Centrum voor Wiskunde en Informatica
**REPORT***RAPPORT*

# Resolution and binary decision diagrams cannot simulate each other polynomially

J.F. Groote, H. Zantema

Software Engineering (SEN)

**SEN-R0009 April 30, 2000**

Report SEN-R0009
ISSN 1386-369X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

# Resolution and Binary Decision Diagrams cannot Simulate each other Polynomially

J.F. Groote[1,2] JanFriso.Groote@cwi.nl

H. Zantema[1,3] hansz@cs.uu.nl

1: *CWI, P.O. Box 94.079, 1090 GB Amsterdam, The Netherlands*

2: *Department of Mathematics and Computing Science, Eindhoven University of Technology P.O. Box 513, 5600 MB Eindhoven, The Netherlands*

3: *Department of Computer Science, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, The Netherlands*

ABSTRACT

There are many different ways of proving formulas in proposition logic. Many of these can easily be characterized as forms of resolution (e.g. [12] and [9]). Others use so-called binary decision diagrams (BDDs) [2, 10]. Experimental evidence suggests that BDDs and resolution based techniques are fundamentally different, in the sense that their performance can differ very much on benchmarks [14]. In this paper we confirm these findings by mathematical proof. We provide examples that are easy for BDDs and exponentially hard for any form of resolution, and vice versa, examples that are easy for resolution and exponentially hard for BDDs.

*2000 Mathematics Subject Classification:* 03B05, 03F20
*1998 ACM Computing Classification System:* F.0
*Keywords and Phrases:* Resolution, Binary Decision Diagrams
*Note:* Work carried out under project SEN2

## 1. Introduction

We consider formulas in proposition logic: formulas consisting of proposition letters from some set $\mathcal{P}$, constants $\mathsf{t}$ (true) and $\mathsf{f}$ (false) and connectives $\vee$, $\wedge$, $\neg$, $\rightarrow$ and $\leftrightarrow$. There are different ways of proving the correctness of these formulas, i.e., proving that a given formula is a tautology. In the automatic reasoning community resolution is a popular proof technique, underlying the vast majority of all proof search techniques in this area, including for instance the well known branch-and-bound based technique named after Davis-Putnam-Loveland [5] or the remarkably effective methods by Stålmarck [12] and the GRASP prover [9].

In the VLSI and the process analysis communities binary decision diagrams (BDDs) are popular [2, 10]. BDDs have caused a considerable increase of the scale of systems that can be verified, far beyond anything a resolution based method has achieved. On the other hand there are many examples where resolution based techniques out-perform BDDs with a major factor, for instance in proving safety of railway interlockings ([7]). Out-performance in both directions has been described in [14].

However, benchmark studies only provide an impression, saying very little about the real relation of resolution and BDDs. The results may be influenced by badly chosen variable orderings in BDDs

or non optimal proof search strategies in resolution. Actually, given such benchmarks it can not be excluded that there exist a resolution based technique that always out-performs BDDs, provided a proper proof search strategy would be chosen. So, a mathematical comparison between the techniques is called for. This is not straightforward, as resolution and BDDs look very different. BDDs work on arbitrary formulas, whereas resolution is strictly linked to formulas in conjunctive normal form. And the resolution rule and the BDD construction algorithms appear of a totally dissimilar nature.

Moreover, classical (polynomial) complexity bounds cannot be used, as the problem we are dealing with is (co-)NP-complete. Fortunately, polynomial simulations provide an elegant way of dealing with this (see e.g. [16]). We say that proof system $A$ polynomially simulates proof system $B$ if for every formula $\phi$ the size of the proof of $\phi$ in system $A$ is smaller than a polynomial applied to the size of the proof of $\phi$ in system $B$. Of course, if the polynomial is more than linear, proofs in system $A$ may still be substantial longer than proofs in system $B$, but at least the proofs in $A$ are never exponentially longer. It is self evident that for practical applications it is important that the order of the polynomial is low. If it can be shown that for some formulas in $B$ the proofs are exponentially longer than those in $A$ we consider $A$ as a strictly better proof system than $B$. It has for instance been shown that 'extended resolution' is strictly better than resolution [8], being strictly better than Davis-Putnam resolution [6]; for an extended overview of comparisons of systems based on resolution, Frege systems and Gentzen systems we refer to [16].

We explicitly construct a sequence of biconditional formulas that are easy for BDDs, but exponentially hard for resolution. The proof that they are indeed hard for resolution is based on results from [15, 1]. The reverse is easier, namely showing that there is a class of formulas easy for any reasonable form of resolution, even only unit resolution, and exponentially hard for BDDs. For a suitable class of formulas including pigeon hole formulas we prove that the BDD approach is exponentially hard. It was proven before in [8] that for the same pigeon hole formulas resolution is exponentially hard for every strategy.

We start with preliminaries on OBDDs in Section 2. In Section 3 we prove that OBDD proofs are exponential for pigeon hole formulas and related formulas. In Section 4 we prove that OBDD proofs are polynomial for biconditional formulas. In Section 5 we present our results on resolution. In Section 6 we present the our main results in comparing resolution and OBDDs. Finally, in Section 7 we describe some points of further research.

**Acknowledgment.** Special thanks go to Oliver Kullmann and Alasdair Urquhart for their help with lower bounds for resolution.

## 2. Binary Decision Diagrams

The kind of Binary Decision Diagrams that we use presupposes a total ordering $<$ on $\mathcal{P}$, and therefore are also called Ordered Binary Decision Diagrams (OBDDs). First we present some basic definitions and properties as they are found in e.g. [2, 10]. An OBDD is a Directed Acyclic Graph (DAG) where each node is labeled by a proposition letter from $\mathcal{P}$, except for nodes that are labeled by 0 and 1. From every node labeled by a proposition letter, there are two outgoing edges, labeled 'left' and 'right', to nodes labeled by 0 or 1, or a proposition letter strictly higher in the ordering $>$. The nodes labeled by 0 and 1 do not have outgoing edges.

An OBDD compactly represents which valuations are valid, and which are not. Given a valuation $\sigma$ and an OBDD $B$, the $\sigma$ walk of $B$ is determined by starting at the root of the DAG, and iteratively following the left edge if $\sigma$ validates the label of the current node, and otherwise taking the right edge. If 0 is reached by a $\sigma$-walk then $B$ makes $\sigma$ invalid, and if 1 is reached then $B$ makes $\sigma$ valid. We say that an OBDD represents a formula if the formula and the OBDD validate exactly the same valuations.

An OBDD is called *reduced* if the following two requirements are satisfied.

1. For no node its left and right edge go to the same node. It is straightforward to see that a node with such a property can be removed. We call this the *eliminate* operation.

2. There are no two nodes with the same label of which the left edges go to the same node, and the right edges go to the same node. If this is the case these nodes can be taken together, which we call the *merge* operation.

Applying the merge and the eliminate operator to obtain a reduced OBDD can be done in linear time. Reduced OBDDs have the following very nice property.

LEMMA **1** *For a fixed order $<$ on $\mathcal{P}$, every propositional formula $\phi$ is uniquely represented by a reduced OBDD $B(\phi,<)$, and $\phi$ and $\psi$ are equivalent if and only if $B(\phi,<) = B(\psi,<)$.*

As a consequence, a propositional formula $\phi$ is a contradiction if and only if $B(\phi,<) = 0$, and it is a tautology if and only if $B(\phi,<) = 1$. Hence by computing $B(\phi,<)$ for any suitable order $<$ we can establish whether $\phi$ is a contradiction, or $\phi$ is a tautology, or $\phi$ is satisfiable. If the order $<$ is fixed we shortly write $B(\phi)$ instead of $B(\phi,<)$. We write $\#(B(\phi))$ for the number of internal nodes in $B(\phi)$.

The main ingredient for the computation of $B(\phi)$ is the *apply*-operation: given the reduced OBBDs $B(\phi)$ and $B(\psi)$ for formulas $\phi$ and $\psi$ and a binary connective $\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$ as parameters, the *apply*-operation computes $B(\phi \diamond \psi)$. For the usual implementation of *apply* as described in [2, 10] both time and space complexity are $O(\#(B(\phi)) * \#(B(\psi)))$. If $B(\phi)$ is known then $B(\neg\phi)$ is computed in linear time simply by replacing every 0 by 1 and vice versa; this computation is considered as a particular case of an *apply*-operation. Now for every $\phi$ its reduced OBDD $B(\phi)$ can be computed by recursively calling the *apply*-operation. As the basis of this recursion we need the reduced OBDDs for the single proposition letters. These are simple: the reduced OBDD for $p$ consists of a node labeled by $p$, having a left outgoing edge to 0 and a right outgoing edge to 1. By maintaining a hash-table for all sub-formulas it can be avoided that for multiple occurrences of sub-formulas the reduced OBDD is computed more than once.

By the *OBDD proof* of a formula $\phi$ we mean the recursive computation of $B(\phi)$ using the *apply*-operation as described above. If $\phi$ consists of $n$ boolean connectives then this proof consists of exactly $n$ calls of the *apply*-operation. However, by the expansion of sizes of the arguments of *apply* this computation can be of exponential complexity, even if it ends in $B(\phi) = 0$. As the satisfiability problem is NP-complete, this is expected to be unavoidable for every way to compute $B(\phi)$. We give an explicit construction of formulas for which we prove that the OBDD proofs are of exponential size, independently of the order $<$ on $\mathcal{P}$. In [3] it was proved that representing the middle bits of a binary multiplier requires an exponential OBDD; this function is easily represented by a small circuit, but not by a small formula, and hence does not serve for our goal of having a small formula with an exponential OBDD proof.

## 3. PIGEON HOLE FORMULAS

In this section we prove lower bounds for OBDD proofs for pigeon hole formulas and related formulas.

DEFINITION **2** *Let $m, n$ be positive integers and let $p_{ij}$ be distinct variables for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Let*

$$C_{m,n} = \bigwedge_{i=1}^{m} (\bigvee_{j=1}^{n} p_{ij}), \qquad R_{m,n} = \bigwedge_{j=1}^{n} (\bigvee_{i=1}^{m} p_{ij}), \qquad \overline{R}_{m,n} = \bigwedge_{j=1,\ldots,n, 1 \le i < k \le m} (\neg p_{ij} \vee \neg p_{kj}),$$

$$CR_{m,n} = C_{m,n} \wedge R_{m,n} \qquad PF_{m,n} = C_{m,n} \wedge \overline{R}_{m,n}.$$

In order to understand these formulas put the variables in a matrix according to the indexes. The formula $C_{m,n}$ states that in every of the $m$ columns at least one variable is true, the formula $R_{m,n}$ states that in every of the $n$ rows at least one variable is true, and the formula $\overline{R}_{m,n}$ states that in every of the $n$ rows at most one variable is true. Hence if $C_{m,n}$ holds then at least $m$ of the variables $p_{ij}$ are true and if $\overline{R}_{m,n}$ holds then at most $n$ of the variables $p_{ij}$ are true. Hence if $m > n$ then

$PF_{m,n}$ is a contradiction. Since this reasoning describes the well-known pigeon hole principle, the formulas $PF_{m,n}$ are called pigeon hole formulas. Note that $PF_{m,n}$ is in conjunctive normal form. In [8] it has been proved that for every resolution proof for $PF_{n+1,n}$ the length is at least exponential in $n$. Here we prove a similar exponential lower bound for OBDD proofs, which is of interest in itself since pigeon hole formulas are widely considered as benchmark formulas. For the main result of the paper however we get better results by using similar lower bounds for $CR_{m,n}$ instead since the size of $CR_{n,n}$ is quadratic in $n$ while pigeon hole formulas have cubic sizes. The contradictory formula in the main result is $p \wedge (\neg p \wedge CR_{n,n})$.

Our proof of these lower bounds has been inspired by the proof from [14] that every OBDD for $CR_{n,n}$ has a size that is exponential in $\sqrt{n}$, which we improve to a size that is exponential in $n$. First we need two lemmas.

LEMMA **3** *Let $\phi$ be a formula over variables in any finite set $\mathcal{P}$. Let $<$ be a total order on $\mathcal{P}$. Let $k < \#\mathcal{P}$. Write $\mathbb{B} = \{0,1\}$. Let $f_\phi : \mathbb{B}^{\#\mathcal{P}} \to \mathbb{B}$ the function representing $\phi$, in such a way that the smallest $k$ elements of $\mathcal{P}$ with respect to $<$ correspond to the first $k$ arguments of $f_\phi$. Let $A \subseteq \{1, \ldots, k\}$. Let $\vec{z} \in \mathbb{B}^k$. Assume that for every distinct $\vec{x}, \vec{x'} \in \mathbb{B}^k$ satisfying $x_i = x'_i = z_i$ for all $i \notin A$ there exists $\vec{y} \in \mathbb{B}^{\#\mathcal{P}-k}$ such that $f_\phi(\vec{x}, \vec{y}) \neq f_\phi(\vec{x'}, \vec{y})$. Then $\#B(\phi, <) \geq 2^{\#A}$.*

PROOF: There are $2^{\#A}$ different ways to choose $\vec{x} \in \mathbb{B}^k$ satisfying $x_i = z_i$ for all $i \notin A$. Now from the assumption it is clear that by fixing the first $k$ arguments of $f_\phi$, at least $2^{\#A}$ different functions in the remaining $\#\mathcal{P} - k$ arguments are obtained. All of these functions correspond to different nodes in the reduced OBDD $B(\phi, <)$, proving the lemma. □

LEMMA **4** *Let $m, n \geq 1$. Consider a matrix of $n$ rows and $m$ columns. Let the matrix entries be colored equally white and black, i.e., the difference between the number of white entries and the number of black entries is at most one. Then at least $\frac{(m-1)\sqrt{2}}{2}$ columns or at least $\frac{(n-1)\sqrt{2}}{2}$ rows contain both a black and a white entry.*

PROOF: If all rows contain both a black and a white entry we are done, so we may assume that at least one row consists of entries of the same color. By symmetry we may assume all entries of this row are white. If also a row exists with only black entries, then all columns contain both a black and a white entry and we are done. Since there is a full white row, we conclude that no full black column exists. Let $r$ be the number of full white rows and $c$ be the number of full white columns. The number of entries in these full white rows and columns together is $mr + cn - cr$, and the total number of white entries is at most $\frac{mn+1}{2}$, hence

$$\frac{mn+1}{2} \geq mr + cn - cr = mn - (m-c)(n-r).$$

Assume the lemma does not hold. Then $m - c < \frac{(m-1)\sqrt{2}}{2}$ and $n - r < \frac{(n-1)\sqrt{2}}{2}$, and

$$\frac{mn+1}{2} \geq mn - (m-c)(n-r) > mn - \frac{(m-1)\sqrt{2}}{2} * \frac{(n-1)\sqrt{2}}{2} = mn - \frac{(m-1)(n-1)}{2}$$

from which we conclude $m + n < 2$, contradiction. □

THEOREM **5** *For $m \geq n \geq 1$ and for every total order $<$ on $\mathcal{P} = \{p_{ij} | i = 1, \ldots, m, j = 1, \ldots, n\}$ both time and space complexity of the OBDD proofs of both $CR_{m,n}$ and $PF_{m,n}$ is $\Omega(1.63^n)$. Moreover, $\#B(CR_{m,n}, <) = \Omega(1.63^n)$.*

PROOF: The last step in the OBDD proof of $PF_{m,n}$ is the application of *apply* on $B(C_{m,n}, <)$ and $B(\overline{R}_{m,n}, <)$.

We prove that $\#B(CR_{m,n},<) \geq 2^{\frac{(n-1)\sqrt{2}}{2}}$ and that either $B(C_{m,n},<)$ has size at least $2^{\frac{(m-1)\sqrt{2}}{2}}$ or $B(\overline{R}_{m,n},<)$ has size at least $2^{\frac{(n-1)\sqrt{2}}{2}}$. Since $m \geq n$ and $2^{\frac{\sqrt{2}}{2}} > 1.63$, then the theorem immediately follows.

Let $\mathcal{P}_< \subset \mathcal{P}$ consist of the $\lfloor \frac{nm}{2} \rfloor$ smallest elements of $\mathcal{P}$ with respect to $<$, and let $\mathcal{P}_> = \mathcal{P} \setminus \mathcal{P}_<$. hence elements of $\mathcal{P}_>$ are greater than elements of $\mathcal{P}_<$. We say that row $j = \{p_{ij} | i = 1, \ldots, m\}$ is mixed if $i, i'$ exist such that $p_{ij} \in \mathcal{P}_<$ and $p_{i'j} \in \mathcal{P}_>$; we say that column $i = \{p_{ij} | j = 1, \ldots, n\}$ is mixed if $j, j'$ exist such that $p_{ij} \in \mathcal{P}_<$ and $p_{ij'} \in \mathcal{P}_>$.

From Lemma 4 we conclude that either at least $\frac{(n-1)\sqrt{2}}{2}$ rows are mixed or at least $\frac{(m-1)\sqrt{2}}{2}$ columns are mixed. For both cases we will apply Lemma 3 for $k = \lfloor \frac{nm}{2} \rfloor$. We number the elements of $\mathcal{P}$ from 1 to $mn$ such that the numbers $1, \ldots, k$ correspond to the elements of $\mathcal{P}_<$.

Assume that at least $\frac{(m-1)\sqrt{2}}{2}$ columns are mixed. In case all columns are mixed, separate one of them and consider it to be non-mixed. For every mixed column fix one element of $\mathcal{P}_<$ in that column; collect the numbers of these elements in the set $A$. For $i = 1, \ldots, k$ define $z_i = 1$ for $i$ corresponding to matrix elements in non-mixed columns and $z_i = 0$ for $i$ corresponding to matrix elements in mixed columns. Choose $\vec{x}, \vec{x'} \in \mathbb{B}^k$ satisfying $\vec{x} \neq \vec{x'}$ and $x_i = x'_i = z_i$ for all $i \notin A$. Then there exists $i \in A$ such that $x_i \neq x'_i$. Now let $\vec{y} = (y_{k+1}, \ldots, y_{mn})$ be the vector defined by $y_j = 0$ if $j \in \mathcal{P}_>$ corresponds to a matrix element in the same column as $i$, and $y_j = 1$ otherwise. Interpret the concatenation of $\vec{x}$ and $\vec{y}$ as an assignment to $\{0,1\}$ on the matrix entries. Non-mixed columns contain only the value 1, and every mixed column contains at least one value 1, except for one column which consists purely of zeros if and only if $x_i = 0$. Since we forced at least one column to be considered as non-mixed and containing only the value 1, every row contains at least one value 1. Hence $f_{CR_{m,n}}(\vec{x}, \vec{y}) = f_{C_{m,n}}(\vec{x}, \vec{y}) = x_i$, and similarly $f_{CR_{m,n}}(\vec{x'}, \vec{y}) = f_{C_{m,n}}(\vec{x'}, \vec{y}) = x'_i$. Since $x_i \neq x'_i$ we obtain $f_{C_{m,n}}(\vec{x}, \vec{y}) \neq f_{C_{m,n}}(\vec{x'}, \vec{y})$ and $f_{CR_{m,n}}(\vec{x}, \vec{y}) \neq f_{CR_{m,n}}(\vec{x'}, \vec{y})$. Now by Lemma 3 we conclude that $\#B(C_{m,n},<) \geq 2^{\#A} \geq 2^{\frac{(m-1)\sqrt{2}}{2}}$ and $\#B(CR_{m,n},<) \geq 2^{\#A} \geq 2^{\frac{(m-1)\sqrt{2}}{2}} \geq 2^{\frac{(n-1)\sqrt{2}}{2}}$.

For the remaining case assume that at least $\frac{(n-1)\sqrt{2}}{2}$ rows are mixed. The required bound for $\#B(CR_{m,n},<)$ follows exactly as above by symmetry. It remains to prove the bound for $\#B(\overline{R}_{m,n},<)$. For every mixed row fix one element of $\mathcal{P}_<$ in that row; collect all these elements in the set $A$. Define $z_i = 0$ for all $i = 1, \ldots, k$. Choose $\vec{x}, \vec{x'} \in \mathbb{B}^k$ satisfying $\vec{x} \neq \vec{x'}$ and $x_i = x'_i = z_i = 0$ for all $i \notin A$. Then there exists $i \in A$ such that $x_i \neq x'_i$. Now define $\vec{y} = (y_{k+1}, \ldots, y_{mn})$ by choosing $y_j = 0$ for all but one $j$, and $y_j = 1$ for one single $j$ for which $i$ and $j$ correspond to matrix elements in the same row. This is possible because $i$ corresponds to an entry in a mixed row. Since in every other row at most one value is set to 1 all corresponding clauses in $\overline{R}_{m,n}$ are true. The only clause in $\overline{R}_{m,n}$ that is possibly false is the one corresponding to $i$ and $j$. We obtain $f_{\overline{R}_{m,n}}(\vec{x}, \vec{y}) = \neg x_i$ and $f_{\overline{R}_{m,n}}(\vec{x'}, \vec{y}) = \neg x'_i$. Since $x_i \neq x'_i$ we have $f_{\overline{R}_{m,n}}(\vec{x}, \vec{y}) \neq f_{\overline{R}_{m,n}}(\vec{x'}, \vec{y})$. Now by Lemma 3 we conclude that $\#B(\overline{R}_{m,n},<) \geq 2^{\#A} \geq 2^{\frac{(n-1)\sqrt{2}}{2}}$. □

Note that we proved that either $C_{m,n}$ or $\overline{R}_{m,n}$ must have an OBDD of exponential size. However, for each of these formulas seperately a properly chosen order may lead to small OBDDs. Indeed, if

$$p_{ij} < p_{i'j'} \iff (i < i') \vee (i = i' \wedge j < j')$$

then $\#B(C_{m,n},<) = mn$ and if

$$p_{ij} < p_{i'j'} \iff (j < j') \vee (j = j' \wedge i < i')$$

then $\#B(R_{m,n},<) = mn$ and $\#B(\overline{R}_{m,n},<) = 2(m-1)n$, all being linear in the number of variables.

## 4. Biconditional formulas

An interesting class of formulas are *biconditional formulas* consisting of proposition letters, biconditionals ($\leftrightarrow$) and negations ($\neg$). Biconditionals have very nice properties: they are associative,

$\phi \leftrightarrow (\psi \leftrightarrow \chi) \equiv (\phi \leftrightarrow \psi) \leftrightarrow \chi$, commutative, $\phi \leftrightarrow \psi \equiv \psi \leftrightarrow \phi$, idempotent, $\phi \leftrightarrow \phi \equiv \mathbf{t}$ and satisfy $\phi \leftrightarrow \neg\psi \equiv \neg(\phi \leftrightarrow \psi)$.

For a string $S = p_1, p_2, p_3, \ldots, p_n$ of proposition letters, where letters are allowed to occur more than once, we write

$$[S] \;=\; p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \cdots (p_{n-1} \leftrightarrow p_n)) \cdots).$$

It is not difficult to see that $[S]$ is a tautology if and only if all letters occur an even number of times in $S$.

A formula of the shape $[S]$ or $\neg[S]$ for a string $S$ in which every symbol occurs at most once, is called a *biconditional normal form*. Using the above properties it is easy to show that for every biconditional formula there exists a logically equivalent biconditional normal form.

The BDD technique turns out to be very effective for biconditional formulas. We show that for any biconditional formula its OBDD proof has a polynomial complexity. For any biconditional formula $\phi$, we write $|\phi|$ for the size of $\phi$, $\alpha(\phi)$ for the number of variables occurring in $\phi$ and $\alpha_{odd}(\phi)$ for the number of variables that occur an odd number of times in $\phi$.

It is useful to speak about the OBDD of $n$ formulas, $\phi_1, \ldots, \phi_n$. This OBDD is a single DAG with up to $n$ root nodes. The notion *reduced* carries over to these OBDDs. In particular, if $\phi_i$ and $\phi_j$ are equivalent, then the $i^{\text{th}}$ and $j^{\text{th}}$ root node are the same. Again the size of a DAG is defined to be the number of its internal nodes.

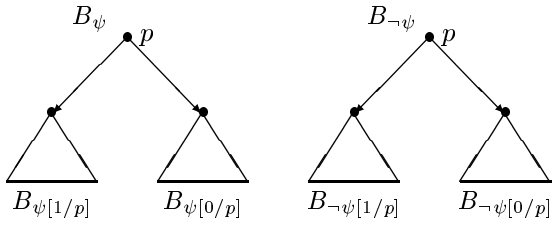
We have the following lemma, showing that each reduced OBDD for a biconditional formula is small.

LEMMA **6** *Let $\phi$ be a biconditional formula. Any reduced OBDD for $\phi$ and $\neg\phi$ has size $2\alpha_{odd}(\phi)$.*

PROOF: First fix an arbitrary ordering $<$ on the proposition letters. Note that there is a biconditional normal form $\psi$ that is equivalent to $\phi$. As by Lemma 1 the reduced OBDD of $\phi$ and $\psi$ are the same, we can as well construct the OBDD of $\psi$. Moreover, $\alpha_{odd}(\phi) = \alpha_{odd}(\psi)$.

We prove the lemma by induction on $\alpha_{odd}(\psi)$.

- $\alpha_{odd}(\psi) = 0$. As $\psi$ is a biconditional normal form, it does not contain any proposition letter, and hence is either equivalent to true or false. So, the reduced OBDD of $\phi$ and $\neg\phi$ does not contain internal nodes at all, and has size 0.
- $\alpha(\psi)_{odd} = n + 1$. Consider the first letter in the ordering $<$ that occurs in $\psi$ and let it be $p$. The OBDDs for $\psi$ and $\neg\psi$ look like:




Here $\psi[v/p]$ is the formula $\psi$ where $v$ has been substituted for $p$. Clearly, as $p$ occurs an odd time in $\psi$, $\psi[0/p] \equiv \neg\psi[1/p]$ and $\psi[1/p] \equiv \neg\psi[0/p]$. So, the reduced OBDD of $\psi[0/p]$, $\neg\psi[1/p]$, $\psi[1/p]$ and $\neg\psi[0/p]$ is the same as the OBDD of $\psi[0/p]$ and $\neg\psi[0/p]$. Using the induction hypothesis, the size of this OBDD must be $2n$. The reduced OBDD for $\psi$ and $\neg\psi$ adds two new nodes. So, the size of the reduced OBDD of $\psi$ and $\neg\psi$ is $2n + 2$. This equals $2\alpha_{odd}(\psi) + 2$, finishing the proof. □

THEOREM **7** *Let $<$ be an ordering on the proposition letters.*

- *The complexity of the corresponding OBDD proof for any biconditional formula $\phi$ is $O(|\phi|^3)$.*
- *The complexity of the corresponding OBDD proof for $[S]$ or $\neg[S]$ for any string $S$ of proposition letters is $O(|[S]|^2)$.*

PROOF: The OBDD proof for $\phi$ consists of $O(|\phi|)$ applications of *apply* applied on reduced OBDDs of sub-formulas of $\phi$. By Lemma 6 each of these reduced OBDDs has size $O(|\phi|)$. Since the complexity of $apply(\leftrightarrow, B, B')$ is $O(\#B * \#B')$ and the complexity of $apply(\neg, B)$ is $O(\#B)$ for every *apply* operation the complexity is $O(|\phi|^2)$, yielding $O(|\phi|^3)$ for the full OBDD proof for $\phi$.

For the OBDD proof for $[S]$ or $\neg[S]$ only applications of $apply(\leftrightarrow, B, B')$ occur with $\#B = 1$, giving the complexity $O(\#B')$, yielding $O(|[S]|^2)$ for the full OBDD proof. □

## 5. RESOLUTION

Resolution is a very common technique to prove formulas. Contrary to the BDD technique, it is applied to formulas in *conjunctive normal form* (*CNF*), i.e. formulas of the form

$$\bigwedge_{i \in I} \bigvee_{j \in J_i} l_{ij}$$

where $I$ and $J_i$ are finite index sets and $l_{ij}$ is a literal, i.e. a formula of the form $p$ or $\neg p$ for a proposition letter $p$. Each sub-formula $\bigvee_{j \in J_i} l_{ij}$ is called a *clause*. As $\wedge$ and $\vee$ are associative, commutative and idempotent it is allowed and convenient to view clauses as sets of literals and CNFs as sets of clauses.

The resolution rule can be formulated by:

$$\frac{\{p, l_1, \ldots, l_n\} \quad \{\neg p, l'_1, \ldots, l'_{n'}\}}{\{l_1, \ldots, l_n, l'_1, \ldots, l'_{n'}\}}$$

where $p$ is a proposition letter and $l_i$, $l'_j$ are literals. A resolution proof of a set of clauses $F$ is a sequence of clauses where the last clause is empty and each clause in the sequence is either taken from $F$, or matches the conclusion of the resolution rule, where both premises occur earlier in the sequence. Such a resolution sequence ending in the empty clause is called a *resolution refutation*, and proves that the conjunction of the set of clauses is a contradiction.

In case one of the clauses involved is a single literal $l$, by this resolution rule all occurrences of the negation of $l$ in all other clauses may be removed. Moreover, all other clauses containing $l$ then may be ignored. Eliminating all occurrences of $l$ and its negation in this way is called *unit resolution*. All practical resolution proof search systems start with doing unit resolution as long as possible.

In order to apply resolution on arbitrary formulas, these formulas must first be translated to CNF. This can be done in linear time maintaining satisfiability using the Tseitin transformation [13]. A disadvantage of this transformation is the introduction of new variables, but it is well-known that a transformation to CNF without the introduction of new variables is necessarily exponential. For instance, it is not difficult to prove that for

$$(\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots\cdots \leftrightarrow p_n)$$

every clause in a CNF contains either $p_i$ or $\neg p_i$ for every $i$. Since one such clause of $n$ literals causes only one zero in the truth table of the formula, the full CNF contains $2^{n-1}$ of these clauses to obtain all $2^{n-1}$ zeros in its truth table. Hence without the introduction of new variables every CNF of this formula is of exponential size. More general for every biconditional formula $\phi$ without the introduction of new variables every CNF consists of at least $2^{\alpha_{odd}(\phi)-1}$ clauses each consisting of at least $\alpha_{odd}(\phi)$ literals.

The *Tseitin transformation* works as follows. Given a formula $\phi$. Every sub-formula $\psi$ of $\phi$ not being a proposition letter is assigned a new letter $p_\psi$. Now the Tseitin transformation of $\phi$ consists of

- the single literal $p_\phi$;
- the conjunctive normal form of $p_\psi \leftrightarrow (p_{\psi_1} \diamond p_{\psi_2})$ for every subterm $\psi$ of $\phi$ of the shape $\psi = \psi_1 \diamond \psi_2$ for a binary operator $\diamond$;
- the conjunctive normal form of $p_\psi \leftrightarrow \neg p_{\psi_1}$ for every subterm $\psi$ of $\phi$ of the shape $\psi = \neg\psi_1$.

Here $p_{\psi_i}$ is identified with $\psi_i$ in case $\psi_i$ is a proposition letter, for $i = 1, 2$. It is easy to see that this set of clauses is satisfiable if and only if $\phi$ is satisfiable. Moreover, every clause consists of at most three literals, and the number of clauses is linear in the size of the original formula $\phi$.

It is not difficult to see that after applying the Tseitin transformation to a CNF, by a number of resolution steps linear in the size of the CNF, the original CNF can be re-obtained. By a resolution proof for an arbitrary formula we mean a resolution proof after the Tseitin transformation has been applied.

We now give a construction of strings $S_n$ in which all letters occur exactly twice by which $\neg[S_n]$ is a contradiction, and for which we prove that every resolution proof of $\neg[S_n]$ is very long. Although the construction is somewhat involved, we think that simpler constructions do not suffice. In [16] for instance it was proved that $\neg[p_1, p_2, \ldots, p_n, p_1, p_2, \ldots, p_n]$ admits a resolution proof that is quadratic in $n$.

For a string $S$ and a label $i$ we write $\mathsf{lab}(S, i)$ for the string obtained from $S$ by replacing every symbol $p$ by a fresh symbol $p_i$. For a string $S$ of length $n * 2^n$ we write $\mathsf{ins}(n, S)$ for the string obtained from $S$ by inserting the symbol $i$ after the $(i * n)$-th symbol for $i = 1, 2, \ldots, n$. We define

$$S_1 = 1, 1, \quad \text{and}$$

$$S_{n+1} = \mathsf{ins}(n, \mathsf{lab}(S_n, 0)), \mathsf{ins}(n, \mathsf{lab}(S_n, 1)),$$

for $n > 0$. For instance, we have

$$S_1 = \underbrace{1}, \underbrace{1},$$

$$S_2 = \underbrace{1_0, 1}, \underbrace{1_0, 2}, \underbrace{1_1, 1}, \underbrace{1_1, 2},$$

$$S_3 = \underbrace{1_{00}, 1_0, 1}, \underbrace{1_{00}, 2_0, 2}, \underbrace{1_{10}, 1_0, 3}, \underbrace{1_{10}, 2_0, 4}, \underbrace{1_{01}, 1_1, 1}, \underbrace{1_{01}, 2_1, 2}, \underbrace{1_{11}, 1_1, 3}, \underbrace{1_{11}, 2_1, 4}.$$

Clearly $S_n$ is a string of length $n * 2^n$ over $n * 2^{n-1}$ symbols each occurring exactly twice. The string $S_n$ can be considered to consist of $2^n$ consecutive groups of $n$ symbols, called $n$-groups. In the examples $S_1$, $S_2$ and $S_3$ above the $n$-groups are under-braced. Write $g_{n,k}$ to be the $k$-th $n$-group in $S_n$, for $n > 1$ and $1 \leq k \leq 2^n$.

LEMMA **8** *Let $A \subseteq \{1, 2, \ldots, 2^n\}$ for any $n > 0$. Then there are at least $\min(\#A, 2^n - \#A)$ pairs $(k, k')$ such that $k, k' \in \{1, 2, \ldots, 2^n\}$, $k \in A$, $k' \notin A$ and $g_{n,k}$ and $g_{n,k'}$ have a common symbol.*

PROOF: We apply induction on $n$; for $n = 1$ the lemma clearly holds. Let $m_0 = \#\{k \in A | k \leq 2^{n-1}\}$ and $m_1 = \#\{k \in A | k > 2^{n-1}\}$. Say that $(k, k')$ is a matching pair if $k \in A$, $k' \notin A$ and $g_{n,k}$ and $g_{n,k'}$ have a common symbol. If $k, k' \leq 2^{n-1}$ then by construction $g_{n,k}$ and $g_{n,k'}$ have a common symbol if $g_{n-1,k}$ and $g_{n-1,k'}$ have a common symbol. If $k, k' > 2^{n-1}$ then by construction $g_{n,k}$ and $g_{n,k'}$ have a common symbol if $g_{n-1,k-2^{n-1}}$ and $g_{n-1,k'-2^{n-1}}$ have a common symbol. Hence by induction hypothesis there are at least $\min(m_0, 2^{n-1} - m_0)$ matching pairs $(k, k')$ with $k, k' \leq 2^{n-1}$ and at least $\min(m_1, 2^{n-1} - m_1)$ matching pairs $(k, k')$ with $k, k' > 2^{n-1}$. Since by construction $g_{n,k}$ and $g_{n,k+2^{n-1}}$

have a common symbol for every $k = 1, 2, \ldots, 2^{n-1}$, there are at least $|m_0 - m_1|$ matching pairs $(k, k')$ with $|k - k'| = 2^{n-1}$. Hence the total number of matching pairs is at least

$$|m_0 - m_1| + \min(m_0, 2^{n-1} - m_0) + \min(m_1, 2^{n-1} - m_1).$$

A simple case analysis shows that this is at least $\min(m_0 + m_1, 2^n - m_0 - m_1) = \min(\#A, 2^n - \#A)$. □

Essentially this lemma states the well-known fact that for any set $A$ of vertices of an $n$-dimensional cube there are at least $\min(\#A, 2^n - \#A)$ edges for which one end is in $A$ and the other is not. It is applied in the next lemma stating a lower bound on connections between separate elements of $S_n$ rather than connections between $n$-groups.

LEMMA **9** *Let $n > 0$ and let $B \subseteq \{1, 2, \ldots, n * 2^n\}$. Let $X \subseteq \{1, 2, \ldots, n * 2^n\}^2$ consist of the pairs $(i, j)$ for which $i \in B$ and $j \notin B$ and for which either $|i - j| = 1$ or the $i$-th element of $S_n$ is equal to the $j$-th element of $S_n$. Then*

$$\#X \geq \frac{\min(\#B, n * 2^n - \#B)}{2n}.$$

PROOF: Assume that $\#B \leq n * 2^{n-1}$, otherwise replace $B$ by its complement. Let $A$ be the set of numbers $k \in \{1, \ldots, 2^n\}$ for which all elements of the corresponding $n$-group $g_{n,k}$ correspond to elements of $B$, i.e., $\{(k-1) * n + 1, \ldots, k * n\} \subseteq B$. Let $m_1 = \#A$. Let $m_2$ be the number of $n$-groups for which none of the elements correspond to elements of $B$, i.e., $m_2 = \#\{k \in \{1, \ldots, 2^n\} | \{(k-1) * n + 1, \ldots, k * n\} \cap B = \emptyset\}$. Let $m_3$ be the number of remaining $n$-groups, i.e., $n$-groups containing elements corresponding to both elements of $B$ and outside $B$. Clearly $n * m_1 \leq \#B \leq n * (m_1 + m_3)$. Each of the $m_3$ remaining groups gives rise to a pair $(i, j) \in X$ for which $|i - j| = 1$. Hence $\#X \geq m_3$.

Now assume that $m_1 > m_3$. Since $n * m_1 \leq \#B \leq n * 2^{n-1}$ we have $m_1 = \#A \leq 2^{n-1}$. By Lemma 8 we obtain at least $m_1$ pairs $(k, k')$ such that $k \in A$, $k' \notin A$ and $g_{n,k}$ and $g_{n,k'}$ have a common symbol. For at least $m_1 - m_3$ of the corresponding $n$-groups $g_{n,k'}$ none of the elements correspond to elements of $B$. Since $g_{n,k}$ and $g_{n,k'}$ have a common symbol for every corresponding pair $(k, k')$ this gives rise to at least $m_1 - m_3$ pairs $(i, j) \in X$ for which the $i$-th element of $S_n$ is equal to the $j$-th element of $S_n$. Hence in case $m_1 > m_3$ we conclude $\#X \geq m_3 + (m_1 - m_3) = m_1$.

We conclude

$$\#X \geq \max(m_3, m_1) \geq \frac{m_1 + m_3}{2} \geq \frac{\#B}{2n}.$$

□

THEOREM **10** *Every resolution proof of $\neg[S_n]$ contains $2^{\Omega(2^n/n)}$ resolution steps.*

PROOF: Let $S_n = p_1, p_2, \ldots, p_{n2^n}$; note that for every $i$ there exists exactly one $j$ with $p_i = p_j$ and $i \neq j$. Introduce distinct help symbols $q_0, q_1, q_2, \ldots, q_{n2^n-1}$. Now the Tseitin transformation of $\neg[S_n]$ consists of

- the single literal $q_0$;
- the conjunctive normal form of $q_0 \leftrightarrow \neg q_1$;
- the conjunctive normal form of $q_i \leftrightarrow (p_i \leftrightarrow q_{i+1})$ for every $i = 1, 2, \ldots, n * 2^n - 2$;
- the conjunctive normal form of $q_i \leftrightarrow (p_i \leftrightarrow p_{i+1})$ for $i = n * 2^n - 1$.

This set of clauses is exactly the same as $\tau(G, f)$, where $\tau$ is Tseitin's graph construction [13] also described in [15, 16, 1] for the graph $G = (V, E)$ where $V = \{-1, 0, 1, 2, \ldots, n * 2^n - 1\}$ and $E$ consists of the edges

- $(i, i+1)$ for $i = -1, 0, 1, 2, \ldots, n * 2^n - 2$,
- $(i, j)$ for $n2^n > j > i > 0$ and $p_i = p_j$,
- $(i, n * 2^n - 1)$ for $i$ with $p_i = p_{n2^n}$,

and the charge function $f : V \to \{0, 1\}$ is defined by $f(-1) = 0$, $f(0) = 1$ and $f(i) = 0$ for $i > 0$. The observation that these sets of clauses coincide essentially goes back to [11].

The expansion $e(G)$ of an undirected graph $G = (V, E)$ is defined to be the smallest number

$$\#\{(v, v') \in E | (v \in B \wedge v' \notin B) \vee (v \notin B \wedge v' \in B)\}$$

for some $B \subseteq V$ satisfying $\frac{1}{3}\#V \leq \#B \leq \frac{2}{3}\#V$. For our graph $G = (V, E)$ the edges $(v, v')$ satisfying $(v \in B \wedge v' \notin B) \vee (v \notin B \wedge v' \in B)$ correspond to pairs $(i, j)$ as occurring in Lemma 9 up to a constant part of $V$. Hence by Lemma 9 we obtain $e(G) = \Omega(\#V/2n) = \Omega(2^n)$. In [1] the following two results were proved:

- Every resolution proof of $\tau(G, f)$ involves clauses with at least $e(G)$ literals.
- If a contradictory CNF on $m$ variables of bounded clause size admits a resolution proof of length $s$, then it also admits a resolution proof only involving clauses of size $O(\sqrt{m \log s})$.

Hence, $\sqrt{n * 2^n * \log s} \geq c * 2^n$ for some $c > 0$, from which we conclude $s = 2^{\Omega(2^n/n)}$.

□

By using *expander graphs* it would be possible to prove the existence of contradictory biconditional formulas of size $\Theta(n)$ such that every resolution proof contains $2^{\Omega(i)}$ resolution steps. However, expressed in the size of the formula this improvement is only logarithmic compared to Theorem 10, while the construction of the formula is much more complicated.

## 6. The main result

We now have collected sufficient observations to come to our main result saying that the binary decision diagram technique is polynomially incomparable with any reasonable proof search technique based on resolution.

THEOREM **11**

- *There is a sequence of contradictory formulas $\phi_i$ of size $\Theta(i \log^2 i)$ $(i \geq 0)$ for which every OBDD proof has time and space complexity $O(i^2 \log^4 i)$, and for which each resolution proof requires $2^{\Omega(i)}$ resolution steps.*
- *There is a sequence of contradictory formulas $\psi_i$ in CNF of size $\Theta(i^2)$ $(i \geq 0)$ that is proven in $O(i^2)$ steps using only unit resolution, and for which every OBDD proof has time and space complexity $\Omega(1.63^i)$.*

PROOF:

- Take the formulas $\phi_i$ to be $\neg[S_n]$ from Theorem 10, where $n$ is the smallest number satisfying $i \leq \frac{2^n}{n}$. Then the size of $\phi_i$ is $\Theta(n * 2^n) = \Theta(i \log^2 i)$, while by Theorem 10 every resolution proof requires $2^{\Omega(2^n/n)} = 2^{\Omega(i)}$ resolution steps. By Theorem 7 every OBDD proof has time and space complexity $O((n * 2^n)^2) = O(i^2 \log^4 i)$[1].
- Let $\psi_i$ be $p \wedge (\neg p \wedge CR_{i,i})$. These formulas have size $\Theta(i^2)$. An OBDD proof of $\psi_i$ contains an OBDD proof of $CR_{i,i}$ as one of its recursive calls; this takes time and space complexity $\Omega(1.63^i)$ by Theorem 5. It is easy to check that after applying the Tseitin transformation on $\psi_i$ only unit resolution leads to a refutation in a number of steps linear in the size of $\psi_i$.

□

[1] By a careful analysis using the specific structure of the formula $\neg[S_n]$ this can be improved to $O(i^2 \log^3 i)$.

## 7. Further research

In this paper we have shown that any technique based on a reasonable form of resolution is essentially different from the standard OBDD technique to prove formulas. However, many questions remain, such as:

1. Is there a natural strengthening of the resolution rule that allows to simulate the construction of OBDDs polynomially by resolution? A good candidate is *extended resolution* (see e.g. [4]) where it is allowed to introduce new proposition letters defined in terms of existing ones. In [4, 8] it has been shown that extended resolution has a much stronger proving power than resolution.

2. On the other hand, there are modifications of the OBDD-technique by which for every formula $\phi$ the contrived example $p \wedge (\neg p \wedge \phi)$ can be handled efficiently, for instance the lazy strategy as described in [17]. How do these modifications of the OBDD-technique relate to resolution?

3. We have shown that biconditional formulas have short OBDD proofs, and after the Tseitin transformation they may require long resolution proofs. One can wonder whether contradictory conjunctive normal forms exist having polynomial OBDD proofs and requiring exponentially long resolution proofs. The Tseitin transformation of our biconditional formulas will not serve for this goal: OBDD proofs of these transformed biconditional formulas appear to be of exponential length.

## References


1. Ben-Sasson, E., and Wigderson, A. Short proofs are narrow – resolution made simple. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing* (1999), pp. 517–526.
2. Bryant, R. E. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers C-35*, 8 (1986), 677–691.
3. Bryant, R. E. On the complexity of VLSI implmentations and graph representations of boolean functions with application to integer multiplication. *IEEE Transactions on Computers 40*, 2 (1991), 205–213.
4. Cook, S. The complexity of theorem proving procedures. *Proceedings of the 3rd annual ACM symposium on the Theory of Computing* (1971), 151–158.
5. Davis, M., Logemann, G., and Loveland, D. A machine program for theorem proving. *Communications of the ACM 5* (1962), 394–397.
6. Goerdt, A. Davis-Putnam resolution versus unrestricted resolution. *Annals of Mathematics and Artificial Intelligence 6* (1992), 169–184.
7. Groote, J. F., van Vlijmen, S. F. M., and Koorn, J. W. C. The safety guaranteeing system at station Hoorn-Kersenboogerd. In *COMPASS-95, proceedings 10th annual Conference on Computer Assurance* (1995), IEEE, pp. 57–68.
8. Haken, A. The intractability of resolution. *Theoretical Computer Science 39* (1985), 297–308.
9. Marques Silva, J. P., and Sakallah, K. M. Grasp – a new search algorithm for satisfiability. Tech. Rep. CSE-TR-292-96, University of Michigan, Department of Electrical Engineering and Computer Science, 1996.
10. Meinel, C., and Theobald, T. *Algorithms and Data Structures in VLSI Design: OBDD — Foundations and Applications.* Springer, 1998.
11. Reckhow, R. *On the lengths of proofs in the propositional calculus.* PhD thesis, University of Toronto, 1975.
12. Stålmarck, G., and Säflund, M. Modeling and verifying systems and software in propositional logic. In *Safety of Computer Control Systems (SAFECOMP '90)* (1990), B. Daniels, Ed., vol. 656, Pergamon Press, pp. 31–36.
13. Tseitin, G. On the complexity of derivation in propositional calculus. In *Studies in Constructive Mathematics and Mathematical Logic, part 2* (1968), pp. 115–125. Reprinted in J. Siekmann and G. Wrightson (editors),*Automation of reasoning* vol. 2, pp. 466-483.,Springer–Verlag Berlin, 1983.
14. Uribe, T. E., and Stickel, M. E. Ordered binary decision diagrams and the Davis-Putnam procedure. In *First conference on Constraints in Computational Logic* (1994), J.-P. Jouannaud, Ed., vol. 845 of *Lecture Notes in Computer Science*, Springer, pp. 34–49.
15. Urquhart, A. Hard examples for resolution. *Journal of the ACM 34*, 1 (1987), 209–219.
16. Urquhart, A. The complexity of propositional proofs. *The Bulletin of Symbolic Logic 1*, 4 (1995), 425–467.
17. van de Pol, J. C., and Zantema, H. Binary decision diagrams by shared rewriting. Tech. Rep. UU-CS-2000-06, Utrecht University, Department of Computer Science, 2000. Available via `http://www.cs.uu.nl/docs/research/publication/TechRep.html`.