Computing Shortest Resolution Proofs*

Carlos Mencía[1] and Joao Marques-Silva[2]

[1] University of Oviedo, Gijón, Spain
menciacarlos@uniovi.es
[2] Faculty of Science, University of Lisbon, Lisbon, Portugal
jpms@ciencias.ulisboa.pt

Abstract. Propositional resolution is a powerful proof system for unsatisfiable propositional formulas in conjunctive normal form. Resolution proofs represent useful explanations of infeasibility, with important applications. This motivates the challenge of computing shortest resolution proofs, i.e. those with the smallest number of inference steps. This paper proposes a SAT-based approach for this problem. Concretely, the paper investigates new propositional encodings for computing shortest resolution proofs and devises a number of optimizations, including symmetry breaking, additional constraints on the structure of proofs, as well as exploiting related concepts in infeasibility analysis, such as minimal correction subsets. Experimental results show the suitability of the proposed approach.

1 Introduction

The importance of the propositional resolution proof system cannot be overstated, being at the core of modern conflict-driven clause learning (CDCL) Boolean satisfiability (SAT) solvers [20,19]. Propositional resolution can be traced to the seminal work of Davis and Putnam [11], and its formalization as a search procedure [10]. Perhaps more significantly, resolution finds fundamental applications in automated reasoning [31], representing one of the most widely used proof procedures in theorem proving [32].

Given an unsatisfiable propositional formula, a natural question is to find a shortest resolution proof, i.e. one with the fewest inference steps. Besides its theoretical interest [14,6], short refutations constitute useful certificates that explain infeasibility, and find important applications in system verification and validation (e.g. by the use of interpolation). In such settings, smaller proofs equate with smaller interpolants.

Computing shortest resolution refutations has been investigated from a theoretical perspective [1], including for restricted formulas [7,8]. From a practical perspective, albeit, to our knowledge, no work has addressed the computation of shortest resolution proofs per se, the related problem of finding unsatisfiability proofs with incomplete methods has been investigated in the past [27,4,26]. In a different context, computing optimal refutations for infeasible CSPs has been studied in [15], where the notion of a refutation is related with the search tree traversed for proving infeasibility.

* This research is supported by the Spanish Government under project TIN2016-79190-R and by the Principality of Asturias under grant IDI/2018/000176. This work is also supported by FCT grants ABSOLV (PTDC/CCI-COM/28986/2017) and FaultLocker (PTDC/CCI-COM/29300/2017).

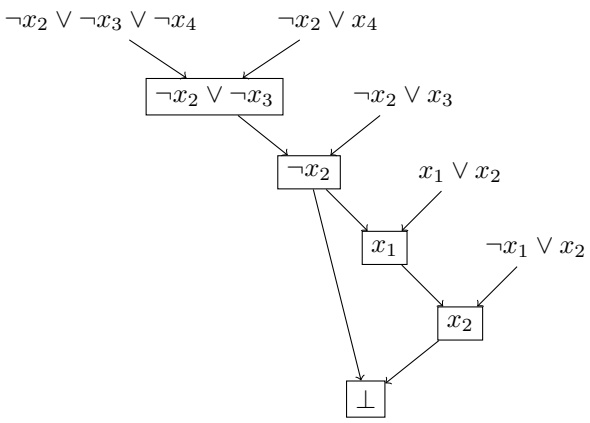


Fig. 1: Resolution proof (with 5 resolvents) for the formula in Example 1.

This paper investigates practical approaches for computing shortest resolution refutations, by iteratively solving the decision problem of whether there exists a resolution proof of size K, for increasing values of K. The straightforward solution of enumerating all possible candidate proofs of a given size is clearly unrealistic, given the sheer number of distinct proofs that need to be considered. This paper follows a different path, and proposes a SAT-based approach that uses SAT solvers in the search for such proofs. The paper builds on earlier work [27] and proposes a propositional encoding for solving the problem. However, whereas earlier work targeted encodings aiming at local search solvers, our proposed encodings target complete search algorithms. Furthermore, the paper devises a number of enhancements to the model, identifying ways to break relevant symmetries in the problem formulation and developing novel insights on how to effectively prune the search space. Experimental results show that the proposed model and its improvements enable computing shortest resolution proofs for formulas of non-trivial sizes.

The paper is organized as follows. The definitions and notation used in the paper are summarized in Section 2. Section 3 details a propositional model for computing shortest resolution proofs, and the enhancements to this model are described in Section 4. The experimental results are analyzed in Section 5. Finally, the paper concludes in Section 6.

2 Preliminaries

We consider propositional formulas in *conjunctive normal form* (CNF), defined as a conjunction, or set, of clauses $\mathcal{F} = \{c_1, c_2, ..., c_M\}$ over a set of variables $V(\mathcal{F}) = \{x_1, x_2, ..., x_N\}$, where a clause is a disjunction of literals, and a literal is a variable x or its negation $\neg x$. Throughout, for a clause $c \in \mathcal{F}$, $L^+(c)$ (resp. $L^-(c)$) denotes the positive (resp. negative) literals in c.

A formula $\mathcal{F}$ is satisfiable iff there exists a *model* for it, i.e. an assignment of truth values to the variables satisfying all the clauses. SAT is the NP-complete problem [9] of

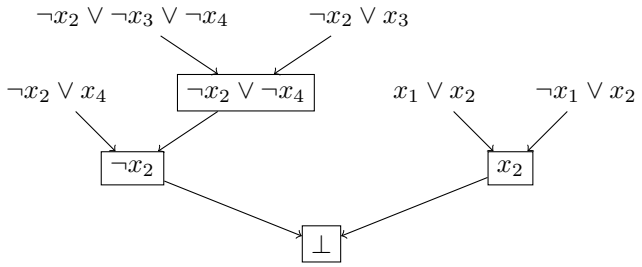


Fig. 2: A shortest resolution proof (with 4 resolvents) for the formula in Example 1.

deciding the satisfiability of a formula. If no model exists for $\mathcal{F}$, proofs represent certificates of its unsatisfiability. Resolution [31] is a well-known proof system for refuting unsatisfiable CNF formulas. It relies on the application of the resolution rule:

$$\frac{\Gamma_1 \vee x \quad \Gamma_2 \vee \neg x}{\Gamma_1 \vee \Gamma_2} \tag{1}$$

In Eq. (1), clauses $\Gamma_1 \vee x$ and $\Gamma_2 \vee \neg x$, referred to as *parents*, are resolved on variable x, inferring the clause $\Gamma_1 \vee \Gamma_2$, which is referred to as *resolvent*. A *resolution proof* or *refutation* is a sequence of clauses ending with the empty clause, each of these being either a clause in $\mathcal{F}$ or a resolvent from two previous clauses in the sequence. In the general case, resolution proofs can be represented as a DAG. This paper is concerned with shortest resolution proofs (SRPs), minimizing the number of resolvents. Computing SRPs is intractable, including the case of Horn formulas [1], although it can be solved in polynomial time for 2-CNF formulas [7,8].

Example 1. Consider the formula $\mathcal{F}_{ex} = \{(x_1 \vee x_2), (\neg x_1 \vee x_2), (\neg x_2 \vee x_3), (\neg x_2 \vee x_4), (\neg x_2 \vee \neg x_3 \vee \neg x_4)\}$, with 5 clauses and 4 variables. Figure 1 shows a resolution refutation of $\mathcal{F}_{ex}$ which uses 5 resolvents (highligthed in a square). Figure 2 shows a shortest resolution proof of the unsatisfiability of $\mathcal{F}_{ex}$, with 4 resolvents.

Besides proofs, other notions have been used as a means to explaining unsatisfiability, such as minimal unsatisfiable subformulas (MUSes) or their minimal hitting set duals [30,18], known as minimal correction subsets (MCSes).

Definition 1. *[MUS] $\mathcal{M} \subseteq \mathcal{F}$ is a* minimal unsatisfiable subformula *(MUS) of $\mathcal{F}$ if and only if $\mathcal{M}$ is unsatisfiable and $\forall c \in \mathcal{M}, \mathcal{M} \setminus \{c\}$ is satisfiable.*

Definition 2. *[MCS] $\mathcal{C} \subseteq \mathcal{F}$ is a* minimal correction subset *(MCS) of $\mathcal{F}$ if and only if $\mathcal{F} \setminus \mathcal{C}$ is satisfiable and $\forall c \in \mathcal{C}, \mathcal{F} \setminus (\mathcal{C} \setminus \{c\})$ is unsatisfiable.*

An MUS is an unsatisfiable subset such that removing any clause renders it satisfiable. Hence, an MUS represents an explanation for the unsatisfiability of $\mathcal{F}$. On the other hand, MCSes are irreducible sets of clauses whose removal renders the formula satisfiable. In Example 1, the formula $\mathcal{F}_{ex}$ represents an MUS itself, and each of the clauses in $\mathcal{F}_{ex}$ represents an MCS. Despite the high complexity of computing MCSes, *efficient* algorithms exist for this task (e.g. [17,5,22,21,28,29,13,23]). In the worst case, there can be an exponential number of MUSes and MCSes [25,16].

3 Deciding Fixed Size Resolution Proofs

For a fixed value K, deciding the existence of a resolution proof with K resolvents can be encoded into a propositional formula [27]. This motivates a SAT-based approach for computing SRPs by solving a sequence of decision problems with varying values of K.

This section describes a propositional model that produces a CNF formula that is satisfiable if and only if the original formula has a resolution refutation involving K resolvents. The original formula $\mathcal{F} = \{c_1, \ldots, c_M\}$ is defined on variables $V(\mathcal{F}) = \{x_1, \ldots, x_N\}$, and the number of resolvents considered is K, with the new clauses indexed from $M+1$ to $M+K$ (i.e., these being c_{M+1} to c_{M+K}).

Resolution proofs are encoded as a sequence of resolvents, preventing a node from being a parent of an earlier resolvent. The propositional model uses the following variables, where j ranges from 1 to N, r ranges from $M+1$ to $M+K$, and s can range from 1 to $M+K$, with $s < r$:

$p_{jr} = 1$ iff there exists a positive literal on variable x_j in resolvent r.
$n_{jr} = 1$ iff there exists a negative literal on variable x_j in resolvent r.
$v_{jr} = 1$ iff variable x_j is resolved away to obtain resolvent r.
$la_{rs} = 1$ iff the left parent of resolvent r is c_s, with $s < r$.
$ra_{rs} = 1$ iff the right parent of resolvent r is c_s, with $s < r$.
$wp_{jr} = 1$ iff a positive literal on x_j is available at resolvent r.
$wn_{jr} = 1$ iff a negative literal on x_j is available at resolvent r.

Notice that the clauses in $\mathcal{F}$ are not explicitly represented in the encoding, but only the list of resolvents in the proof, with information of their parents and the variables used at each resolution step. Variables wp_{jr} and wn_{jr} are used as an intermediate step in the application of the resolution rule, indicating that the literals exist in some parent of r. These are referred to as *working literals*.

The propositional model enforces the following constraints:

1. A resolvent cannot contain a variable and its complement, which is accomplished by the following AtMost1 constraints:

$$(\neg p_{jr} \vee \neg n_{jr}) \tag{2}$$

2. Computation of working literals of resolvents from the literals in the parents: If a parent of a resolvent has a positive (negative) literal, the corresponding working literal in the resolvent is activated.

$$\begin{aligned} (la_{rs} \vee ra_{rs}) \wedge p_{js} \rightarrow wp_{jr} \\ (la_{rs} \vee ra_{rs}) \wedge n_{js} \rightarrow wn_{jr} \end{aligned} \tag{3}$$

3. If the working literals exist, they must also exist in the declared parents.

$$\begin{aligned} wp_{jr} \rightarrow \textstyle\bigvee_s (la_{rs} \vee ra_{rs}) \wedge p_{js} \\ wn_{jr} \rightarrow \textstyle\bigvee_s (la_{rs} \vee ra_{rs}) \wedge n_{js} \end{aligned} \tag{4}$$

4. Definition of literals in resolvents: A resolvent will include all its working literals except those on the variable resolved away.

$$\begin{array}{l} p_{jr} \leftrightarrow wp_{jr} \wedge \neg v_{jr} \\ n_{jr} \leftrightarrow wn_{jr} \wedge \neg v_{jr} \end{array} \tag{5}$$

5. Definition of resolution variable given working literals:

$$v_{jr} \leftrightarrow wp_{jr} \wedge wn_{jr} \tag{6}$$

6. Resolvents have exactly one left and one right parent, chosen among earlier clauses.

$$\begin{array}{l} \sum_{s=1}^{r-1} la_{rs} = 1 \\ \sum_{s=1}^{r-1} ra_{rs} = 1 \end{array} \tag{7}$$

7. Exactly one variable is resolved away for creating each resolvent.

$$\sum_{j=1}^{N} v_{jr} = 1 \tag{8}$$

8. The final resolvent, with index $M + K$, must be the empty clause.

$$\neg p_{jM+K} \wedge \neg n_{jM+K} \tag{9}$$

For the sake of clarity, in equations (3) and (4) there is a slight abuse of notation. Note that p_{js} and n_{js} are not defined for $s \leq M$. In these cases, constraints are added *on demand*, without the p_{js} and n_{js} terms. For instance, in (3) the clause $c_s \in \mathcal{F}$, yields the constraints $(la_{rs} \vee ra_{rs}) \rightarrow wp_{jr}$ for $j \in L^+(c_s)$ and $(la_{rs} \vee ra_{rs}) \rightarrow wn_{jr}$ for $j \in L^-(c_s)$. The sets $L^+(c_s)$ and $L^-(c_s)$ are also exploited in (4). Besides, transforming (4) into CNF requires adding $\mathcal{O}(NK^2)$ new variables and clauses encoding the equivalences $yp_{rsj} \leftrightarrow (la_{rs} \vee ra_{rs}) \wedge p_{js}$ and $yn_{rsj} \leftrightarrow (la_{rs} \vee ra_{rs}) \wedge n_{js}$, with $s > M$.

Cardinality constraints of the form $\sum_{i=1}^{n} x_i = 1$ are encoded as the conjunction of one *at least one* constraint, represented by the single clause $\bigvee_{i=1}^{n} x_i$, and one *at most one* constraint, which is encoded using *sequential counters* [33]. For a constraint $\sum_{i=1}^{n} x_i \leq k$, sequential counters produce $\mathcal{O}(nk)$ variables and clauses. Clearly, other encodings could be considered as well (e.g. [3,24]).

The encoded formula has $\mathcal{O}(NK^2 + MK)$ variables. Regarding the number of clauses, constraints (3) and (4) result in $\mathcal{O}(NK^2 + K||\mathcal{F}||)$ clauses, with $||\mathcal{F}||$ the number of literals in $\mathcal{F}$. Constraints (2),(5), (6) and (8) result in $\mathcal{O}(NK)$ clauses (7) amounts to $\mathcal{O}(K(M + K))$ clauses and (9) yields $\mathcal{O}(N)$ clauses. The encoding has $\mathcal{O}(NK^2+K||\mathcal{F}||+K(M+K))$ clauses and $\mathcal{O}(NK(M+K)+K||\mathcal{F}||+K(M+K))$ literals, the increase due to some large clauses from (4).

The encoding is related to the model proposed in [27], with several differences. In [27] the encoding targeted computing resolution proofs with local search. It used refinements to reduce its size and increase the solution density. These included not

enforcing that literals in a resolvent must exist in some parent (known as the *weakening rule*) or allowing for more than two parents for a node. The resulting encoding had $\mathcal{O}(K^2 + KN + KM)$ variables and $\mathcal{O}(NK^2 + K||\mathcal{F}||)$ literals, smaller than the one herein. However, when looking for SRPs with a complete solver it is necessary to prove unsatisfiabilty for some values of K, and constraining formulas as much as possible, promoting propagation, is beneficial. So, we opted for not using the weakening rule, at the expense of a larger encoding. Anyway, the results in [27] showed that applying the weakening rule was not beneficial in practice. The proposed encoding represents proofs more explicitly than in [27], e.g. by distinguishing left and right parents, which is useful for enforcing additional constraints, as shown in the next section.

4 Enhancements to the Model

This section devises a number of enhancements to the model above. These constrain the structure of resolution proofs, remove symmetries and exploit the information given by a collection of MCSes to prune the search space.

4.1 Constraints on the structure of the proofs

We first enforce symmetry breaking constraints, establishing that the left parent of any resolvent must have a lower index than its right parent. Any resolution DAG can be rewritten without additional nodes to fulfill this property. Besides, for each clause $c_s \in \mathcal{F}$, we compute beforehand the set of clauses $R(c_s) \subseteq \mathcal{F}$ that can be resolved with c_s producing a valid resolvent, which restricts the selection of parents as follows:

$$la_{rs} \rightarrow \bigvee_{t \in R(c_s); t > s}^{M} ra_{rt} \vee \bigvee_{t=M+1; t>s}^{r-1} ra_{rt} \tag{10}$$

The following constraints avoid computing resolvents that are subsumed by any original clause in $\mathcal{F}$, which would not be useful in the proof. Here $s \in [1, M]$.

$$\bigvee_{j \in L^+(c_s)} \neg p_{jr} \vee \bigvee_{j \in L^-(c_s)} \neg n_{jr} \tag{11}$$

In an SRP, all resolvents but the last one are used as a parent of later resolvents [27] (otherwise, such resolvent would be useless). Imposing these constraints has the drawback that for some values of K overestimating the length of the SRPs, the encoded formula may be unsatisfiable, thus requiring the search method to iteratively refine lower bounds from 1 on. Here, $r \in [M + 1, M + K - 1]$.

$$\bigvee_{u=r+1}^{M+K} (la_{ur} \vee ra_{ur}) \tag{12}$$

Equations (10) and (12) entail that resolvent $K - 1$ must be the right parent of K. This is enforced by setting $ra_{(M+K)(M+K-1)} = 1$. Besides, both parents of K must

be unit clauses. The following constraints prevent the left parent of K from being a non-unit input clause. $U(\mathcal{F})$ denotes the unit clauses in $\mathcal{F}$.

$$\bigwedge_{s \in \mathcal{F} \setminus U(\mathcal{F})} \neg la_{(M+K)s} \tag{13}$$

In addition we add constraints enforcing the resolvent $K-1$ (the right parent of K) to be a unit clause.

$$\sum_{j=1}^{N} p_{j(M+K-1)} + n_{j(M+K-1)} = 1 \tag{14}$$

The constraints above do not increase the asymptotic number of variables and clauses of the encoding. However, the $\mathcal{O}(K(M+K))$ clauses from (10) can be large, adding in total $\mathcal{O}(K(M+K)^2)$ literals to the encoding in the worst case.

4.2 Levels in the resolution DAG

Nodes in the resolution DAG can be associated a *level*. Original clauses have level 0, whereas the level of a resolvent is given by the maximum of the levels of its parents plus 1. We focus on the property of having level 1, indicating that both its parents are original clauses. The following result establishes bounds on the number of resolvents of each kind:

Proposition 1. *In an SRP with K resolvents, at least $\lfloor K/2 \rfloor$ resolvents do not have level 1.*

Proof. In an SRP, all resolvents but the last one are used in the proof. Hence, $K-1$ items need to be allocated at least once as a parent of other resolvent. Nodes at level 1 have both parents as input clauses. Suppose there are $\lfloor K/2 \rfloor$ resolvents not at level 1. In the worst case K is odd, so $\lfloor K/2 \rfloor = (K-1)/2$. Each of these nodes has two parents, so there are $K-1$ positions for the $K-1$ items. This represents the limit case; if there were fewer nodes at level greater than 1, not all $K-1$ items could be used in the proof.

From these observations, we further restrict the search space. For each resolvent r, a variable is defined as $l1_r = 1$ iff resolvent r is at level 1. Then, we set that the first node has level 1, i.e. $l1_{M+1} = 1$, and that the last $\lfloor K/2 \rfloor$ nodes are not at level 1, i.e. $l1_r = 0$ for $r \in [h+1, M+K]$, where $h = M + \lceil K/2 \rceil$. The following constraints reduce the number of possible parents depending on levels and break symmetries:

1. A resolvent is at level 1 if and only if its right parent is not an earlier resolvent.

$$l1_r \leftrightarrow \bigwedge_{s=M+1}^{r-1} \neg ra_{rs} \tag{15}$$

2. All the resolvents at level 1 are computed at the beginning.

$$\begin{aligned} l1_r &\rightarrow \textstyle\bigwedge_{u=M+2}^{r-1} l1_u \\ \neg l1_r &\rightarrow \textstyle\bigwedge_{u=r+1}^{h} \neg l1_u \end{aligned} \tag{16}$$

3. If two consecutive resolvents are at level 1, the left parent of the first one has an index not greater than the left parent of the second one. Here $r \in [M+2, h]$.

$$(l1_r \wedge la_{rs}) \rightarrow \bigvee_{t=1}^{s} la_{(r-1)t} \tag{17}$$

Enforcing constraints (16) and (17) does not prevent from finding a shortest refutation, since any resolution DAG can be transformed to fulfill these conditions by reordering some nodes in the proof. Regarding space, the constraints above do not affect the asymptotic number of variables, clauses or literals in the encoding. Constraints (17) result in $\mathcal{O}(K(M+K)^2)$ literals, matching those analyzed previously from (10).

4.3 Exploiting Minimal Correction Subsets

The last enhancement proposed in the paper is aimed at further reducing the search space by exploiting minimal correction subsets (MCSes) in the encoding. It is based on the following result:

Proposition 2. *Let $\mathcal{C} \subsetneq \mathcal{F}$ be an MCS. All resolution proofs of $\mathcal{F}$ use some clause in $\mathcal{C}$.*

Proof. By Definition 2, $\mathcal{F} \setminus \mathcal{C}$ is satisfiable. Hence, if all the clauses in $\mathcal{C}$ were dropped, there would not exist a proof of unsatisfiability.

So, in a pre-processing step a collection of MCSes is enumerated, and for each MCS $\mathcal{C}$ we enforce that at least one of its clauses is used at least once in the proof:

$$\bigvee_{r=M+1}^{M+K} \Big(\bigvee_{c_s \in \mathcal{C}} la_{rs} \vee ra_{rs} \Big) \tag{18}$$

We exploit the fact that for a proof with K resolvents, at most $2K$ clauses can be used. We define, for each clause c_s, a variable $used_s$ = 1 if c_s is used at least once in the proof, and add the following constraints:

1. All the resolvents, but the last one must be used for computing later resolvents in the proof.

$$\bigwedge_{r=M+1}^{M+K-1} (used_r) \tag{19}$$

2. At least one clause in each MCS must be used in the computation of some resolvent. For each MCS $\mathcal{C}$ we add a clause as the following:

$$\bigvee_{c_s \in \mathcal{C}} used_s \tag{20}$$

3. If a clause is a parent of a resolvent, it is marked as used.

$$(la_{rs} \vee ra_{rs}) \rightarrow used_s \tag{21}$$

4. At most $2K$ different clauses can be used.

$$\sum_{s=1}^{M+K-1} used_s \leq 2K \tag{22}$$

Constraints (18) produce one clause of size $\mathcal{O}(K(M+K))$ for each MCS. So, if C MCSes are considered, the encoding grows in $\mathcal{O}(C)$ clauses and $\mathcal{O}(CK(M+K))$ literals. The last constraints do not affect the asymptotic size of the model. The final encoding has $\mathcal{O}(NK^2 + MK)$ variables, $\mathcal{O}(NK^2 + K||\mathcal{F}|| + K(M+K) + C)$ clauses and $\mathcal{O}(NK(M+K) + K||\mathcal{F}|| + CK(M+K) + K(M+K)^2)$ literals.

5 Experimental Results

This section evaluates the proposed encodings for computing SRPs. For this purpose, we implemented a prototype in Python 2.7, interfacing the SAT solver minisat (v 2.2) [12]. The tool computes SRPs by iteratively refining lower bounds on the number of resolvents. Starting with $K = 1$, while the encoded formula is found unsatisfiable K is increased in one unit. The process terminates upon a satisfiable call, in which case an SRP is extracted from the computed model. Notice that this procedure iteratively proves that no resolution refutation of size K exists for increasing values of K, until the last iteration where resolution proof is found. This way, the computed resolution proof is guaranteed to be an SRP.

The experiments have been carried out over a set of unsatisfiable Horn formulas (whose clauses contain at most one positive literal) derived from the domain of axiom pinpointing in lightweight description logics [2]. We considered the ones with a number of clauses in the range $[20, 594]$, making 278 instances in all. The number of variables ranges from 17 to 493. A number of these formulas have short proofs, so they represent an adequate benchmark for assessing the effectiveness of the encodings. Recall that computing SRPs for Horn formulas is intractable [1] (in contrast to 2-CNF formulas [7,8]). We distinguish different versions of the encoding: B refers to the base encoding from Section 3, S includes the additional constraints from Section 4.1, L adds the constraints related to levels from Section 4.2 and versions with prefix X exploit MCSes as well, as described in Section 4.3. X_1, X_{10} and X_{100} establish a limit on the number of MCSes enumerated to 100, 1000 and 10000 respectively. MCSes are computed with the tool mcsls [17]. All the experiments were run on a Linux cluster, with a time limit of 600 seconds. The computation of MCSes is included in the time limit. To this respect, the average (maximum) time taken for enumerating MCSes was 0.01 (0.14), 0.08 (0.58) and 0.83 (5.74) seconds for X_1, X_{10} and X_{100} respectively. Besides, complete enumeration of MCSes was possible for a number cases: Out of the 278 instances, 130, 156 and 188 have less than 100, 1000 and 10000 MCSes respectively.

Figure 3 shows the running times taken by each of the aforementioned versions of the encoding in solving the considered instances. For a given version of the encoding, a point (x, y) in the plot indicates that x instances were solved taking up to y seconds. As we can observe, the basic encoding B yields the worst results overall, solving 49 instances. The optimizations included in S and L allow for computing SRPs for more instances (87 and 93 respectively). Although L is able to solve a only few more instances

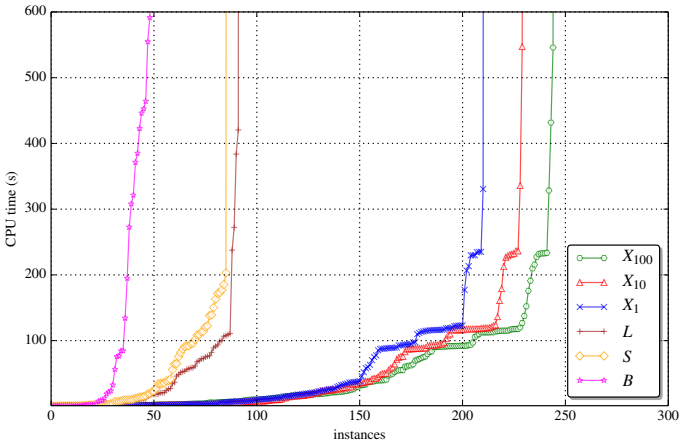
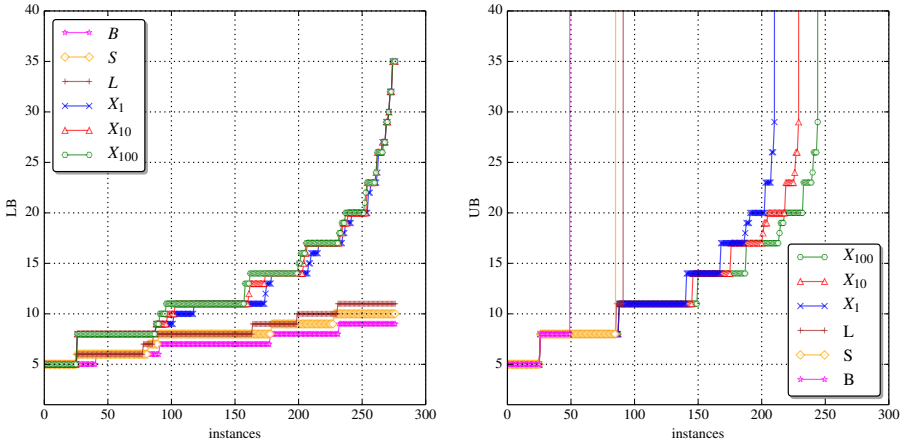


Fig. 3: Running times (s).



(a) Lower bounds (b) Upper bounds

Fig. 4: Summary of results.

than S, there is an observable gain in terms of running times. Noticeably, exploiting MCSes brings the most significant improvements, these being directly related to the number of MCSes enumerated beforehand. X_1, X_{10} and X_{100} can cope with far more challenging instances than before, solving 212, 231 and 246 instances respectively.

Figure 4 provides a more detailed view on the performance of the encodings. Figure 4a reports the best lower bounds on the size of the SRPs (LBs) computed by the time limit, and Figure 4b shows the actual size of the SRPs (UBs) computed. For a given version of the encoding, a point (x, y) in Figure 4a means that an LB lower than or equal to y was obtained for x instances using such encoding within the time limit. Analo-

gously, in Figure 4b, a point (x, y) means that the given encoding enabled computing SRPs with up to y resolvents for x instances. For the solved instances, LBs match the size of the SRPs. As we can observe, B is only able to cope with formulas with very short proofs, computing SRPs with 5 to 8 resolvents and proving LBs of at most 9 for some instances. On the other hand, S and L are capable of proving larger LBs (up to values 10 and 11 respectively), being L able to find SRPs with 11 resolvents for a few instances. Exploiting MCSes results in very significant gains, proving LBs of size 35 and computing SRPs with up to 29 resolvents.

The proposed improvements also allow for coping with larger formulas, solving several of the largest ones in the set (with more than 350 variables and near 600 clauses). These results indicate that the proposed enhancements are effective at pruning the search space and guiding the construction of resolution proofs, enabling the computation of SRPs for non-trivial formulas.

6 Conclusions

This paper addresses the problem of computing shortest resolution proofs (SRPs) of unsatisfiable CNF formulas and develops a SAT-based approach for this task. SRPs are computed by solving a sequence of decision problems encoding the computation of resolution refutations of fixed size K, with increasing values of K. Building on an initial reference propositional model, the paper devises several enhancements to the encoding, which constrain the structure of the proofs and exploit minimal correction subsets to further reduce the search space. The enhancements are shown to be very effective in practice, enabling the computation of SRPs for more challenging formulas.

References

1. Alekhnovich, M., Buss, S.R., Moran, S., Pitassi, T.: Minimum propositional proof length is NP-hard to linearly approximate. J. Symb. Log. **66**(1), 171–191 (2001)
2. Arif, M.F., Mencía, C., Marques-Silva, J.: Efficient MUS enumeration of Horn formulae with applications to axiom pinpointing. In: SAT. pp. 324–342 (2015)
3. Asín, R., Nieuwenhuis, R., Oliveras, A., Rodríguez-Carbonell, E.: Cardinality networks: a theoretical and empirical study. Constraints **16**(2), 195–221 (2011)
4. Audemard, G., Simon, L.: GUNSAT: A greedy local search algorithm for unsatisfiability. In: IJCAI. pp. 2256–2261 (2007)
5. Bacchus, F., Davies, J., Tsimpoukelli, M., Katsirelos, G.: Relaxation search: A simple way of managing optional clauses. In: AAAI. pp. 835–841 (2014)
6. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow - resolution made simple. J. ACM **48**(2), 149–169 (2001)
7. Buresh-Oppenheim, J., Mitchell, D.G.: Minimum witnesses for unsatisfiable 2CNFs. In: SAT. pp. 42–47 (2006)
8. Buresh-Oppenheim, J., Mitchell, D.G.: Minimum 2CNF resolution refutations in polynomial time. In: SAT. pp. 300–313 (2007)
9. Cook, S.A.: The complexity of theorem-proving procedures. In: STOC. pp. 151–158 (1971)
10. Davis, M., Logemann, G., Loveland, D.W.: A machine program for theorem-proving. Commun. ACM **5**(7), 394–397 (1962)

11. Davis, M., Putnam, H.: A computing procedure for quantification theory. J. ACM **7**(3), 201–215 (1960)
12. Eén, N., Sörensson, N.: An extensible SAT-solver. In: SAT. pp. 502–518 (2003)
13. Grégoire, É., Izza, Y., Lagniez, J.: Boosting MCSes enumeration. In: IJCAI. pp. 1309–1315 (2018)
14. Haken, A.: The intractability of resolution. Theor. Comput. Sci. **39**, 297–308 (1985)
15. Hulubei, T., O'Sullivan, B.: Optimal refutations for constraint satisfaction problems. In: IJCAI. pp. 163–168 (2005)
16. Liffiton, M.H., Sakallah, K.A.: Algorithms for computing minimal unsatisfiable subsets of constraints. J. Autom. Reasoning **40**(1), 1–33 (2008)
17. Marques-Silva, J., Heras, F., Janota, M., Previti, A., Belov, A.: On computing minimal correction subsets. In: IJCAI. pp. 615–622 (2013)
18. Marques-Silva, J., Janota, M., Mencía, C.: Minimal sets on propositional formulae. Problems and reductions. Artif. Intell. **252**, 22–50 (2017)
19. Marques-Silva, J., Malik, S.: Propositional SAT solving. In: Handbook of Model Checking, pp. 247–275. Springer (2018)
20. Marques-Silva, J., Sakallah, K.A.: GRASP - a new search algorithm for satisfiability. In: ICCAD. pp. 220–227 (1996)
21. Mencía, C., Ignatiev, A., Previti, A., Marques-Silva, J.: MCS extraction with sublinear oracle queries. In: SAT. pp. 342–360 (2016)
22. Mencía, C., Previti, A., Marques-Silva, J.: Literal-based MCS extraction. In: IJCAI. pp. 1973–1979 (2015)
23. Narodytska, N., Bjørner, N., Marinescu, M., Sagiv, M.: Core-guided minimal correction set and core enumeration. In: IJCAI. pp. 1353–1361 (2018)
24. Ogawa, T., Liu, Y., Hasegawa, R., Koshimura, M., Fujita, H.: Modulo based CNF encoding of cardinality constraints and its application to MaxSAT solvers. In: ICTAI. pp. 9–17 (2013)
25. O'Sullivan, B., Papadopoulos, A., Faltings, B., Pu, P.: Representative explanations for over-constrained problems. In: AAAI. pp. 323–328 (2007)
26. Pereira, D., Lynce, I., Prestwich, S.D.: On improving local search for unsatisfiability. In: LSCS. pp. 41–53 (2009)
27. Prestwich, S.D., Lynce, I.: Local search for unsatisfiability. In: SAT. pp. 283–296 (2006)
28. Previti, A., Mencía, C., Järvisalo, M., Marques-Silva, J.: Improving MCS enumeration via caching. In: SAT. pp. 184–194 (2017)
29. Previti, A., Mencía, C., Järvisalo, M., Marques-Silva, J.: Premise set caching for enumerating minimal correction subsets. In: AAAI. pp. 6633–6640 (2018)
30. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence **32**(1), 57–95 (1987)
31. Robinson, J.A.: A machine-oriented logic based on the resolution principle. J. ACM **12**(1), 23–41 (1965)
32. Robinson, J.A., Voronkov, A. (eds.): Handbook of Automated Reasoning. Elsevier and MIT Press (2001)
33. Sinz, C.: Towards an optimal CNF encoding of boolean cardinality constraints. In: CP. pp. 827–831 (2005)