathematics, 2000.