Augmenting the Power of (Partial) MaxSat Resolution with Extension[*]

Javier Larrosa, Emma Rollon
Universitat Politècnica de Catalunya,
Jordi Girona 1-3, Barcelona 08034
{larrosa, erollon}@cs.upc.edu

Abstract

The refutation power of SAT and MaxSAT resolution is challenged by problems like the *soft* and *hard Pigeon Hole Problem* PHP for which short refutations do not exist. In this paper we augment the MaxSAT resolution proof system with an *extension* rule. The new proof system MaxResE is sound and complete, and more powerful than plain MaxSAT resolution, since it can refute the soft and hard PHP in polynomial time. We show that MaxResE refutations actually subtract lower bounds from the objective function encoded by the formulas. The resulting formula is the *residual* after the lower bound extraction. We experimentally show that the residual of the soft PHP (once its necessary cost of 1 has been efficiently subtracted with MaxResE) is a concise, easy to solve, satisfiable problem.

Introduction

The MaxSAT resolution proof system (Larrosa and Heras 2005; Bonet, Levy, and Manyà 2007; Larrosa, Heras, and de Givry 2008) generalizes SAT resolution (Robinson 1965) allowing it to reason with both *hard* and *soft* clauses. A refutation under this proof system corresponds to a subtraction of a lower bound of the optimum from the formula. The remaining formula is called *residual* since it is what remains after the extraction. Although MaxSAT is a more general language than SAT, it is known that it does not improve over problems such as the hard and soft *Pigeon Hole Problems* PHP for which no short refutations exist.

In this paper we augment the MaxSAT resolution proof system with an *extension* rule. Roughly, the new rule allows to reason in the realm of negative weights, which is not possible using just resolution. We show that the new proof system **MaxResE** is sound and complete. The potential of MaxResE is demonstrated by the fact that it can produce short refutations for the hard and soft PHP.

We conjecture that a potential application of MaxResE is to extract from a formula lower bounds that would not be obtained efficiently otherwise, and solve the residual with an off-the-shelf solver. We demonstrate this idea with the soft PHP: The size of the original formula is $O(m^3)$ with m being the number of holes. We show that a lower bound of 1 can be extracted in $O(m^3)$ inference steps producing a residual formula of size $O(m^4)$. We show experimentally that the satisfiability of the residual can be proved in time linear on its size (i.e, $O(m^4)$).

Our work shares some similarities with the *dual rail encoding* (Ignatiev, Morgado, and Marques-Silva 2017), a method to refute unsatisfiable SAT instances by transforming them into horn MaxSAT. Our approach can do similar SAT refutations (as we show when we address the hard PHP) but it is more general since it can also compute lower bounds, compute optimums and proof unsatisfiability in MaxSAT formulas.

Our approach also shares similarities with the notion of *soft arc consistency* (Cooper et al. 2010) in the context of *Cost Networks* where weights are soundly propagated and, as we do in this paper, weights may be negative during the process as long as they are positive at the end. The main difference is that MaxResE is a complete (possibly exponentially costly) method, while soft arc consistency is incomplete but polynomial method.

Preliminaries

MaxSAT

In this paper we will denote boolean variables as $x, p, q, \ldots$. A *literal* is a boolean variable (positive literal) or its negation (negative literal) and a *clause* is a disjunction of literals. We will denote (parts of) clauses as $A, B, C, \ldots$. The *empty clause*, noted $\square$, cannot be satisfied and will play a special role in this paper. A *weighted clause* is a pair (C, w) where C is a clause and w is its weight given as a positive natural. The weight indicates the cost of falsifying it.

A MaxSAT *formula* $\mathcal{F}$ is a set of weighted clauses and the (weighted) *MaxSAT problem* corresponds to finding the truth assignment that minimizes the sum of weights of falsified weighted clauses. We will write $MaxSAT(\mathcal{F})$ to denote that minimum value.

MaxSAT formulas can be simplified using simple rules such as the elimination of tautological clauses, or replacing

[*]This work was funded partially by the Spanish MINECO and MICINN projects TIN2015-69175-C4-3-R and RTI2018-094403-B-C33.

pairs of identical clauses $(C, u), (C, v)$ by $(C, u + v)$.

Abusing notation sometimes we will write $(A \vee \neg(l_1 \vee l_2 \vee \cdots l_r), w)$ with $r > 1$, which means that a cost w is incurred if $A \vee \neg(l_1 \vee l_2 \vee \cdots l_r)$ is falsified. It is a short-hand for its clausal equivalent $\{(A \vee \neg l_1, w), (A \vee l_1 \vee \neg l_2, w), \ldots, (A \vee l_1 \vee l_2 \vee \ldots \vee l_{r-1} \vee \neg l_r, w)\}$.

MaxSAT formulas may contain the *weighted empty clause* $(\Box, k)$. We will often refer to formulas as,

$$\mathcal{F} = \{(\Box, k)\} \cup \mathcal{F}^\delta$$

making explicit their weighted empty clause. If $\mathcal{F}$ does not contain the empty clause, then $k = 0$ and $\mathcal{F} = \mathcal{F}^\delta$. Note that, since $\Box$ is unsatisfiable, k is a MaxSAT lower bound (i.e, if $(\Box, k) \in \mathcal{F}$ then $k \leq MaxSAT(\mathcal{F})$).

MaxSAT Resolution

MaxSAT resolution (Larrosa and Heras 2005; Larrosa, Heras, and de Givry 2008) is the generalization of standard resolution (Robinson 1965) to MaxSAT. It is written as,

$$\frac{(x \vee A, v) \quad (\neg x \vee B, w)}{\begin{array}{c}(A \vee B, m) \\ (x \vee A, v - m) \quad (\neg x \vee B, w - m) \\ (x \vee A \vee \neg B, m) \quad (\neg x \vee B \vee \neg A, m)\end{array}}$$

where $m = \min\{v, w\}$. When A (resp. B) is empty, $\neg A$ (resp. $\neg B$) is constant true, so $x \vee \neg A \vee B$ (resp. $x \vee A \vee \neg B$) is tautological. In MaxSAT resolution the clauses at the top are *replaced* by the clauses at the bottom in the formula.

Example 1. *The application of MaxSAT resolution to $(x \vee y \vee z, 2)$ and $(\neg x \vee y \vee p, 1)$ corresponds to,*

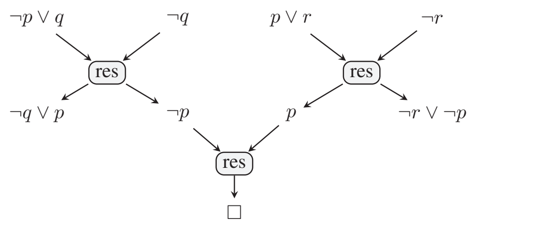
$$\frac{(x \vee y \vee z, 2) \quad (\neg x \vee y \vee p, 1)}{\begin{array}{c}(y \vee z \vee p, 1) \\ (x \vee y \vee z, 1) \quad (\neg x \vee y \vee p, 0) \\ (x \vee y \vee z \vee \neg y, 1) \quad (\neg x \vee \neg y \vee y \vee p, 1) \\ (x \vee y \vee z \vee y \vee \neg p, 1) \quad (\neg x \vee y \vee \neg z \vee y \vee p, 1)\end{array}}$$

Removing zero-cost clauses, tautologies and repeated literals, the resulting set of clauses is $\{(y \vee z \vee p, 1), (x \vee y \vee z, 1), (x \vee y \vee z \vee \neg p, 1), (\neg x \vee y \vee \neg z \vee p, 1)\}$.

A *proof* of *length* e under the MaxSAT resolution proof system **MaxRes** is a finite sequence $\mathcal{F}_0 \vdash \mathcal{F}_1 \vdash \ldots \vdash \mathcal{F}_e$ where $\mathcal{F}_0$ is the original formula and each $\mathcal{F}_i$ is obtained by applying resolution to two clashing clauses in $\mathcal{F}_{i-1}$ (simplification rules are assumed to be applied implicitly). We will use $\vdash^*$ to denote an arbitrary number of inference steps. A short proof is a proof whose length can be bounded by a polynomial on $|\mathcal{F}|$.

Theorem 1. *(Soundness (Larrosa, Heras, and de Givry 2008)) Given a MaxRes proof $\mathcal{F}_0 \vdash \mathcal{F}_1 \vdash \ldots \vdash \mathcal{F}_e$, we have that $MaxSAT(\mathcal{F}_0) = MaxSAT(\mathcal{F}_i)$ for all $0 < i \leq e$.*

Let $\mathcal{F}_0$ be a formula not containing the empty clause and such that $MaxSAT(\mathcal{F}_0) > 0$. A MaxRes *refutation* is a proof $\mathcal{F}_0 \vdash^* \{(\Box, k)\} \cup \mathcal{F}_e^\delta$ with $k > 0$. We call formula $\mathcal{F}_e^\delta$ the *residual* of $\mathcal{F}_0$. When $MaxSAT(\mathcal{F}_e^\delta) = 0$, due to soundness we know that $k = MaxSAT(\mathcal{F}_0)$. In that case, we say that the refutation is *full*.



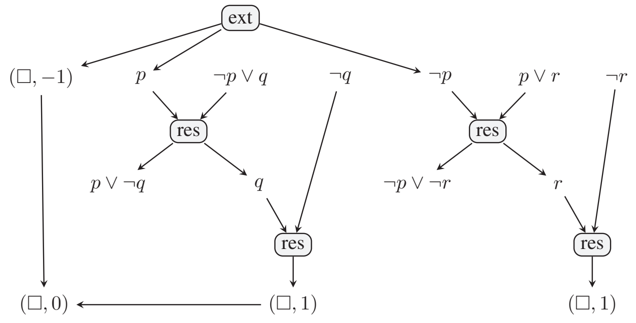


Figure 1: DAG representation of a MaxRes refutation (top) and a MaxResE refutation (bottom). All clauses have cost 1 unless otherwise indicated.

Theorem 2. *(Completeness (Bonet, Levy, and Manyà 2007; Larrosa, Heras, and de Givry 2008)) For any formula $\mathcal{F}_0$ not containing the empty clause and such that $MaxSAT(\mathcal{F}_0) = k > 0$, there is a MaxRes full refutation $\mathcal{F}_0 \vdash^* \{(\Box, k)\} \cup \mathcal{F}_e^\delta$.*

Example 2. *Refutations can be graphically represented as directed acyclic graphs DAGs, where parentless nodes are original clauses, childless nodes are final clauses, circled nodes correspond to resolution steps (parents are replaced by children). Figure 1 (top) depicts a length 3 refutation of formula $\mathcal{G} = \{(\neg p \vee q, 1), (\neg q, 1), (p \vee r, 1), (\neg r, 1)\}$. Note that because MaxSAT resolution replaces clauses, each clause in the DAG has at most one child. The residual is $\mathcal{G}^\delta = \{(\neg q \vee p, 1), (\neg r \vee \neg p, 1)\}$. Since $MaxSAT(\mathcal{G}^\delta) = 0$, the refutation is full, which implies that $MaxSAT(\mathcal{G}) = 1$.*

The Soft Pigeon Hole Problem

In the (m holes) *Pigeon Hole Problem* PHP the goal is to assign $m+1$ pigeons to m holes without any pair of pigeons sharing their hole. In the usual encoding there is a boolean variable x_{ij} (with $1 \leq i \leq m+1$, and $1 \leq j \leq m$) associated to pigeon i being in hole j. There are two groups of clauses. For each pigeon i, we have the clause,

$$P_i = \{x_{i1} \vee x_{i2} \vee \ldots \vee x_{im}\}$$

indicating that the pigeon must be assigned to a hole. For each hole j we have the set of clauses,

$$H_j = \{\neg x_{ij} \vee \neg x_{i'j} \mid 1 \leq i < i' \leq m+1\}$$

indicating that the hole is occupied by at most one pigeon. Let PHP be the union of all these sets of clauses,

$$PHP = \cup_{1 \leq i \leq m+1} P_i \ \cup_{1 \leq j \leq m} H_j$$

Note that $|PHP| = O(m^3)$

In the *soft* PHP the goal is to find the assignment that falsifies the minimum number of clauses. In MaxSAT language it is encoded as,

$$PHP_{soft} = \{(C, 1) \mid C \in PHP\}$$

Although the solution to this problem is obvious ($MaxSAT(PHP_{soft}) = 1$), it is known that there is no short MaxRes refutation for it (Bonet, Levy, and Manyà 2007).

MaxRes with Extension

Here we present a new MaxSAT proof system, called **MaxResE**. The difference with respect to MaxRes is that we remove the condition of weights being strictly positive. Negative weights may appear during proofs with the application of the following new inference rule, called *extension*:

$$\frac{}{(C, -u) \quad (x \vee C, u) \quad (\neg x \vee C, u)}$$

where C is an arbitrary clause, u is an arbitrary natural number and x is an arbitrary variable not in C. Note that the rule also applies with $C = \Box$. The extension rule adds a triplet of fresh clauses that cancel each other into the formula.

A proof under the MaxResE proof system is a sequence $\mathcal{F}_0 \vdash \mathcal{F}_1 \vdash \ldots \vdash \mathcal{F}_e$ where $\mathcal{F}_0$ is the original formula and each $\mathcal{F}_i$ is obtained by applying *resolution to clauses with positive weight* or *extension* to $\mathcal{F}_{i-1}$.

Theorem 3. *(Soundness) Given a MaxResE proof $\mathcal{F}_0 \vdash \mathcal{F}_1 \vdash \ldots \vdash \mathcal{F}_e$, we have that $MaxSAT(\mathcal{F}_0) = MaxSAT(\mathcal{F}_i)$ for all $0 < i \leq e$.*

Proof. *It is known that MaxSAT resolution is sound in MaxRes. By definition, MaxResE only applies resolution to clauses with positive weight, so its application is also sound. Therefore, we only need to prove that the cost associated to an arbitrary truth assignment does not change with an extension inference step $\mathcal{F}_{i-1} \vdash \mathcal{F}_i$ where $(C, -u), (x \vee C, u), (\neg x \vee C, u)$ are the added clauses. There are three possible cases: If the truth assignment satisfies C, then the new clauses do not affect its cost; if the truth assignment does not satisfy C and satisfies x (respectively, does not satisfy x), the cost will be decreased by $-u$ because of the first clause and increased by u because of the third (respectively, the second) clause.*

Let $\mathcal{F}_0$ be a formula not containing the empty clause and such that $MaxSAT(\mathcal{F}_0) > 0$. A MaxResE *refutation* is a proof $\mathcal{F}_0 \vdash^* \{(\Box, k)\} \cup \mathcal{F}_e^\delta$ with $k > 0$ and *all weights of $\mathcal{F}_e^\delta$ being positive*.

Example 3. *Figure 1 (bottom) shows a MaxResE refutation of formula $\mathcal{G}$ in Example 2. In this particular toy example, it is longer than the alternative MaxRes refutation at the top of the figure (length five vs three). The MaxResE potential of producing shorter refutations will become apparent in the next Section. Note that the negative weight introduced by the extension is eventually cancelled.*

Next we show that in MaxResE proofs it is possible to use extensions without losing completeness. We find useful the following lemma.

Lemma 1. *There is a MaxResE proof $\mathcal{F} \vdash^* \mathcal{F} \cup \{(\Box, -w), (C, w), (\neg C, w)\}$ for any formula $\mathcal{F}$, clause C and weight $0 < w$.*

Proof. *Let $C = l_1 \vee l_2 \vee \cdots \vee l_r$. The inference is done as a sequence of r extensions,*

$$\begin{aligned}
&\mathcal{F} \vdash \mathcal{F} \cup \{(\Box, -w), (\neg l_1, w), (l_1, w)\} \vdash \\
&\vdash \mathcal{F} \cup \{(\Box, -w), (\neg l_1, w), (l_1 \vee \neg l_2, w), (l_1 \vee l_2, w)\} \vdash \\
&\ldots \\
&\vdash \mathcal{F} \cup \{(\Box, -w), (\neg l_1, w), (l_1 \vee \neg l_2, w), \ldots, \\
&(l_1 \vee l_2 \vee \ldots \vee \neg l_r, w), (l_1 \vee l_2 \vee \ldots \vee l_r, w)\} = \\
&= \mathcal{F} \cup \{(\Box, -w), (\neg C, w), (C, w)\}
\end{aligned}$$

Theorem 4. *(Completeness) Consider a formula $\mathcal{F}_0$ not containing the empty clause and such that $MaxSAT(\mathcal{F}_0) = k > 0$. Let $\mathcal{F}_0 \vdash^* \mathcal{F}_i$ be an arbitrary MaxResE proof (possibly containing extensions). There is a MaxResE full refutation $\mathcal{F}_0 \vdash^* \mathcal{F}_i \vdash^* (\Box, k) \cup \mathcal{F}_e^\delta$.*

Proof. *We only need to proof that there is a MaxResE proof $\mathcal{F}_i \vdash^* (\Box, k) \cup \mathcal{F}_e^\delta$, where $\mathcal{F}_i$ may contain negatively weighted clauses.*

Let $k = MaxSAT(\mathcal{F}_0)$ and $\mathcal{N} \subseteq \mathcal{F}_i$ the set of clauses with negative weights. If $\mathcal{N} = \emptyset$ then completeness follows trivially by completeness of MaxRes. Otherwise, for each $(C, -w)$ in $\mathcal{N}$ we add (using the previous lemma) $\{(\Box, -w), (C, w), (\neg C, w)\}$ Clearly, after simplification (clause aggregation), the resulting formula is $\mathcal{F}_j = \{(\Box, r)\} \cup \mathcal{F}'_j$ with $r = \sum_{(C,-w) \in \mathcal{N}} -w$ being a negative number and $\mathcal{F}'_j$ contains only positive weights because each $(C, -w)$ vanishes when aggregating (C, w). Since MaxResE is sound, we have that $k = r + MaxSAT(\mathcal{F}'_j)$. Because MaxRes is complete, we can derive a full refutation $\mathcal{F}'_j \vdash^* (\Box, k - r) \cup \mathcal{F}^\delta$ made exclusively of resolution steps. Joining the two proofs we obtain a full refutation $\mathcal{F}_0 \vdash^* \mathcal{F}_i \vdash^* \{(\Box, r)\} \cup \mathcal{F}'_j \vdash^* \{(\Box, r), (\Box, k - r)\} \cup \mathcal{F}_e^\delta = \{(\Box, k)\} \cup \mathcal{F}_e^\delta$.

MaxResE and the Soft Pigeon Hole Problem

In this section we show that there is a short MaxResE full refutation for the soft PHP. The refutation extracts a unit cost from PHP_{soft} producing its residual PHP_{soft}^δ. Because it does not have any obvious syntactical property characterizing its (un) satisfiability, it is not obvious if the refutation is full or not [1]. However, our experiments will show that its satisfiability can be trivially proved by any SAT or MaxSAT solver, which indicates that the refutation is a full refutation and therefore $MaxSAT(PHP_{soft}) = 1$.

Theorem 5. *Consider the encoding of the soft PHP,*

$$PHP_{soft} = \{(C, 1) \mid C \in PHP\}$$

and let m be the number of holes. There is a MaxResE refutation $PHP_{soft} \vdash^* \{(\Box, 1)\} \cup PHP_{soft}^\delta$ of length $O(m^3)$

[1]Since we know the semantics of the problem, we know that the residual has to be satisfiable, but this is not obvious from just inspecting the formula.

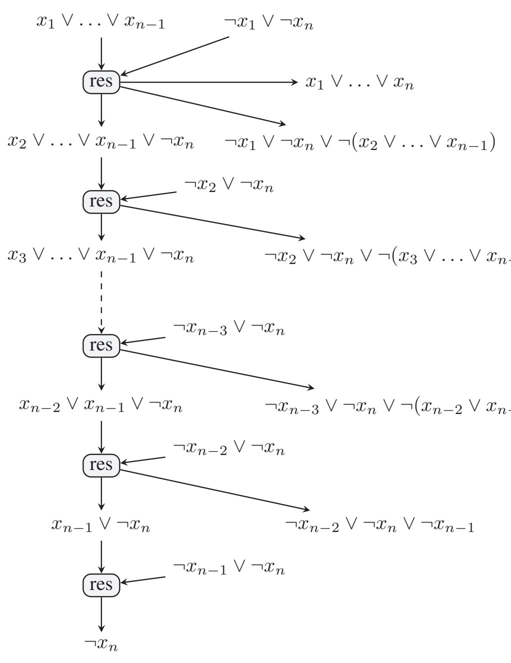

$x_1 \vee \ldots \vee x_{n-1}$
$\neg x_1 \vee \neg x_n$
res
$x_1 \vee \ldots \vee x_n$
$x_2 \vee \ldots \vee x_{n-1} \vee \neg x_n$
$\neg x_1 \vee \neg x_n \vee \neg(x_2 \vee \ldots \vee x_{n-1})$
res
$\neg x_2 \vee \neg x_n$
$x_3 \vee \ldots \vee x_{n-1} \vee \neg x_n$
$\neg x_2 \vee \neg x_n \vee \neg(x_3 \vee \ldots \vee x_{n-1})$
res
$\neg x_{n-3} \vee \neg x_n$
$x_{n-2} \vee x_{n-1} \vee \neg x_n$
$\neg x_{n-3} \vee \neg x_n \vee \neg(x_{n-2} \vee x_{n-1})$
res
$\neg x_{n-2} \vee \neg x_n$
$x_{n-1} \vee \neg x_n$
$\neg x_{n-2} \vee \neg x_n \vee \neg x_{n-1}$
res
$\neg x_{n-1} \vee \neg x_n$
$\neg x_n$

Figure 2: Proof of Lemma 2. All clauses have cost 1.

where,

$$PHP^{\delta}_{soft} = \cup_{1 \le i \le m+1} \mathcal{P}^{\delta}_i \ \cup_{1 \le j \le m} \mathcal{H}^{\delta}_j$$

$$\mathcal{P}^{\delta}_i = \{(\neg x_{ij} \vee \neg(x_{ij+1} \vee \ldots \vee x_{im}), 1) \mid 1 \le j < m\}$$

$$\mathcal{H}^{\delta}_j = \{(\neg x_{ij} \vee \neg x_{i'j} \vee \neg(x_{i+1j} \vee \ldots \vee x_{i'-1j}), 1) \mid 1 \le i < i' - 1 \le m\} \cup \{(x_{1j} \vee \ldots \vee x_{m+1j}, 1)\}$$

Note that $|PHP_{soft}|$ is $O(m^3)$ and $|PHP^{\delta}_{soft}|$ is $O(m^4)$.

In the rest of this section we give the proof of Theorem 5. Herein all clauses have weight 1, so we omit it for the sake of clarity. Also for clarity purposes, we will give proofs graphically as DAGs.

Let us first consider three useful lemmas.

Lemma 2. *Consider a MaxSAT formula* $\mathcal{M} = \{x_1 \vee \ldots \vee x_{n-1}\} \cup \{\neg x_i \vee \neg x_n \mid 1 \le i < n\}$. *There is a proof*

$$\mathcal{M} \vdash^* \{\neg x_n\} \cup \{x_1 \vee \ldots \vee x_n\} \cup \cup \{\neg x_i \vee \neg x_n \vee \neg(x_{i+1} \vee \ldots \vee x_{n-1}) \mid 1 \le i < n-1\}$$

of length $n - 1$.

Proof. *The resolution proceeds as shown in Figure 2.*

Lemma 3. *Consider a MaxSAT formula* $\mathcal{M} = \{x_1\} \cup \{\neg x_i \vee \neg x_p \mid 1 \le i < p \le n\}$. *There is a proof*

$$\mathcal{M} \vdash^* \{x_1 \vee \ldots \vee x_n\} \cup \{\neg x_i \mid 2 \le i \le n\} \cup \cup \{\neg x_i \vee \neg x_p \vee \neg(x_{i+1} \vee \ldots \vee x_{p-1}) \mid 1 \le i < p-1 < n\}$$

of length $O(n^2)$.

Proof. *The resolution proceeds as shown in Figure 3, where each circled node corresponds to an application of Lemma 2 with its incoming/outcoming clauses.*

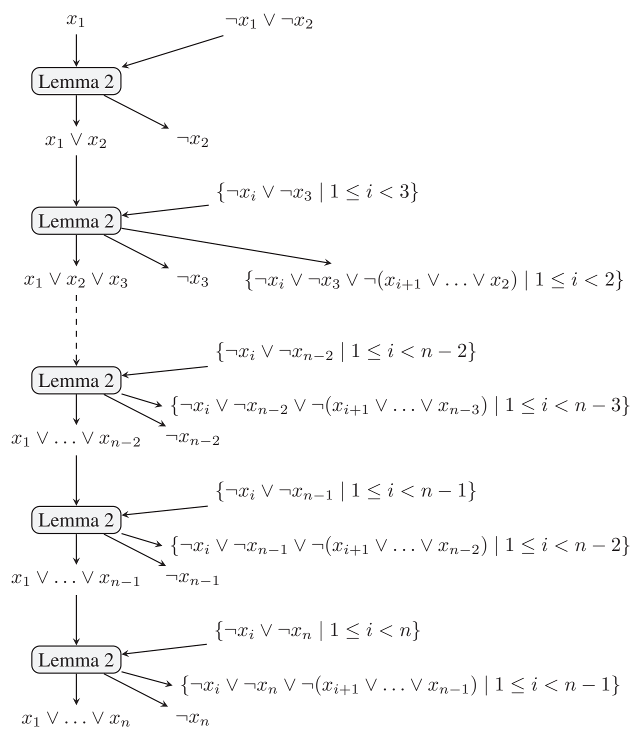

x_1
$\neg x_1 \vee \neg x_2$
Lemma 2
$x_1 \vee x_2$
$\neg x_2$
$\{\neg x_i \vee \neg x_3 \mid 1 \le i < 3\}$
Lemma 2
$x_1 \vee x_2 \vee x_3$
$\neg x_3$
$\{\neg x_i \vee \neg x_3 \vee \neg(x_{i+1} \vee \ldots \vee x_2) \mid 1 \le i < 2\}$
$\{\neg x_i \vee \neg x_{n-2} \mid 1 \le i < n-2\}$
Lemma 2
$\{\neg x_i \vee \neg x_{n-2} \vee \neg(x_{i+1} \vee \ldots \vee x_{n-3}) \mid 1 \le i < n-3\}$
$x_1 \vee \ldots \vee x_{n-2}$
$\neg x_{n-2}$
$\{\neg x_i \vee \neg x_{n-1} \mid 1 \le i < n-1\}$
Lemma 2
$\{\neg x_i \vee \neg x_{n-1} \vee \neg(x_{i+1} \vee \ldots \vee x_{n-2}) \mid 1 \le i < n-2\}$
$x_1 \vee \ldots \vee x_{n-1}$
$\neg x_{n-1}$
$\{\neg x_i \vee \neg x_n \mid 1 \le i < n\}$
Lemma 2
$\{\neg x_i \vee \neg x_n \vee \neg(x_{i+1} \vee \ldots \vee x_{n-1}) \mid 1 \le i < n-1\}$
$x_1 \vee \ldots \vee x_n$
$\neg x_n$

Figure 3: Proof of Lemma 3. All clauses have cost 1.

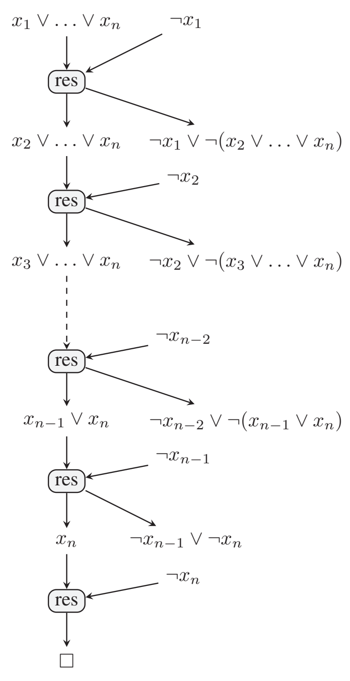

$x_1 \vee \ldots \vee x_n$
$\neg x_1$
res
$x_2 \vee \ldots \vee x_n$
$\neg x_1 \vee \neg(x_2 \vee \ldots \vee x_n)$
res
$\neg x_2$
$x_3 \vee \ldots \vee x_n$
$\neg x_2 \vee \neg(x_3 \vee \ldots \vee x_n)$
res
$\neg x_{n-2}$
$x_{n-1} \vee x_n$
$\neg x_{n-2} \vee \neg(x_{n-1} \vee x_n)$
res
$\neg x_{n-1}$
x_n
$\neg x_{n-1} \vee \neg x_n$
res
$\neg x_n$
$\square$

Figure 4: Proof of Lemma 4. All clauses have cost 1.

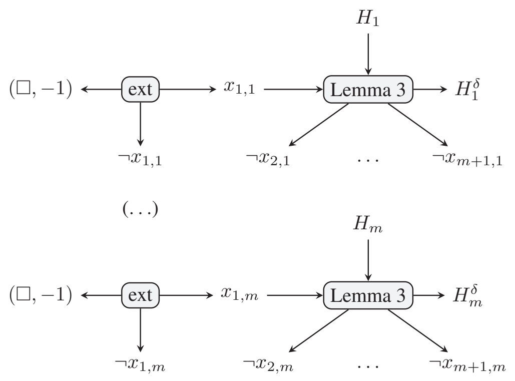


Figure 5: Proof of Theorem 5, step 1. All clauses have cost 1 unless otherwise indicated.

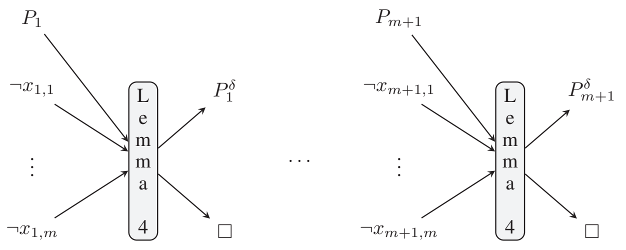


Figure 6: Proof of Theorem 5, step 2. All clauses have cost 1.

Lemma 4. *Consider a MaxSAT formula $\mathcal{M} = \{\neg x_i \mid 1 \leq i \leq n\} \cup \{x_1 \vee x_2 \vee \ldots \vee x_n\}$. There is a refutation*

$$\mathcal{M} \vdash^* \{\Box\} \cup \{\neg x_i \vee \neg(x_{i+1} \vee \ldots \vee x_n) \mid 1 \leq i < n\}$$

of length n.

Proof. *The resolution proceeds as shown in Figure 4.*

The proof of Theorem 5 only requires to extend $(\Box, -1), (x_{1j}, 1), (\neg x_{1j}, 1)$ for each hole $1 \leq j \leq m$. As a result, we obtain $(\Box, -m)$. Then, we apply the previous lemmas in the following appropriate order. First, we apply Lemma 3 for each set of clauses referring to hole j plus unit clause $(x_{1j}, 1)$. We obtain the set of unit clauses $\{(\neg x_{ij}, 1) \mid 2 \leq i \leq m+1\}$ for each hole j (Figure 5). Now we apply Lemma 4 for each set of clauses referring to pigeon i and the corresponding negative unit clauses (Figure 6). Recall that clauses $(\neg x_{1j}, 1)$ are obtained thanks to the extension rule. We obtain $m+1$ clauses $(\Box, 1)$ (i.e., $(\Box, m+1)$). After simplification (clause aggregation on $(\Box, -m)$ and $(\Box, m+1)$), we obtain $(\Box, 1)$, plus the residual $\mathcal{P}_i^\delta$ and $\mathcal{H}_j^\delta$ for each pigeon i and hole j, respectively. The length $O(m^3)$ of the refutation and the size $O(m^4)$ of the residual is obtained by a tedious but trivial calculation.

MaxResE and the Hard PHP

In this Section we show that MaxResE can also produce short refutations for the hard PHP, unlike MaxRes. Because in the hard PHP case clauses are hard (i.e., they must be satisfied) we need to allow hard clauses in MaxSAT formula. This generalization of MaxSAT is sometimes referred to as *Partial* (weighted) MaxSAT (Morgado et al. 2013).

In Partial (weighted) MaxSAT hard clauses (C, ∞) are allowed in the formulas and the goal is to find the truth assignment that satisfies all hard clauses and minimizes the sum of weights of soft clauses. If there is no such assignment we say that the formula is unsatisfiable and $MaxSAT(\mathcal{F}) = \infty$.

MaxSAT resolution can be generalized to Partial MaxSAT just by defining $\infty - w = \infty$ for all weight w. It is known that the MaxRes proof system remains sound and complete in the Partial MaxSAT language (Larrosa, Heras, and de Givry 2008).

As in MaxSAT, a MaxRes refutation in Partial MaxSAT is a proof $\mathcal{F}_0 \vdash^* \{(\Box, k)\} \cup \mathcal{F}_e^\delta$ with $k > 0$. However, because of hard clauses, k can be arbitrarily large. The following result characterizes the conditions under which a refutation actually solves the $MaxSAT(\cdot)$ problem.

Property 1. *Consider a Partial MaxSAT formula $\mathcal{F}_0$ and a refutation $\mathcal{F}_0 \vdash^* \{(\Box, k)\} \cup \mathcal{F}_e^\delta$. If k is larger than the sum of weights of all soft clauses (i.e, $k > \sum_{(C,w) \in \mathcal{F}_0 \mid w \neq \infty} w$), then $\mathcal{F}_0$ is unsatisfiable. Else if $MaxSAT(\mathcal{F}_e^\delta) = 0$, then $k = MaxSAT(\mathcal{F}_0)$.*

Proof. *To proof that a refutation where $k > \sum_{(C,w) \in \mathcal{F}_0 \mid w \neq \infty} w$ implies unsatisfiability we proceed by contradiction. Let us suppose that $\mathcal{F}_0$ is satisfiable. Therefore, $MaxSAT(\mathcal{F}_0) \leq \sum_{(C,w) \in \mathcal{F}_0 | w \neq \infty} w$. Since MaxRes is sound, $maxSAT(\mathcal{F}_0) = k + MaxSAT(\mathcal{F}_e^\delta)$. Since $MaxSAT(\mathcal{F}_e^\delta) \geq 0$, $k \leq \sum_{(C,w) \in \mathcal{F}_0 | w \neq \infty} w$, which is a contradiction. The else if part of the property follows directly from MaxRes soundness.*

Note that with the partial MaxSAT language the hard PHP problem is written as,

$$PHP_{hard} = \{(C, \infty) \mid C \in PHP\}$$

Theorem 6. *There is no short MaxRes refutation for PHP_{hard}.*

Proof. *All the clauses in PHP_{hard} are hard. MaxSAT resolution applied to hard clauses is equivalent to classical resolution. Therefore MaxRes proofs can only generate new hard clauses at each inference step and these could also be generated with standard resolution. Therefore MaxRes can only do what standard resolution can. There is no short refutation for PHP_{hard} with standard resolution (Haken 1985), therefore there is no short refutation with MaxSAT resolution.*

The extension rule can also be used in partial MaxSAT and it does not affect soundness or completeness (the proofs of Theorems 3 and 4 do not contain any assumption about clauses being soft). Therefore MaxResE is a proof system also for Partial MaxSAT. The following theorem combined with Property 1 shows that there is a short MaxResE refutation proving that PHP_{hard} is unsatisfiable.

Theorem 7. *There is a MaxResE refutation $PHP_{hard} \vdash^* \{(\Box, 1)\} \cup PHP_{hard}^\delta$ of length $O(m^3)$.*

The proof, which roughly follows the same strategy as the proof of Theorem 5, is given as a supplemental material.

Related Work

Dual Rail Encoding

In their recent work (Ignatiev, Morgado, and Marques-Silva 2017; Bonet et al. 2018) introduce the dual rail encoding which transforms a SAT formula $\mathcal{F}$ over variables $X = \{x_1, \ldots, x_s\}$ (i.e., all clauses are hard) into a MaxSAT formula $\mathcal{M}$ over variables $N = \{n_1, \ldots, n_s\}$ and $P = \{p_1, \ldots, p_s\}$. The dual encoding of clause $C \in \mathcal{F}$ is a hard clause in which each unnegated literal x_i in C is replaced by $\neg n_i$, and each negated literal $\neg x_i$ in C is replaced by $\neg p_i$. Additionally, for each variable x_i the dual encoding adds three new clauses: $(p_i, 1)$, $(n_i, 1)$ and $(\neg p_i \vee \neg n_i, \infty)$. The resulting MaxSAT formula $\mathcal{M}$ is made exclusively of horn clauses, where only unit clauses are soft.

It is shown that $\mathcal{F}$ is satisfiable iff $s = MaxSAT(\mathcal{M})$. They also show that $s \leq MaxSAT(\mathcal{M})$. Accordingly, a *dual rail MaxSAT refutation*, which is a proof of $\mathcal{F}$ unsatisfiability, is defined as $\mathcal{M} \vdash^* \{(\square, s+1)\} \cup \mathcal{M}^\delta$.

Somehow unexpectedly, applying this idea to the PHP one can proof its unsatisfiability using MaxRes in a polynomial number of steps (the refutation shares similarities with the proof of Theorem 7). From this work one can conclude that MaxSAT resolution with the dual rail encoding dominates the SAT resolution proof system. In their work it is not clear which of the dual rail ingredients (e.g. horn, unit cost soft clauses, renaming,...) if not all, are really needed for this domination.

The following Theorem shows that MaxResE is at least as powerful as the dual encoding, which seems to indicate that the true power of the dual encoding comes only from the introduction of the unary costs.

Theorem 8. *MaxResE with variable aliases can simulate the dual encoding.*

Proof. *In the proof we allow MaxResE to add for every original variable x_i a new variable y_i such that $x_i \leftrightarrow \neg y_i$. It is easy to see that the fresh variables are only syntactical sugar in the proof (there is no gain in a proof system from adding variable aliases) making it more intuitive. The proof shows that any SAT formula can be transformed to its dual rail encoding using MaxResE inference only.*

Let $\mathcal{F}$ be a SAT formula over $X = \{x_1, \ldots x_n\}$. For each variable x_i, we add hard clauses $x_i \vee y_i$ and $\neg x_i \vee \neg y_i$, where y_i is a fresh variable. The clauses only indicate that x_i and $\neg y_i$ are equivalent (i.e, no new information is added). Now, resolve each clause $x_i \vee A \in \mathcal{F}$ with $\neg x_i \vee \neg y_i$ which means that a new clause $\neg y_i \vee A$ is added to the formula. Clearly, at the end of this process we have for each original clause C, a new clause C′ where positive literals in C have been replaced by their $\neg y_i$ equivalent.

Next, for each x_i we do an extension,

$$\frac{}{(\square, -1) \quad (x_i, 1) \quad (\neg x_i, 1)}$$

Note that at the end of this process we have $(\square, -n)$. Next, we resolve each $(\neg x_i, 1)$ with $(x_i \vee y_i, \infty)$

$$\frac{(\neg x_i, 1) \quad (x_i \vee y_i, \infty)}{\begin{array}{c}(y_i, 1) \\ (\neg x_i \vee \neg y_i, 1) \quad (x_i \vee y_i, \infty)\end{array}}$$

where the last clause can be removed because it is subsumed by the already existing clause $(\neg x_i \vee \neg y_i, \infty)$. The resulting formula contains all the clauses of the dual rail encoding, so we can simulate any dual rail refutation which, by definition, ends up generating $(\square, n+1)$. The aggregation of $(\square, -n)$ and $(\square, n+1)$ produces $(\square, 1)$. Using Property 1 we know that this refutation proofs unsatisfiability.

MaxResE generalizes the dual encoding because its use is not restricted to refute SAT formulas. MaxResE can be seen as a general method for obtaining MaxSAT lower bounds or true optimums and their corresponding residuals.

Weighted CSPs

Weighted Constraint Satisfaction Problems (WCSPs) are optimization problems defined by a network of local cost functions defined over discrete variables. Thus, MaxSAT can be seen as a particular type of WCSP where the local cost functions are the clauses and variables are boolean. WCSP solvers compute lower bounds by enforcing *local consistency*. This is achieved by moving costs around the network using two equivalence preserving operations: *projection* and *extension*. WCSP projection is similar to neighborhood resolution (Larrosa and Heras 2005) and WCSP extension is similar to a restricted application of the extension rule of MaxResE that maintains the formula with positive weights. The main difference is that in the WCSPs movements are restricted to pre-defined subsets of variables (i.e, the scopes of the original cost functions), while in MaxResE we give complete freedom on the variables involved in the clauses. This freedom is needed to guarantee completeness, which is not a problem in the WCSP context where local consistency is not used as a stand-alone algorithm, but only as a heuristic.

Optimal Soft Arc Consistency OSAC (Cooper et al. 2010) introduced the idea of allowing weights to become negative during the process. As in our case, it is shown that the lower bound is valid (i.e, sound) as long as all the weights are positive at the end of the process. Interestingly, OSAC can be enforced with a linear program and the optimal lower bound is obtained (optimal with respect to the pre-defined scopes on which costs can be moved to).

Thus, OSAC is reminiscent to a guided MaxResE process restricting new clauses to pre-defined (and of bounded size) sets of variables. Interestingly, the efficiency of MaxResE on the PHP$_{soft}$ problem does not rely on the size of the clauses which is as high as the number of pigeons and holes, and therefore unbounded.

Empirical Results

To corroborate the ideas developed in the paper and demonstrate the potential of MaxResE, we conducted some experiments on hard and soft PHPs with $5 \leq m \leq 75$ using SAT solvers Glucose 3.0 (Audemard and Simon 2018) and Minisat 2.2 (Eén and Sörensson 2003), and MaxSAT solvers RC2 (Ignatiev, Morgado, and Marques-Silva 2018b) and FM (Manquinho, Marques-Silva, and Planes 2009) (extended to the case of weighted partial formulas) provided by the PySat 0.1.4 toolkit (Ignatiev, Morgado, and Marques-Silva 2018a). For both MaxSAT solvers we used Minisat

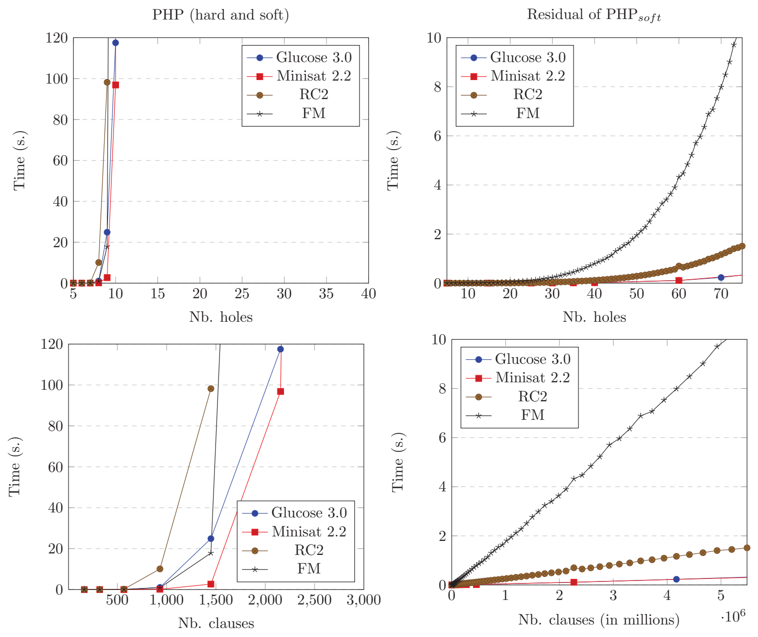


Figure 7: Solving time (in seconds) as a function of the number of holes (top row) and on the number of clauses (bottom row) on PHP$_{hard}$ and PHP$_{soft}$ (left column) and on the residual of PHP$_{soft}$ (right column).

2.2 as their underline SAT solver. All experiments were performed in Ubuntu Linux 16.04.6 LTS with 8 Gb RAM and CPU Intel i5-7400 @ 3.00 GHz.

Figure 7 shows the experimental results. The x-axis for plots on the top is the number of holes m. The top-left plot shows running time for solving PHP_{hard} and PHP_{soft} with the two SAT and MaxSAT solvers, respectively. As could be expected, only the smallest m could be solved. The reason is that pigeon hole problems are exponentially hard for resolution and SAT and MaxSAT solvers have their refutation power linked to the refutation power of resolution (Zhang and Malik 2003).

The top-right plot shows running time for solving the residual of PHP$_{soft}$ after the refutation (i.e., PHP^{δ}_{soft} in Theorem 5) with the four solvers: With the two SAT solvers all clauses are hard, so the goal is to find out whether all the clauses can be simultaneously satisfied (i.e, proof the optimality of the refutation). With the two MaxSAT solvers all clauses are soft, so the goal is to find the minimum number of clauses that must be falsified (which we know is zero). Comparing times between left and right plots one can see that the residual is much easier than the original problem. All the residuals could be solved in less than 20 seconds. The two SAT executions finish almost instantly (less than a second) with both Glucose and Minisat being equally effective. The two MaxSAT executions, although efficient in comparison to the left plot, require significantly more time, with FM being much slower than RC2.

From the plots on the top, it is not clear what is the complexity of solving the residual with the different algorithms. The plots on the bottom report the same results but now the x-axis is the actual size of the formula. From the bottom-right plot we observe that solving the residual is time linear with respect to the formula size both with SAT and MaxSAT solvers. Therefore, the highest cost of MaxSAT solvers is just constant overhead. Also note that this overhead is not that high taking into account that we are dealing with formulas with several millions of clauses.

Conclusions and Future Work

In this paper we have extended the MaxRes proof system from (Larrosa, Heras, and de Givry 2008) and (Bonet, Levy, and Manyà 2007). The new proof system, called MaxResE, is stronger since it can produce short refutations for the hard and soft pigeon hole problem. We have also shown that it generalizes the recently proposed dual rail encoding (Bonet et al. 2018) and it is closely related to optimal soft arc-consistency (Cooper et al. 2010).

In our future work we want to take practical advantage of this, mainly theoretical, result. We would like to find other problems beyond the PHP where MaxResE dominates MaxRes (and maybe Res). One serious problem of our approach is that its advantage over MaxRes seems to be hardly automatizable (the polynomial refutation of PHP is way too handcrafted), so we need to explore syntactical features (such as clause width) to effectively guide MaxResE proofs. Finally, and probably most importantly, we want to study how to incorporate the potential of the new proof system into branch-and-bound solvers.

References

Audemard, G., and Simon, L. 2018. On the glucose SAT solver. *International Journal on Artificial Intelligence Tools* 27(1):1–25.

Bonet, M. L.; Buss, S.; Ignatiev, A.; Marques-Silva, J.; and Morgado, A. 2018. Maxsat resolution with the dual rail encoding. In McIlraith, S. A., and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 6565–6572. AAAI Press.

Bonet, M. L.; Levy, J.; and Manyà, F. 2007. Resolution for max-sat. *Artif. Intell.* 171(8-9):606–618.

Cooper, M. C.; de Givry, S.; Sánchez-Fibla, M.; Schiex, T.; Zytnicki, M.; and Werner, T. 2010. Soft arc consistency revisited. *Artif. Intell.* 174(7-8):449–478.

Eén, N., and Sörensson, N. 2003. An extensible sat-solver. In Giunchiglia, E., and Tacchella, A., eds., *Theory and Applications of Satisfiability Testing, 6th International Conference, SAT 2003. Santa Margherita Ligure, Italy, May 5-8, 2003 Selected Revised Papers*, volume 2919 of *Lecture Notes in Computer Science*, 502–518. Springer.

Haken, A. 1985. The intractability of resolution. *Theoretical Computer Science* 39:297 – 308. Third Conference on Foundations of Software Technology and Theoretical Computer Science.

Ignatiev, A.; Morgado, A.; and Marques-Silva, J. 2017. On tackling the limits of resolution in SAT solving. In Gaspers, S., and Walsh, T., eds., *Theory and Applications of Satisfiability Testing - SAT 2017 - 20th International Conference, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings*, volume 10491 of *Lecture Notes in Computer Science*, 164–183. Springer.

Ignatiev, A.; Morgado, A.; and Marques-Silva, J. 2018a. Pysat: A python toolkit for prototyping with SAT oracles. In Beyersdorff, O., and Wintersteiger, C. M., eds., *Theory and Applications of Satisfiability Testing - SAT 2018 - 21st International Conference, SAT 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 9-12, 2018, Proceedings*, volume 10929 of *Lecture Notes in Computer Science*, 428–437. Springer.

Ignatiev, A.; Morgado, A.; and Marques-Silva, J. 2018b. RC2: A Python-based maxsat solver. In *MaxSAT Evaluation*, 22.

Larrosa, J., and Heras, F. 2005. Resolution in max-sat and its relation to local consistency in weighted csps. In Kaelbling, L. P., and Saffiotti, A., eds., *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*, 193–198. Professional Book Center.

Larrosa, J.; Heras, F.; and de Givry, S. 2008. A logical approach to efficient max-sat solving. *Artif. Intell.* 172(2-3):204–233.

Manquinho, V. M.; Marques-Silva, J.; and Planes, J. 2009. Algorithms for weighted boolean optimization. In *SAT*, 495–508.

Morgado, A.; Heras, F.; Liffiton, M. H.; Planes, J.; and Marques-Silva, J. 2013. Iterative and core-guided maxsat solving: A survey and assessment. *Constraints* 18(4):478–534.

Robinson, J. A. 1965. A machine-oriented logic based on the resolution principle. *J. ACM* 12(1):23–41.

Zhang, L., and Malik, S. 2003. Validating sat solvers using an independent resolution-based checker: Practical implementations and other applications. In *Proceedings of the Conference on Design, Automation and Test in Europe - Volume 1*, DATE '03, 10880–. Washington, DC, USA: IEEE Computer Society.