Journal on Satisfiability, Boolean Modeling, and Computation 10 (2016) 1–9

On Exponential Lower Bounds for Partially Ordered Resolution

Mikoláš Janota mikolas.janota@gmail.com
Microsoft Research
Cambridge
United Kingdom

Abstract

Is well-known that the proof size in propositional resolution is sensitive to the order of how variables are resolved on. Indeed, imposing a certain resolution order can lead to an exponential blowup compared to unrestricted resolution. In this paper we study even a weaker restriction. We require that one partition of variables is resolved on before the second partition (this is a special case of a partial order). We show that this also can lead to an exponential blowup. This helps us to understand why dynamic variable orderings in SAT solvers is so successful but also it motivates further investigation in variable orderings in SAT solvers.

KEYWORDS: *resolution, lower bounds, resolution order, partial order*

Submitted 1 Jan 2015; revised 30 Mar 2016; published 2 May 2016

1. Introduction

The study of *proof complexity* has both theoretical and practical roots. It is well-known that proof size is tightly linked to *computational complexity* [5]. Sizes of proofs are interesting from a practical perspective because the runtime of decision procedures can be modeled by the size of the corresponding proof. In particular, modern SAT solvers rely heavily on *propositional resolution* [16] and thus this system is particularly of interest.

A number of variants of resolution were studied throughout the years. Most notably *regular resolution* where a variable might be resolved on at most once on any resolution path [19]. Regular resolution was separated from unrestricted resolution [6, 7, 1]. Another special case of resolution is *ordered resolution*, or *DPLL resolution*, where variables must be resolved on in a certain order. This is in fact a special case of regular resolution but a super-polynomial lower bound was already shown by Goerdt in '92 [7]. Another well-known type of resolution is *tree resolution*, where the resolution graph is represented as a tree (essentially it means that derived clauses cannot be reused). Tree and ordered resolution were exponentially separated from unrestricted resolution by Bonet et al. [4]. Other known resolution types include *negative resolution* [8] and *SLD-resolution* [14]. Exponential lower bounds for unrestricted resolution were demonstrated by Haken [9].

Since SAT solvers rely heavily on *unit propagation* and *clause learning*, their strengths have been extensively studied [2, 18]. Another concept related to resolution order is the order of *decisions* made by a CDCL SAT solver. Indeed, a backtracking SAT solver, with no

unit-propagation or clause learning leads to a resolution proof where the resolution order is the same as the order of decisions in the algorithm. In CDCL solvers this is complicated by the fact that unit propagation implicitly influences the order of how variables are resolved on. The effects of decision order have been also extensively studied [13, 15, 17, 12].

In this paper we show two classes of formulas where considering a specific partial order leads to an exponential blowup compared to unrestricted resolution. Section 2 introduces notation and concepts used in the remainder of the paper; in particular *partially ordered resolution* is defined. Section 3 introduces a simple one-player game where specific partial variable order leads to exponential blowup. Section 4 studies a class of formulas that correspond to an encoding of a simple tautology $F \vee \neg F$; here resolving on the auxiliary variables first leads to exponential blowup. Finally, Section 5 concludes the paper.

2. Preliminaries

A *literal* is a Boolean variable or its negation. A *term* is a conjunction of literals and a *clause* is a disjunction of literals. A formula in *conjunctive normal form* (CNF) is a conjunction of clauses. A CNF formula may be treated as a set of clauses and a clause as a set of literals.

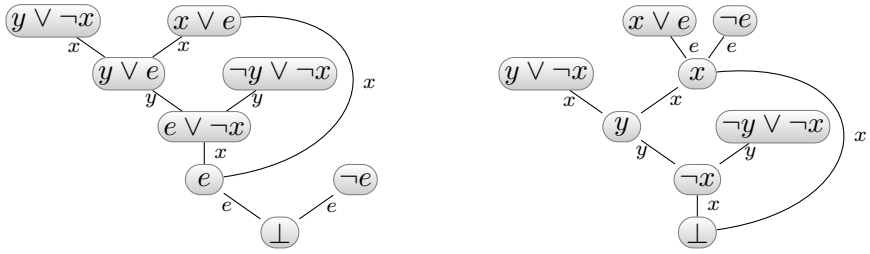
An *assignment* is a mapping from variables to the Boolean constants $\{0, 1\}$. We write $F \restriction \tau$ for a *restriction* of F by the assignment τ. An assignment is *complete* for a set of variables X when it maps all variables from X to a constant. We slightly abuse the notation so that for a Boolean function $f : X \to \{0, 1\}$ a complete assignment to X can be given as an argument.

For Boolean function $f : X \to \{0, 1\}$, an *implicant* is a term t that contains only variables X and for any complete assignment τ to X, if $t \restriction \tau$ is true then $f(\tau) = 1$. An *implicate* is defined analogously as a clause C so that $f(\tau) = 1$ implies $C \restriction \tau$ is true. Note that for any term t, $\neg t$ is a clause and vice versa. A term t is an implicant of f iff $\neg t$ is an implicate of $\neg f$.

The resolution rule is defined as follows. For two clauses $C_1 \vee x$ and $C_2 \vee \neg x$, called the *antecedents*, derive the clause $C_1 \vee C_2$, which is called the *resolvent*. The variable x is called the *pivot*, and we say that it is being *resolved on*. A *resolution proof* of a clause C from a set of clauses ϕ is a finite sequence of clauses $C_1, \ldots, C_n$ with $C_n = C$ and such that each clause either belongs to ϕ or is derived by the resolution rule from some of the previous ones. A resolution proof is called a *refutation* if it derives the empty clause, denoted as $\bot$. Resolution is sound and refutation complete, i.e. a formula is unsatisfiable if and only if it can be refuted.

A resolution proof π of a clause C corresponds to a (rooted) directed acyclic graph (DAG) constructed as follows. The clause C is the root; any clause that is the result of resolution has out-degree 2 connecting it to the antecedents; clauses coming from π have out-degree 0. The clauses going from π are commonly referred to as *axioms* and they form *leafs* of the proof DAG. The *size* of a resolution proof is the number of clauses appearing in it.

Definition 1. Let $\prec$ be a partial order on variables and let π be a resolution proof. We say that π is $\prec$*-ordered* if for any path P from a leaf to the root in π, if x is resolved on before y on P, then it does *not* hold that $y \prec x$.



(a) $\{x, y\}<\{e\}$-ordered refutation

(b) $\{e\}<\{x, y\}$-ordered refutation

Figure 1. Example refutations.

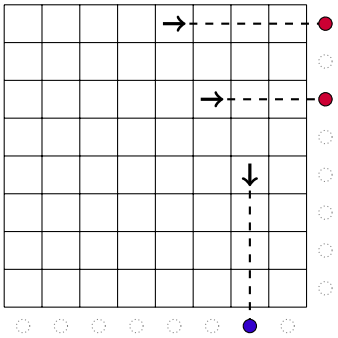


Figure 2. SE-game illustration with 3 arrows placed so far.

Notation. We say that a resolution proof π is $X<Y$-ordered if it is $\prec$-ordered for $\prec$ defined as $\{x \prec y \mid x \in X, y \in Y\}$.

Observe that a partially-ordered resolution may not be regular nor ordered. For instance, $\{x_1, \ldots, x_n\}<\{y\}$-ordered enables resolution proofs where x_i variables are resolved on in an arbitrary order and may repeat—as long as the last resolution is on y.

Example 1. Consider the refutation in Figure 1(a). The refutation is $\{x, y\}<\{e\}$-ordered but it is not $\{y\}<\{x\}$-ordered nor $\{x\}<\{y\}$-ordered. In contrast, the refutation in Figure 1(b) is $\{e\}<\{x, y\}$-ordered.

3. SE-Projection Game Formulas

We consider the following game, called the *South-East Projection Game*.[1] It is a one-player game played on an $n \times n$ board, where the player places an arrow on each square (see Figure 2). A placed arrow either faces *south* or *east*. Whenever the arrow faces south, a ball is placed in the same column as the arrow, south of the board, i.e. the arrow is *projected south*. Analogously, placing an east arrow on a square results in placing a ball in the same

1. This concept is inspired by formulas previously used for lower bounds in QBF [10].

row. The player loses when there is a ball for every column or when there is a ball for every row.

It is easy to see that the player must necessarily lose the game. The only way how the player can avoid placing a ball on a row i, is by placing south arrows on each square of that row. This, however, leads to placing a ball in each column.

The CNF formula SEG_n, which encodes the game, is defined as follows. The following sets of variables are introduced. $X = \{x_{ij} \mid i, j \in 1..n\}$—for the direction of the arrows; $S = \{s_i \mid i \in 1..n\}$—for south balls; $E = \{e_i \mid i \in 1..n\}$—for east balls. To formula itself comprises the following clauses.

$$x_{ij} \vee s_i,\ i, j \in 1..n \tag{1}$$
$$\neg x_{ij} \vee e_j,\ i, j \in 1..n \tag{2}$$
$$\bigvee\nolimits_{i \in 1..n} \neg s_i \tag{3}$$
$$\bigvee\nolimits_{i \in 1..n} \neg e_i \tag{4}$$

We show that $(S \cup E){<}X$-ordered resolution refutation π is exponential with respect to n. We start off by some technical lemmas.

Lemma 1. *Let π be a $(S \cup E){<}X$-ordered resolution refutation of SEG_n. Let C be a clause in π such that it does not contain any s_i or e_i variables and let $\pi_C \subseteq \pi$ be a proof of C from SEG_n. The resolution proof π_C contains one of the clauses (3), (4).*

Proof. Construct a path P that starts at C and ends in some leaf of π_C. The path is constructed so that $\neg s_i$, resp. $\neg e_i$, is followed whenever s_i, resp. e_i, is resolved on. Since all axiom clauses (1), (2) contain s_i or e_i positively, one of (3), (4) must be at the end of P. □

Lemma 2. *Let π be a $(S \cup E){<}X$-ordered resolution refutation of SEG_n. Let P be a path in π from the root of π that contains clauses with only x_{ij} variables and is maximal in that respect, i.e. P cannot be extended by a clause that would have only x_{ij} variables. Let C be the clause in which P ends. The clause C that contains either a literal x_{ij} for each $i \in 1..n$ or literal $\neg x_{ij}$ for each $j \in 1..n$. Hence, C contains at least n different variables from X.*

Proof. Let π_C be the sub-proof of π proving C. Since C contains only x_{ij} variables, from Lemma 1 there is one of (3), (4) in π_C. Consider the case when (3) is in π_C. The clause (3) introduces $\neg s_i$ for each $i \in 1..n$. Since C does not contain any $\neg s_i$, all these must have been resolved on. As the only clauses containing the positive literals s_i are clauses of type (1), a literal x_{ij} must be introduced for each $i \in 1..n$. Since no literal x_{ij} could have been resolved on in π_C, all the introduced x_{ij} literals must appear also in C. Analogously, we derive that $\neg x_{ij} \in C$ for every $j \in 1..n$ if (4) is in π_C. □

Theorem 1. *Any $(S \cup E){<}X$-ordered resolution refutation of SEG_n is exponential in n.*

Proof. Let π be a $(S \cup E){<}X$-ordered resolution refutation of SEG_n. Pick an assignment τ to all variables x_{ij}. Construct a path P as follows. Start from $P = \bot$. If the end of P is derived by a resolution step over some variable x_{ij}, extend P with the antecedent that contains the literal l s.t. $\tau(l) = 0$ and $\mathsf{var}(l) = x_{ij}$. If the end of P is derived by a resolution

over some s_i, e_i variable, stop. Due to Lemma 2, the path P ends with a clause C_τ that contains n different variables x_{ij} and $\tau(C_\tau) = 0$. There are $2^{n\times n}$ different assignments τ to the x_{ij} variables and each clause C_τ covers at most $2^{n\times n-n}$ assignments. Hence there must be at least $2^{n\times n}/2^{n\times n-n} = 2^n$ clauses C_τ in π altogether. □

Theorem 2. *The formula* SEG_n *has polynomial size refutation in tree resolution. Further, this refutation is ordered.*

Proof. Pick an $i \in 1..n$. Using the clauses $x_{ij} \vee s_i$ and $\neg x_{ij} \vee e_j$ derive the clause $B_{ij} = s_i \vee e_j$ for all $j \in 1..n$ (resolution on x_{ij}). Using linear steps, resolve each B_{ij}, $j \in 1..n$ with the clause $\neg e_1 \vee \cdots \vee \neg e_n$ to obtain the clause s_i (resolution on e_i).

Repeat this process for all $i \in 1..n$, thus obtaining s_i for each. Using linear steps resolve each s_i the clause $\neg s_1 \vee \ldots \neg s_n$, obtaining thus $\bot$ (resolution on s_i). The resolution tree is ordered in the order $x_{11}, \ldots x_{nn}, e_1, \ldots, e_n, s_1, \ldots, s_n$. □

4. Function Encoding

For a set of variables X consider a Boolean function $f : X \to \{0, 1\}$ and the simple tautology $f(X) \vee \neg f(X)$. As usual we consider its negation $f(X) \wedge \neg f(X)$. We use this contradiction to derive a lower bound in the following way. We encode $f(X)$ and $\neg f(X)$ as two separate CNF formulas, where fresh auxiliary variables might be introduced. Then we consider a refutation of the conjunction of these two formulas where the auxiliary variables are resolved on first. We show that such refutation needs to derive implicants of f, which permit us to derive an exponential lower bound for functions with long implicants.[2.]

For a function f, let F^+ denote a formula so that $F^+ \restriction \tau$ is satisfiable iff $f(\tau) = 1$. Analogously, F^- denotes a formula for which $F^- \restriction \tau$ is satisfiable iff $f(\tau) = 0$. The formula F^+ may introduce auxiliary fresh variables in order to achieve polynomial encoding of f, this set will be denoted as T^+. Analogously, T^- denotes the fresh variables in F^-. Hence, we have $f = \exists T^+ . F^+$ and $\neg f = \exists T^- . F^-$. We assume that T^+ and T^- are disjoint.

Lemma 3. *Let* π *be a* $(T^+ \cup T^-)$<X*-ordered resolution refutation of* $F^+ \wedge F^-$*. There is no clause* C *in* π *that contains both* T^- *and* T^+ *variables.*

Proof. (by induction on derivation depth) The hypothesis is true for all of the axiom clauses, since they come from $F^+ \cup F^-$ and T^+ and T^- are disjoint.

The hypothesis is preserved by any resolution step on a variable $t \in T^- \cup T^+$ since both antecedents must contain t. Any resolution step on $x \in X$ must also preserve the hypothesis because none of the antecedents may contain a variable from $T^- \cup T^+$ due to the order condition. □

Lemma 4. *Let* π *be a* $(T^+ \cup T^-)$<X*-ordered resolution refutation of* $F^+ \wedge F^-$*. Any clause* C *in* π *that contains any* T^- *variables is an implicate of* F^-*. Analogously, a clause that contains any* T^+ *variables is an implicate of* F^+*.*

Proof. (by induction on derivation depth) Since T^+ and T^- are disjoint, any axiom clause that contains a variable from T^- must be from F^-, and therefore is trivially its implicate.

2. This idea is inspired by formulas previously used for lower bounds in QBF [3].

To show that the hypothesis is preserved by resolution step consider a C that contains some variables from T^- with the antecedents C_1 and C_2. We split on the following cases, the pivot variable is either from: T^+, T^-, or X. If the pivot is from T^-, then both C_1 and C_2 are implicates of F^- due to the induction hypothesis. Hence, C is also its implicate by the properties of resolution. If the pivot is from T^+, it must be contained in both antecedents but from Lemma 3 there are no T^- variables and therefore C wouldn't contain any T^- variables either. If the pivot were from X, C wouldn't contain any $T^- \cup T^+$ due to the condition on order of resolution. □

Lemma 5. *Let π be a $(T^+ \cup T^-)<X$-ordered resolution refutation of $F^+ \wedge F^-$. Let C be a clause in π such that it does not contain any $T^- \cup T^+$ variables and it is a resolvent of antecedents that contain a variable from $T^- \cup T^+$. Then $\neg C$ is an implicant of f or of $\neg f$.*

Proof. Let C_1 and C_2 be the antecedents of C. Since C_1 and C_2 contain a variable from $T^- \cup T^+$, but C does not contain any, it must be that the resolution on C_1 and C_2 is on some variable from $T^- \cup T^+$. First let us assume that the variable is from T^-. From Lemma 4, the clauses C_1 and C_2 are implicates of F^- and therefore C is also its implicate. Since C is an implicate of F^- that does not contain any auxiliary variables (it contains only the variables X), it is also an implicate of $\neg f$. Hence, $\neg C$ is an implicant of f.

Similarly we show that if C_1 and C_2 are resolved on T^+, then $\neg C$ is an implicant of $\neg f$. □

In the following we consider f as the parity function, i.e. $f(x_1, \ldots, x_n) = x_1 \oplus \cdots \oplus x_n$, where $\oplus$ is the *exclusive or* operation. We write $\mathsf{xor}(l_1, l_2, r)$ as a shorthand for the conjunction of the following clauses $\{\neg l_1 \vee \neg l_2 \vee \neg r, \neg l_1 \vee l_2 \vee r, l_1 \vee \neg l_2 \vee r, l_1 \vee l_2 \vee \neg r\}$. For a natural number $n > 3$, define the formula $\textsc{ParityCont}_n$ as the following CNF formula.

$$\begin{aligned} &\mathsf{xor}(x_1, x_2, t_2) \wedge \bigwedge\nolimits_{i \in 3..n} \mathsf{xor}(t_{i-1}, x_i, t_i) \\ \wedge\ &\mathsf{xor}(x_1, x_2, t'_2) \wedge \bigwedge\nolimits_{i \in 3..n} \mathsf{xor}(t'_{i-1}, x_i, t'_i) \\ \wedge\ &t_n \wedge \neg t'_n \end{aligned}$$

Theorem 3. *Any $(T^+ \cup T^-)<X$-ordered resolution refutation π of $\textsc{ParityCont}_n$ is exponential in n.*

Proof. We observe that $\textsc{ParityCont}_n$ is of the form $F^+ \wedge F^-$, which will enable us to use Lemma 5 to construct wide clauses in the refutation.

Pick a complete assignment τ to the variables X s.t. $f(\tau) = 1$, i.e. it assigns to 1 an odd number of variables. Starting from the clause $\bot$ in π construct a path that corresponds to the assignment τ, i.e. all clauses on the path are falsified by τ. The path ends in a clause C that contains only X variables and its antecedents contain some $T^+ \cup T^-$ variables. Hence, C satisfies the precondition of Lemma 5, from which we obtain that $\neg C$ is an implicant of f or $\neg f$. As the assignment τ was picked to be an implicant of f, $\neg C$ must also be an implicant of f. From the properties of the parity function, the implicant has to contain all variables X, i.e. $|C| = |X|$.

Since we picked τ arbitrarily out for the $2^{|X|-1}$ possible ones and there is only one clause C with the property above for a given τ, the refutation π must contain at least $2^{|X|-1}$ clauses. □

Theorem 4. *The formula* $\textsc{ParityCont}_n$ *has a polynomial resolution refutation. Further, this refutation is ordered.*

Proof. (sketch) Semantically, $\mathsf{xor}(x_1, x_2, t_2)$ corresponds to $t_2 \Leftrightarrow (x_1 \oplus x_2)$. This lets us derive $t_2 \Leftrightarrow t'_2$ from the clauses $\mathsf{xor}(x_1, x_2, t_2) \cup \mathsf{xor}(x_1, x_2, t'_2)$. In particular, by first greedily resolving on x_1 and then on x_2, obtaining, among others, the clauses $\neg t_2 \vee t'_2$ and $t_2 \vee \neg t'_2$.

Using the binary clauses above, by resolving on t'_2, replace in $\mathsf{xor}(t'_2, x_3, t'_3)$ the variable t'_2 with t_2, thus obtaining $\mathsf{xor}(t_2, x_3, t'_3)$. Just as above, derive $\neg t_3 \vee t'_3$ and $t_3 \vee \neg t'_3$.

Repeat this process until deriving $\neg t_n \vee t'_n$ and $t_n \vee \neg t'_n$. This gives contradiction for $i = n$ with the unit clauses t_n and $\neg t'_n$. The proof is ordered with the order $x_1, x_2, t'_2, t_2, x_3, t_3, \ldots$. □

5. Conclusion

This paper studies a restriction of propositional resolution where one partition of variables must be resolved on before the second partition. Since there are no further restrictions, such resolution proofs are not required to be ordered or even regular.

We have seen two different classes of formulas where such restriction gives an exponential lower bound, while the same formulas have short general resolution refutations. What is interesting about these classes is that the formulas are rather natural and have a clear interpretation. The first class of formulas we consider describes a simple one-player game, which is necessarily losing for the player. The first partition of variables models the results of the actions of the player and the second partition models the actions themselves. This partial order leads to an exponential blow up while the same formula has a polynomial unrestricted tree resolution. This contrasts with the known result that minimal tree resolutions are regular [20, Lem 5.1].

The second class of formulas deals with the simple contradiction $f \wedge \neg f$ for some Boolean function f. Once f is encoded using auxiliary (Tseitin) variables and these variables are resolved on first, we are forced to derive implicants of f, which leads to long proofs for functions with long implicants, e.g. the parity function. This result is contrasting to the results of Järvisalo and Junttila [11] who show that there are formulas for which it is harmful for a SAT solver to make decisions only on non-auxiliary variables. This suggests that in general, it is neither beneficial to start deciding on auxiliary variables nor on the non-auxiliary ones.

On the positive side, the presented formulas do have ordered refutations, with the catch that the right order is needed. An interesting case represents the formula $\textsc{ParityCont}_n$, whose short proof uses an order that interleaves the input and auxiliary variables.

Overall, these results further support the use of *dynamic decision orderings* in modern SAT solves (e.g. VSIDS [17], DLIS [15]). But we also hope, that these aforementioned contrasting theoretical results, might inspire further heuristics for decision ordering.

Acknowledgments.

This work was supported by FCT grants POLARIS (PTDC/EIA-CCO/123051/2010) and AMOS (CMUP-EPB/TIC/0049/2013), INESC-ID's multiannual PIDDAC funding PEst-OE/EEI/LA0021/2013.

References

[1] Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. *Theory of Computing*, **3**(5):81–102, 2007. http://www.theoryofcomputing.org/articles/v003a005.

[2] Paul Beame, Henry A. Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *J. Artif. Intell. Res. (JAIR)*, **22**:319–351, 2004. http://dx.doi.org/10.1613/jair.1410.

[3] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. Proof complexity of resolution-based QBF calculi. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, 2015. http://dx.doi.org/10.4230/LIPIcs.STACS.2015.76.

[4] Maria Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM J. Comput.*, **30**(5):1462–1484, 2000. http://dx.doi.org/10.1137/S0097539799352474.

[5] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, **44**(1):36–50, 1979. http://dx.doi.org/10.2307/2273702.

[6] Zvi Galil. On the complexity of regular resolution and the Davis-Putnam procedure. *Theor. Comput. Sci.*, **4**(1):23–46, 1977. http://dx.doi.org/10.1016/0304-3975(77)90054-8.

[7] Andreas Goerdt. Davis-Putnam resolution versus unrestricted resolution. *Ann. Math. Artif. Intell.*, **6**(1–3):169–184, 1992. http://dx.doi.org/10.1007/BF01531027.

[8] Andreas Goerdt. Unrestricted resolution versus N-resolution. *Theor. Comput. Sci.*, **93**(1):159–167, 1992. http://dx.doi.org/10.1016/0304-3975(92)90216-3.

[9] Armin Haken. The intractability of resolution. *Theor. Comput. Sci.*, **39**:297–308, 1985. http://dx.doi.org/10.1016/0304-3975(85)90144-6.

[10] Mikoláš Janota and Joao Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theoretical Computer Science*, **577**(0):25–42, April 2015. http://dx.doi.org/10.1016/j.tcs.2015.01.048.

[11] Matti Järvisalo and Tommi A. Junttila. Limitations of restricted branching in clause learning. *Constraints*, **14**(3):325–356, 2009. http://dx.doi.org/10.1007/s10601-008-9062-z.

[12] Matti Järvisalo, Tommi A. Junttila, and Ilkka Niemelä. Unrestricted vs restricted cut in a tableau method for Boolean circuits. *Ann. Math. Artif. Intell.*, **44**(4):373–399, 2005. http://dx.doi.org/10.1007/s10472-005-7034-1.

[13] Robert G. Jeroslow and Jinchang Wang. Solving propositional satisfiability problems. *Ann. Math. Artif. Intell.*, **1**:167–187, 1990. http://dx.doi.org/10.1007/BF01531077.

[14] Robert Kowalski and Donald Kuehner. Linear resolution with selection function. *Artificial Intelligence*, **2**(3):227–260, 1972. http://dx.doi.org/10.1016/0004-3702(71)90012-9.

[15] João P. Marques Silva. The impact of branching heuristics in propositional satisfiability algorithms. In *Progress in Artificial Intelligence (EPIA)*, **1695**, pages 62–74. Springer, 1999. http://dx.doi.org/10.1007/3-540-48159-1_5.

[16] João P. Marques Silva and Karem A. Sakallah. GRASP: A search algorithm for propositional satisfiability. *IEEE Trans. Computers*, **48**(5):506–521, 1999. http://doi.ieeecomputersociety.org/10.1109/12.769433.

[17] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. pages 530–535. ACM, 2001. http://doi.acm.org/10.1145/378239.379017.

[18] Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artif. Intell.*, **175**(2):512–525, 2011. http://dx.doi.org/10.1016/j.artint.2010.10.002.

[19] Grigory S. Tseitin. On the complexity of derivation in propositional calculus. In *Automation of Reasoning*, Symbolic Computation, pages 466–483. Springer, 1983. http://dx.doi.org/10.1007/978-3-642-81955-1_28.

[20] Alasdair Urquhart. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, **1**(4):425–467, 1995. http://www.math.ucla.edu/~asl/bsl/0104/0104-003.ps.