Available online at www.sciencedirect.com
ScienceDirect
Artificial Intelligence 171 (2007) 606–618
Artificial Intelligence
www.elsevier.com/locate/artint

Resolution for Max-SAT ☆

María Luisa Bonet [a], Jordi Levy [b,*], Felip Manyà [b]

[a] *Dept. Llenguatges i Sistemes Informàtics (LSI), Universitat Politècnica de Catalunya (UPC), Jordi Girona, 1-3, 08034 Barcelona, Spain*
[b] *Artificial Intelligence Research Institute (IIIA), Spanish Scientific Research Council (CSIC), Campus UAB, 08193 Bellaterra, Spain*

Received 1 September 2006; received in revised form 26 February 2007; accepted 1 March 2007

Available online 12 March 2007

Abstract

Max-SAT is the problem of finding an assignment minimizing the number of unsatisfied clauses in a CNF formula. We propose a resolution-like calculus for Max-SAT and prove its soundness and completeness. We also prove the completeness of some refinements of this calculus. From the completeness proof we derive an exact algorithm for Max-SAT and a time upper bound.

We also define a weighted Max-SAT resolution-like rule, and show how to adapt the soundness and completeness proofs of the Max-SAT rule to the weighted Max-SAT rule.

Finally, we give several particular Max-SAT problems that require an exponential number of steps of our Max-SAT rule to obtain the minimal number of unsatisfied clauses of the combinatorial principle. These results are based on the corresponding resolution lower bounds for those particular problems.

Keywords: Satisfiability; Resolution; Completeness; Saturation; Max-SAT; Weighted Max-SAT

1. Introduction

The Max-SAT problem for a CNF formula ϕ is the problem of finding an assignment of values to variables that minimizes the number of unsatisfied clauses in ϕ. Max-SAT is an optimization version of SAT which is NP-hard (see [25]).

Competitive exact Max-SAT solvers—as the ones developed by [2–4,17,22,23,30,32–34]—implement variants of the following branch and bound (BnB) schema: Given a CNF formula ϕ, BnB explores the search tree that represents the space of all possible assignments for ϕ in a depth-first manner. At every node, BnB compares the upper bound (*UB*), which is the best solution found so far for a complete assignment, with the lower bound (*LB*), which is the sum of the number of clauses unsatisfied by the current partial assignment plus an underestimation of the number of

☆ This research has been partially founded by the CICYT research projects iDEAS (TIN2004-04343) and Mulog (TIN2004-07933-C03-01/03). The first author also wants to thank the Isaac Newton Institute for Mathematical Sciences for hosting her while some of the ideas of this paper where thought and presented.
* Corresponding author.
E-mail addresses: bonet@lsi.upc.edu (M.L. Bonet), levy@iiia.csic.es (J. Levy), felip@iiia.csic.es (F. Manyà).
URLs: http://www.lsi.upc.edu/~bonet (M.L. Bonet), http://www.iiia.csic.es/~levy (J. Levy).

0004-3702/$ – see front matter
doi:10.1016/j.artint.2007.03.001

clauses that will become unsatisfied if the current partial assignment is completed. If $LB \geqslant UB$ the algorithm prunes the subtree below the current node and backtracks to a higher level in the search tree. If $LB < UB$, the algorithm tries to find a better solution by extending the current partial assignment by instantiating one more variable. The solution to Max-SAT is the value that UB takes after exploring the entire search tree.

The amount of inference performed by BnB at each node of the proof tree is poor compared with the inference performed in DPLL-style SAT solvers. The inference rules that one can apply in Max-SAT have to transform the current instance ϕ into another instance ϕ' in such a way that ϕ and ϕ' have the same number of unsatisfied clauses for every possible assignment; in other words, the inference rules have to be *sound*. It is not enough to preserve satisfiability as in SAT. Unfortunately, unit propagation, which is the most powerful inference technique applied in SAT, is unsound for Max-SAT,[1] and many Max-SAT solvers apply rules which are far from being as powerful as unit propagation in SAT.

A basic BnB algorithm, when branches on literal l, enforces the following inference: removes the clauses containing l and deletes the occurrences of $\bar{l}$, but the new unit clauses derived as a consequence of deleting the occurrences of $\bar{l}$ are not propagated as in unit propagation. Typically, that inference is enhanced by applying simple inference rules such as (i) the pure literal rule [13]; (ii) the dominating unit clause rule [24], (iii) the almost common clause rule [8], and (iv) the complementary unit clause rule [24]. All these rules, which are sound but not complete, have proved to be useful in a number of solvers [2,4,13,30,33].

A recent trend, that we believe will remain in future Max-SAT solvers, is to design solvers that incorporate resolution-like inference rules that can be applied efficiently at every node of the proof tree. This is the case of MaxSatz,[2] the best performing Max-SAT solver of the SAT-2006 Max-SAT Evaluation.[3] For example, one of the derived resolution rules that implements MaxSatz is the *star rule*:

$$\begin{array}{c} x \\ y \\ \bar{x} \vee \bar{y} \\ \hline\hline \Box \\ x \vee y \end{array}$$

x	y	$\bar{x} \vee \bar{y}$	$\Box$	$x \vee y$
0	0	1	0	0
0	1	1	0	1
1	0	1	0	1
1	1	0	0	1

where we have added the truth table of the rule to verify its soundness.

Max-SAT inference rules like the star rule *replace* the premises of the rule by its conclusion instead of *adding* the conclusion to the premises, which might increase the number of clauses unsatisfied by some assignment. The star rule preserves the number of unsatisfied clauses by replacing $x, y, \bar{x} \vee \bar{y}$ with $\Box, x \vee y$, where $\Box$ is the empty clause. Because these rules substitute a set of clauses by another, in some articles they are called transformation rules (see [24]) instead of resolution rules. See also [20] for other examples of rules for Max-SAT.

The main objective of this paper is to make a step forward in the study of resolution inference rules for Max-SAT by defining a sound and complete resolution rule. We want a rule such that the existing inference rules for Max-SAT either are particular cases of our rule (like the complementary unit clause rule or the almost common clause rule) or are rules that can be derived from our rule (like the star rule). We also want our rule to provide a general framework for extending our results to Weighted Max-SAT, defining complete refinements of resolution and devising faster Max-SAT solvers.

Firstly, we observe that the classical resolution rule $x \vee A, \bar{x} \vee B \vdash A \vee B$ is not sound for Max-SAT, because an assignment satisfying x and A, and falsifying B, would falsify one of the premises, but would satisfy the conclusion. So the number of unsatisfied clauses would not be preserved for every truth assignment.

[1] The set of clauses $\{a, \bar{a} \vee b, \bar{a} \vee \bar{b}, \bar{a} \vee c, \bar{a} \vee \bar{c}\}$ has a minimum of one unsatisfied clause (setting a to false). However, performing unit propagation with a leads to a non-optimal assignment falsifying at least two clauses.

[2] URL: http://web.udl.es/usuaris/m4372594/software.html.

[3] URL: http://www.iiia.csic.es/~maxsat06/.

Secondly, there is a natural extension to Max-SAT of the classical resolution rule in [21]:

$$\frac{\begin{array}{c} x \vee A \\ \bar{x} \vee B \end{array}}{\begin{array}{c} A \vee B \\ x \vee A \vee \bar{B} \\ \bar{x} \vee \bar{A} \vee B \end{array}}$$

In [21], Larrosa and Heras present this rule and ask whether it is complete for Max-SAT. However, two of the conclusions of this rule are not in clausal form, and the trivial application of distributivity results into an unsound rule:

$$\frac{\begin{array}{c} x \vee a_1 \vee \cdots \vee a_s \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \end{array}}{\begin{array}{c} a_1 \vee \cdots \vee a_s \vee b_1 \vee \cdots \vee b_t \\ x \vee a_1 \vee \cdots \vee a_s \vee \bar{b}_1 \\ \cdots \\ x \vee a_1 \vee \cdots \vee a_s \vee \bar{b}_t \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \vee \bar{a}_1 \\ \cdots \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \vee \bar{a}_s \end{array}}$$

Therefore, our first objective was to modify the previous rule to obtain a sound and *complete* resolution rule in which the conclusions are in clausal form, as well as analyzing the complexity of applying the rule and finding out if there is some complete refinement. As we show in the next sections, we achieve our objective by providing a *sound and complete resolution rule for Max-SAT* in which both premises and conclusions are in clausal form. Moreover, we describe an *exact algorithm for Max-SAT* which is derived from the completeness proof. We also *obtain an upper bound of the complexity of applying our rule and prove the completeness of the ordered resolution refinement.*

In classical resolution, different copies of a clause are eliminated leaving just one copy of each clause. In the context of the Max-SAT optimization problem, clearly this is not sound and we must keep repeated copies of a clause. This is why instead of working with sets of clauses we will work with multisets of clauses. A way to make the representation of this multisets more compact is to substitute several copies of a clause by a weighted clause, where the weight represents the number of times that the clause appears. So, our second objective was to extend our Max-SAT resolution rule to weighted clauses. As a result, we obtain a *sound and complete resolution rule for Weighted Max-SAT.*

Our third objective was to study the complexity of our calculus from the point of view of the number of steps it might need to tell us the minimal number of unsatisfied clauses. Since the Max-SAT problem is hard for the optimization problem corresponding to NP, we expect to find classes of instances that require an exponential number of steps to give the minimal number of unsatisfied clauses. As a result, we prove such lower bounds for various combinatorial principles.

Finally, in this paper we use the term of Max-SAT meaning Min-SAT. This is because, with respect to exact computations, finding an assignment that minimizes the number of unsatisfied clauses is equivalent to finding an assignment that maximizes the number of satisfied clauses. This is not necessarily the case for approximability results (see [18]).

This paper proceeds as follows. First, in Section 2 we define Max-SAT resolution and prove its soundness. Despite of the similitude of the inference rule with the classical resolution rule, it is not clear how to simulate classical inferences with the new rule. To obtain a complete strategy, we need to apply the new rule repeatedly to get a saturated set of clauses, as described in Section 3. In Section 4 we prove the completeness of the new rule, and the extension to ordered resolution. In Section 5 we deduce an exact algorithm and give a worst-case time upper bound in Section 6. Section 7 contains a rule for weighted Max-SAT and the soundness and completeness of the rule. Section 8 has the lower bound results for our Max-SAT rule. Finally, we present some concluding remarks.

2. The Max-SAT resolution rule and its soundness

In Max-SAT we need to keep repeated clauses. Therefore, we use multisets of clauses instead of just sets. For instance, the multiset $\{a, \bar{a}, \bar{a}, a \vee b, \bar{b}\}$, where a clause is repeated, has a minimum of two unsatisfied clauses.

Max-SAT resolution, like classical resolution, is based on a unique inference rule. In contrast to the resolution rule, the premises of the Max-SAT resolution rule are *removed* from the multiset after applying the rule. Moreover, apart from the classical conclusion where a variable has been cut, we also conclude some additional clauses that contain one of the premises as subclause.

Definition 1. The *Max-SAT resolution* rule is defined as follows:

$$\begin{array}{l} x \vee a_1 \vee \cdots \vee a_s \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \\ \hline\hline a_1 \vee \cdots \vee a_s \vee b_1 \vee \cdots \vee b_t \\ x \vee a_1 \vee \cdots \vee a_s \vee \bar{b}_1 \\ x \vee a_1 \vee \cdots \vee a_s \vee b_1 \vee \bar{b}_2 \\ \cdots \\ x \vee a_1 \vee \cdots \vee a_s \vee b_1 \vee \cdots \vee b_{t-1} \vee \bar{b}_t \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \vee \bar{a}_1 \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \vee a_1 \vee \bar{a}_2 \\ \cdots \\ \bar{x} \vee b_1 \vee \cdots \vee b_t \vee a_1 \vee \cdots \vee a_{s-1} \vee \bar{a}_s \end{array}$$

This inference rule is applied to multisets of clauses, and replaces the premises of the rule by its conclusions.

We say that the rule *cuts* the variable x.

The tautologies concluded by the rule are removed from the resulting multiset. Similarly, repeated literals in a clause are collapsed into one.

Definition 2. We write $\mathcal{C} \vdash \mathcal{D}$ when the multiset of clauses $\mathcal{D}$ can be obtained from the multiset $\mathcal{C}$ applying the Max-SAT resolution rule finitely many times. We write $\mathcal{C} \vdash_x \mathcal{D}$ when this sequence of applications only cuts the variable x.

The Max-SAT resolution rule may conclude more clauses than the classical resolution rule. Notice though that the number of conclusions of the rule is at most the number of literals in the premises. However, when the two premises share literals, some of the conclusions are tautologies, hence removed. In particular we have $x \vee A, \bar{x} \vee A \vdash A$. Moreover, as we will see when we study the completeness of the rule, there is no need to cut the conclusions of a rule among themselves. Finally, we will also see that the size of the worst-case proof of a set of clauses is similar to the size for classical resolution.

Notice that an instance of the rule not only depends on the two clauses of the premise and the cut variable (like in resolution), but also on the order of the literals. Notice also that, like in classical resolution, this rule concludes a new clause not containing the variable x, except when this clause is a tautology.

Example 3. The Max-SAT resolution rule removes clauses after using them in an inference step. Therefore, it could seem that it can not simulate classical resolution when a clause needs to be used more than once, like in the example of Fig. 1 (left). However, this is not the case, as it can be seen in the same figure (right). More precisely, we derive

$$a, \quad \bar{a} \vee b, \quad \bar{a} \vee c, \quad \bar{b} \vee \bar{c} \vdash \Box, \quad a \vee \bar{b} \vee \bar{c}, \quad \bar{a} \vee b \vee c$$

where any truth assignment satisfying $\{a \vee \bar{b} \vee \bar{c}, \bar{a} \vee b \vee c\}$ minimizes the number of falsified clauses in the original formula.

Notice that the structure of the classical resolution proof and the Max-SAT resolution proof is quite different. It seems difficult to adapt a classical resolution proof to get a Max-SAT resolution proof, and it is an open question if this is possible without increasing substantially the size of the proof.

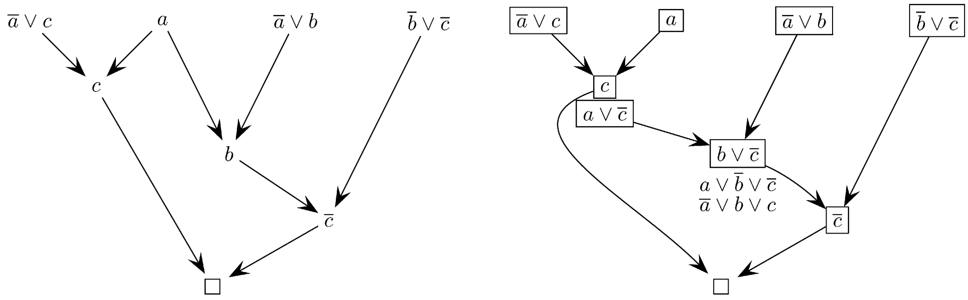

Fig. 1. An example of inference with classical resolution (left) and its equivalence with Max-SAT resolution (right). We put a box around the already used clauses.

Theorem 4 *(Soundness). The Max-SAT resolution rule is sound*; *i.e., the rule preserves the number of unsatisfied clauses for every truth assignment.*

Proof. For every assignment I, we will prove that the number of clauses that I falsifies in the premises of the inference rule is equal to the number of clauses that it falsifies in the conclusions.

Let I be any assignment. I can not falsify both premises, since it satisfies either x or $\bar{x}$.

Suppose I satisfies $x \vee a_1 \vee \cdots \vee a_s$ but not $\bar{x} \vee b_1 \vee \cdots \vee b_t$. Then I falsifies all b_j's and sets x to true. Now, suppose that I satisfies at least one literal among $\{a_1 \vee \cdots \vee a_s\}$. Say a_i is the first such literal. Then I falsifies $\bar{x} \vee b_1 \vee \cdots \vee b_t \vee a_1 \vee \cdots \vee a_{i-1} \vee \bar{a}_i$ and it satisfies all the others in the set of conclusions. Suppose now that I falsifies all a_i's. Then, it falsifies $a_1 \vee \cdots a_s \vee b_1 \vee \cdots \vee b_t$ but satisfies all the other conclusions.

If I satisfies the second premise but not the first, then by a similar argument we can show that I falsifies only one conclusion.

Finally, suppose that I satisfies both premises. Suppose that I sets x to true. Then, for some j, b_j is true and I satisfies all the conclusions since all of them have either b_j or x. The argument works similarly for I falsifying x. □

3. Saturated multisets of clauses

In this section we define *saturated* multisets of clauses. This definition is based on the classical notion of sets of clauses closed by (some restricted kind of) inference, in particular, on sets of clauses closed by cuts of some variable. In classical resolution, given a set of clauses and a variable, we can saturate the set by cutting the variable exhaustively, obtaining a superset of the given clauses. If we repeat this process for all the variables, we get a complete resolution algorithm, i.e. we obtain the empty clause whenever the original set was unsatisfiable. Our completeness proof is based on this idea. However, notice that the classical saturation of a set w.r.t. a variable is unique, whereas in Max-SAT, it is not (see Remark 8). In fact, it is not even a superset of the original set. Moreover, in general, if we saturate a set w.r.t. a variable, and then w.r.t. another variable, we obtain a set that is not saturated w.r.t. both variables.

What we will do is to first saturate with respect to a variable x. This way we create two multisets of variables. One with clauses that don't contain the variable x, and another with clauses that still contain x. We will then saturate with respect to the following variable only in the multiset of clauses that doesn't contain the first variable x. We will do the same with the rest of the variables. Also, the saturation procedure keeps a good property: given a multiset of clauses saturated w.r.t. a variable x, if there exists an assignment satisfying all the clauses not containing x, then it can be extended (by assigning x) to satisfy all the clauses (see Lemma 9).

Definition 5. A multiset of clauses $\mathcal{C}$ is said to be *saturated w.r.t. x* if for every pair of clauses $C_1 = x \vee A$ and $C_2 = \bar{x} \vee B$ of $\mathcal{C}$, there is a literal l such that l is in A and $\bar{l}$ is in B.

A multiset of clauses $\mathcal{C}'$ is a *saturation of* $\mathcal{C}$ w.r.t. x if $\mathcal{C}'$ is saturated w.r.t. x and $\mathcal{C} \vdash_x \mathcal{C}'$; i.e., $\mathcal{C}'$ can be obtained from $\mathcal{C}$ applying the inference rule cutting x finitely many times.

Trivially, by the previous definition, a multiset of clauses $\mathcal{C}$ is saturated w.r.t. x if, and only if, every possible application of the inference rule cutting x only introduces clauses containing x (since tautologies get eliminated).

We assign a function $P:\{0,1\}^n \to \{0,1\}$ to every clause, and a function $P:\{0,1\}^n \to \mathbb{N}$ to every multiset of clauses as follows.

Definition 6. For every clause $C = x_1 \vee \cdots \vee x_s \vee \bar{x}_{s+1} \vee \cdots \vee \bar{x}_{s+t}$ we define its *characteristic function* as $P_C(\vec{x}) = (1-x_1)\ldots(1-x_s)\,x_{s+1}\ldots x_{s+t}$.

For every multiset of clauses $\mathcal{C} = \{C_1,\ldots,C_m\}$, we define its *characteristic function* as $P_{\mathcal{C}} = \sum_{i=1}^m P_{C_i}(\vec{x})$.

Notice that for every assignment I, $P_{\mathcal{C}}(I)$ is the number of clauses of $\mathcal{C}$ falsified by I. Also, by the soundness of our rule, a step of the Max-SAT resolution rule replaces a multiset of clauses by another with the same characteristic function.

Before stating and proving the following lemma, let us recall the usual order relation among functions: $f \leqslant g$ if for all x, $f(x) \leqslant g(x)$, and $f < g$ if for all x, $f(x) \leqslant g(x)$ and for some x, $f(x) < g(x)$. Since the functions have finite domain and the order relation on the range is well-founded, the order relation $<$ on the functions is also well-founded.

Lemma 7. *For every multiset of clauses $\mathcal{C}$ and variable x, there exists a multiset $\mathcal{C}'$ such that $\mathcal{C}'$ is a saturation of $\mathcal{C}$ w.r.t. x. Moreover, this multiset $\mathcal{C}'$ can be computed by applying the inference rule to any pair of clauses $x \vee A$ and $\bar{x} \vee B$ with the restriction that $A \vee B$ is not a tautology, using any ordering of the literals, until we can not apply the inference rule any longer with this restriction.*

Proof. We proceed by applying nondeterministically the inference rule cutting x, until we obtain a saturated multiset. We only need to prove that this process terminates in finitely many inference steps, i.e. that there does not exist an infinite sequence $\mathcal{C} = \mathcal{C}_0 \vdash \mathcal{C}_1 \vdash \cdots$, where at every inference step we cut the variable x and none of the sets $\mathcal{C}_i$ are saturated.

At every step, we can divide $\mathcal{C}_i$ into two multisets: $\mathcal{E}_i$ with all the clauses that do not contain x, and $\mathcal{D}_i$ with the clauses that contain the variable x (in positive or negative form). When we apply the inference rule we replace two clauses of $\mathcal{D}_i$ by a multiset of clauses, where one of them, say A, does not contain x. Therefore, we obtain a distinct multiset $\mathcal{C}_{i+1} = \mathcal{D}_{i+1} \cup \mathcal{E}_{i+1}$, where $\mathcal{E}_{i+1} = \mathcal{E}_i \cup \{A\}$. Since A is not a tautology, the characteristic function P_A is not the constant zero function. Then, since $P_{\mathcal{C}_{i+1}} = P_{\mathcal{C}_i}$ and $P_{\mathcal{E}_{i+1}} = P_{\mathcal{E}_i} + P_A$, we obtain $P_{\mathcal{D}_{i+1}} = P_{\mathcal{D}_i} - P_A$ and $P_{\mathcal{D}_{i+1}} < P_{\mathcal{D}_i}$. Therefore, the characteristic function of the multiset of clauses containing x strictly decreases after every inference step. Since the order relation between characteristic functions is well-founded, this proves that we can not perform infinitely many inference steps. □

Remark 8. Although every multiset of clauses is saturable, its saturation is not unique. For instance, the multiset $\{a, \bar{a} \vee b, \bar{a} \vee c\}$ has two possibles saturations w.r.t. variable a: the multiset $\{b, \bar{b} \vee c, a \vee \bar{b} \vee \bar{c}, \bar{a} \vee b \vee c\}$ and the multiset $\{c, b \vee \bar{c}, a \vee \bar{b} \vee \bar{c}, \bar{a} \vee b \vee c\}$.

Another difference with respect to classical resolution is that we can not saturate a set of clauses simultaneously w.r.t. two variables by saturating w.r.t. one, and then w.r.t. the other. For instance, if we saturate $\{\bar{a} \vee c, a \vee b \vee c\}$ w.r.t. a, we obtain $\{b \vee c, \bar{a} \vee \bar{b} \vee c\}$. This is the only possible saturation of the original set. If now we saturate this multiset w.r.t. b, we obtain again the original set $\{\bar{a} \vee c, a \vee b \vee c\}$. Therefore, it is not possible to saturate this multiset of clauses w.r.t. a and b simultaneously.

Lemma 9. *Let $\mathcal{C}$ be a saturated multiset of clauses w.r.t. x. Let $\mathcal{C}'$ be the subset of clauses of $\mathcal{C}$ not containing x. Then, any assignment I satisfying $\mathcal{C}'$ (and not assigning x) can be extended to an assignment satisfying $\mathcal{C}$.*

Proof. We have to extend I to satisfy the whole $\mathcal{C}$. In fact we only need to set the value of x. If x has a unique polarity in $\mathcal{C} \setminus \mathcal{C}'$, then the extension is trivial ($x =$ *true* if x always occurs positively, and $x =$ *false* otherwise). If, for any clause of the form $x \vee A$ or $\bar{x} \vee A$, the assignment I already satisfies A, then any choice of the value of x will work. Otherwise, assume that there is a clause $x \vee A$ (similarly for $\bar{x} \vee A$) such that I sets A to false. We set x to true. All the clauses of the form $x \vee B$ will be satisfied. For the clauses of the form $\bar{x} \vee B$, since $\mathcal{C}$ is saturated, there exists a literal l such that $l \in A$ and $\bar{l} \in B$. This ensures that, since I falsifies A, $I(l) =$ *false* and I satisfies B. □

4. Completeness of Max-SAT resolution

Now, we prove the main result of this paper, the completeness of Max-SAT resolution. The main idea is to prove that we can get a complete algorithm by successively saturating w.r.t. all the variables. However, notice that after saturating w.r.t. x_1 and then w.r.t. x_2, we get a multiset of clauses that is not saturated w.r.t. x_1 anymore. Therefore, we will use a variant of this basic algorithm: we saturate w.r.t. x_1, then we remove all the clauses containing x_1, and saturate w.r.t. x_2, we remove all the clauses containing x_2 and saturate w.r.t. x_3, etc. Using Lemma 9, we prove that, if the original multiset of clauses was unsatisfiable, then with this process we get the empty clause. Even better, we get as many empty clauses as the minimum number of unsatisfied clauses in the original formula.

Theorem 10 *(Completeness). For any multiset of clauses $\mathcal{C}$, we have*

$$\mathcal{C} \vdash \underbrace{\Box, \ldots, \Box}_{m}, \mathcal{D}$$

where $\mathcal{D}$ is a satisfiable multiset of clauses, and m is the minimum number of unsatisfied clauses of $\mathcal{C}$.

Proof. Let $x_1, \ldots, x_n$ be any list of the variables of $\mathcal{C}$. We construct two sequences of multisets $\mathcal{C}_0, \ldots, \mathcal{C}_n$ and $\mathcal{D}_1, \ldots, \mathcal{D}_n$ such that

(i) $\mathcal{C} = \mathcal{C}_0$,
(ii) for $i = 1, \ldots, n$, $\mathcal{C}_i \cup \mathcal{D}_i$ is a saturation of $\mathcal{C}_{i-1}$ w.r.t. x_i, and
(iii) for $i = 1, \ldots, n$, $\mathcal{C}_i$ is a multiset of clauses not containing $x_1, \ldots, x_i$, and $\mathcal{D}_i$ is a multiset of clauses containing the variable x_i.

By Lemma 7, this sequences can effectively be computed: for $i = 1, \ldots, n$, we saturate $\mathcal{C}_{i-1}$ w.r.t. x_i, and then we partition the resulting multiset into a subset $\mathcal{D}_i$ containing x_i, and another $\mathcal{C}_i$ not containing this variable.

Notice that, since $\mathcal{C}_n$ does not contain any variable, it is either the empty multiset $\emptyset$, or it only contains (some) empty clauses $\{\Box, \ldots, \Box\}$.

Now we are going to prove that the multiset $\mathcal{D} = \bigcup_{i=1}^{n} \mathcal{D}_i$ is satisfiable by constructing an assignment satisfying it. For $i = 1, \ldots, n$, let $\mathcal{E}_i = \mathcal{D}_i \cup \cdots \cup \mathcal{D}_n$, and let $\mathcal{E}_{n+1} = \emptyset$. Notice that, for $i = 1, \ldots, n$,

(i) the multiset $\mathcal{E}_i$ only contains the variables $\{x_i, \ldots, x_n\}$,
(ii) $\mathcal{E}_i$ is saturated w.r.t. x_i, and
(iii) $\mathcal{E}_i$ decomposes as $\mathcal{E}_i = \mathcal{D}_i \cup \mathcal{E}_{i+1}$, where all the clauses of $\mathcal{D}_i$ contain x_i and none of $\mathcal{E}_{i+1}$ contains x_i.

Claims (i) and (iii) are trivial. For claim (ii), notice that, since $\mathcal{C}_i \cup \mathcal{D}_i$ is saturated w.r.t. x_i, the subset $\mathcal{D}_i$ is also saturated. Now, since $\mathcal{D}_{i+1} \cup \cdots \cup \mathcal{D}_n$ does not contain x_i, the set $\mathcal{E}_i$ will be saturated w.r.t. x_i.

Now, we construct a sequence of assignments $I_1, \ldots, I_{n+1}$, where I_{n+1} is the empty assignment, hence satisfies $\mathcal{E}_{n+1} = \emptyset$. Now, I_i is constructed from I_{i+1} as follows. Assume by induction hypothesis that I_{i+1} satisfies $\mathcal{E}_{i+1}$. Since $\mathcal{E}_i$ is saturated w.r.t. x_i, and decomposes into $\mathcal{D}_i$ and $\mathcal{E}_{i+1}$, by Lemma 9, we can extend I_{i+1} with an assignment for x_i to obtain I_i satisfying $\mathcal{E}_i$. Iterating, we get that I_1 satisfies $\mathcal{E}_1 = \mathcal{D} = \bigcup_{i=1}^{n} \mathcal{D}_i$.

Since the inference rule is sound (Theorem 4), and by the previous argument $\mathcal{D}$ is satisfiable, we conclude that $m = |\mathcal{C}_n|$ is the minimum number of unsatisfied clauses of $\mathcal{C}$. □

In classical resolution we can assume a given total order on the variables $x_1 > x_2 > \cdots > x_n$ and restrict inferences $x \vee A, \bar{x} \vee B \vdash A \vee B$ to satisfy that x is maximal in $x \vee A$ and in $\bar{x} \vee B$. This refinement of resolution is complete, and has some advantages: the set of possible proofs is smaller, thus its search is more efficient.

The same result holds for Max-SAT Resolution:

Corollary 11. *For any multiset of clauses $\mathcal{C}$, and for every ordering $x_1 > \cdots > x_n$ of the variables, we have*

$$\mathcal{C} \vdash_{x_1} \mathcal{C}' \vdash_{x_2} \cdots \vdash_{x_n} \underbrace{\Box, \ldots, \Box}_{m}, \mathcal{D}$$

where $\mathcal{D}$ is a satisfiable multiset of clauses, m is the minimum number of unsatisfied clauses of $\mathcal{C}$, and in every inference step the cut variable is maximal.

Proof. The proof is similar to Theorem 10. First, given the ordering $x_1 > x_2 > \cdots > x_n$, we start by computing the saturation w.r.t. x_1 and finish with x_n. Now, notice that, when we saturate $\mathcal{C}_0$ w.r.t. x_1 to obtain $\mathcal{C}_1 \cup \mathcal{D}_1$, we only cut x_1, and this is the biggest variable. Then, when we saturate $\mathcal{C}_1$ w.r.t. x_2 to obtain $\mathcal{C}_2 \cup \mathcal{D}_2$, we have to notice that the clauses of $\mathcal{C}_1$, and the clauses that we could obtain from them, do not contain x_1, and we only cut x_2 which is the biggest variable in all the premises. In general, we can see that at every inference step performed during the computation of the saturations (no matter how they are computed) we always cut a maximal variable. We only have to choose the order in which we saturate the variables coherently with the given ordering of the variables. □

5. An algorithm for Max-SAT

From the proof of Theorem 10, we can extract the following algorithm:

input : C
$C_0 := C$
for $i := 1$ **to** n
  $C := \mathit{saturation}(C_{i-1}, x_i)$
  $\langle C_i, D_i \rangle := \mathit{partition}(C, x_i)$
endfor
$m := |C_n|$
$I := \emptyset$
for $i := n$ **downto** 1
  $I := I \cup [x_i \mapsto \mathit{extension}(x_i, I, D_i)]$
output : m, I

Given an initial multiset of clauses C, this algorithm obtains the minimum number m of unsatisfied clauses and an optimal assignment I for C.

Function *saturation*(C, x) computes a saturation of C w.r.t. x. As we have already said, the saturation of a multiset is not unique, but the proof of Theorem 10 does not depends on which particular saturation we take. Therefore, this computation can be done with "don't care" non-determinism.

Function *partition*(C_i, x) computes a partition of C into the subset of clauses containing x and the subset of clauses not containing x, D_i and C_i respectively.

Function *extension*(x, I, D) computes a truth assignment for x such that, if I assigns the value true to all the clauses of D containing x, then the function returns false, if I assigns true to all the clauses of D containing $\bar{x}$, then returns true. According to Lemma 9 and the way the D_i's are computed, I evaluates to true all the clauses containing x or all the clauses containing $\bar{x}$.

The order on the saturation of the variables can also be freely chosen; i.e., the sequence $x_1, \ldots, x_n$ can be any enumeration of the variables.

6. Efficiency

In classical resolution, we know that there are formulas that require exponentially long refutations on the number of variables, and even on the size of the formula. On the other hand, no formula requires more than 2^n inference steps to be refuted, being n the number of variables. Fortunately, in many practical cases the number of resolution steps required is polynomial. Obviously, we do not have a better situation in Max-SAT resolution. Moreover, since we can have repeated clauses, and we may need to generate more than one empty clause, the number of inference steps is not only bounded by the number of variables. It also depends on the number of original clauses. Again, in many practical cases of Max-SAT resolution, the number of resolution steps is also polynomial. In contrast, bucket elimination for soft constraints [29], which is also a complete procedure for Max-SAT, *always* requires exponential time, even worse, exponential space.

The following theorem states an upper bound on the number of inference steps, using the strategy of saturating variable by variable:

Theorem 12. *For any multiset $\mathcal{C}$ of m clauses on n variables, we can deduce $\mathcal{C} \vdash \Box, \ldots, \Box, \mathcal{D}$, where $\mathcal{D}$ is satisfiable, in less than $n \cdot m \cdot 2^n$ inference steps. Moreover, the search of this proof can also be done in time $\mathcal{O}(m\, 2^n)$.*

Proof. Assign to every clause C a weight $w(C)$ equal to the number of assignments to the n variables that falsify the clause. The weight of a multiset of clauses is then the sum of the weights of its clauses. Obviously the weight of a clause is bounded by the number of possible assignments $w(C) \leqslant 2^n$, being $w(C) = 0$ true only for tautologies. Therefore, the weight of the original multiset is bounded by $m\, 2^n$.

As with the characteristic function, when $\mathcal{C} \vdash \mathcal{D}$, we have $w(\mathcal{C}) = w(\mathcal{D})$.

A similar argument to Lemma 7 can be used to prove that we can obtain a saturation $\mathcal{D}$ of any multiset $\mathcal{C}$ w.r.t. any variable x in less than $w(\mathcal{C})$ many inference steps. If we compute the weight of the clauses containing x and of those not containing x separately, we see that, in each inference step, the first weight strictly decreases while the second one increases. Therefore, the saturation w.r.t. the first variable needs no more than $m\, 2^n$ inference steps.

When we partition $\mathcal{C}$ into a subset containing x and another not containing x, both subsets will have weight smaller than $w(\mathcal{C})$, so the weight of $\mathcal{C}$ when we start the second round of saturations will also be bounded by the original weight. We can repeat the same argument for the saturation w.r.t. the n variables, and conclude that the total number of inference steps is bounded by $n\, m\, 2^n$.

The proof of completeness for ordered Max-SAT resolution, does not depend on which saturation we compute. Each inference step can be computed in time $\mathcal{O}(n)$. This gives the worst-case time upper bound. □

7. Weighted Max-SAT

In Weighted Max-SAT we use multisets of weighted clauses. A weighted clause is a pair (C, w), where C is a clause and w is a natural number meaning the penalty for falsifying the clause C. The pair (C, w) is clearly equivalent to having w copies of clause C in our multiset.

Given a truth assignment I and a multiset of weighted clauses $\mathcal{C}$, the *cost of assignment I* on $\mathcal{C}$ is the sum of the weights of the clauses falsified by I.

The *Weighted Max-SAT problem* for a multiset of weighted clauses $\mathcal{C}$ is the problem of finding an assignment to the variables of $\mathcal{C}$ that minimizes the cost of the assignment on $\mathcal{C}$.

For the Weighted Max-SAT problem a resolution-style inference rule can also be defined. The following rule is an extension of the Max-SAT resolution rule with weights like the one defined by [21]. Here we will convert the rule to clausal form, and prove its completeness.

Definition 13. The *Weighted Max-SAT resolution* rule is defined as follows:

$$
\begin{array}{l}
(x \vee a_1 \vee \cdots \vee a_s, u) \\
(\bar{x} \vee b_1 \vee \cdots \vee b_t, w) \\
\hline\hline
(a_1 \vee \cdots \vee a_s \vee b_1 \vee \cdots \vee b_t, \min(u, w)) \\
(x \vee a_1 \vee \cdots \vee a_s, u - \min(u, w)) \\
(\bar{x} \vee b_1 \vee \cdots \vee b_t, w - \min(u, w)) \\
(x \vee a_1 \vee \cdots \vee a_s \vee \bar{b}_1, \min(u, w)) \\
(x \vee a_1 \vee \cdots \vee a_s \vee b_1 \vee \bar{b}_2, \min(u, w)) \\
\cdots \\
(x \vee a_1 \vee \cdots \vee a_s \vee b_1 \vee \cdots \vee b_{t-1} \vee \bar{b}_t, \min(u, w)) \\
(\bar{x} \vee b_1 \vee \cdots \vee b_t \vee \bar{a}_1, \min(u, w)) \\
(\bar{x} \vee b_1 \vee \cdots \vee b_t \vee a_1 \vee \bar{a}_2, \min(u, w)) \\
\cdots \\
(\bar{x} \vee b_1 \vee \cdots \vee b_t \vee a_1 \vee \cdots \vee a_{s-1} \vee \bar{a}_s, \min(u, w))
\end{array}
$$

This inference rule is applied to multisets of clauses, and replaces the premises of the rule by its conclusions.

We say that the rule *cuts* the variable x.

The tautologies concluded by the rule and the clauses with weight zero are removed from the resulting multiset. Similarly, repeated literals in a clause are collapsed into one.

We also consider the following optional rule, called the *contraction rule*:

$$\frac{\begin{array}{l}(A, u)\\(A, w)\end{array}}{(A, u+w)}$$

Notice that the application of the weighted rule is clearly equivalent to the application of the unweighted rule $\min(u, w)$ many times.

Theorem 14 *(Soundness). The Weighted Max-SAT resolution rule is sound; i.e., for every truth assignment the cost of the assignment on the set of premises of the rule is equal to the cost of the assignment on the set of conclusions.*

Proof. Without loss of generality suppose that $u \leqslant w$. Then $\min(u, w) = u$, $u - \min(u, w) = 0$ and $w - \min(u, w) = w - u$. Also, $\{(x \vee A, u), (\bar{x} \vee B, w)\}$ is equivalent to $\{(x \vee A, u), (\bar{x} \vee B, u), (\bar{x} \vee B, w - u)\}$. Using this fact and the soundness of the unweighted Max-SAT rule applied u times to $\{(x \vee A, u), (\bar{x} \vee B, u)\}$ we can obtain the soundness of the weighted rule. □

Theorem 15 *(Completeness). For any multiset of clauses $\mathcal{C}$, we have*

$$\mathcal{C} \vdash (\Box, u_1), \ldots, (\Box, u_s), \mathcal{D}$$

where $\mathcal{D}$ is a satisfiable multiset of clauses, and $u_1 + \cdots + u_s$ is the minimal cost of $\mathcal{C}$.

Moreover, if we also consider the contraction rule, then we have

$$\mathcal{C} \vdash (\Box, u), \mathcal{D}$$

where u is the minimal cost.

Proof. The proof is similar to the unweighted case, generalizing the definition of characteristic function. For every weighted clause $C = (x_1 \vee \cdots \vee x_s \vee \bar{x}_{s+1} \vee \cdots \vee \bar{x}_{s+t}, u)$ we define its characteristic function as $P_C(\vec{x}) = u(1 - x_1) \cdots (1 - x_s)x_{s+1} \ldots x_{s+t}$. □

8. Hard instances for the Max-SAT resolution rule. Lower bounds

In this section we will give various examples of classes of multisets of clauses that require an exponential number of steps (respect to the number of initial clauses) to generate the minimal number of unsatisfied clauses using Max-SAT resolution.

One of these examples is the pigeon-hole principle. We will formalize it as a multiset of clauses saying that $n + 1$ pigeons cannot be placed into n holes unless a hole contains more than one pigeon. Actually, we will formalize the negation of the principle to have an unsatisfiable multiset of clauses. We will use variables $p_{i,j}$ meaning that pigeon i goes to hole j. So the multiset of clauses will be $\{p_{i,1} \vee \cdots \vee p_{i,n}\colon 1 \leqslant i \leqslant n + 1\} \cup \{\bar{p}_{i,j} \vee \bar{p}_{k,j}\colon 1 \leqslant i, k \leqslant n + 1, 1 \leqslant j \leqslant n\}$. We will call this multiset $\overline{PHP_n^{n+1}}$. From this multiset of clauses we can define the class of multisets $\{\overline{PHP_n^{n+1}}\colon n \in N\}$. For this principle, we will show that any sequence of Max-SAT resolution rule applications will need an exponential (in n) number of steps to show that one of the clauses cannot be satisfied.

To be able to show our result we will use the fact that any resolution refutation of $\overline{PHP_n^{n+1}}$ requires an exponential (in n) number of steps. Recall that resolution is a propositional proof system to show the unsatisfiability of a set of clauses based on the following inference rule.

$$\frac{\begin{array}{l}x \vee A\\\bar{x} \vee B\end{array}}{A \vee B}$$

Also we will use the fact that from a Max-SAT resolution proof we can extract a resolution refutation of the same principle. The following two theorems will be the basic ingredients of the main result of this section.

Theorem 16. *(See [10,16].) For sufficiently large n, any resolution refutation of* $\overline{PHP_n^{n+1}}$ *requires* $2^{n/20}$ *clauses (inference steps).*

Theorem 17. *Let* $\mathcal{C}$ *be an unsatisfiable multiset of clauses. Suppose*

$$\mathcal{C} \vdash \underbrace{\Box, \ldots, \Box}_{k}, \mathcal{D},$$

where $\mathcal{D}$ *is satisfiable, k the minimum number of unsatisfied clauses, and S is the number of steps in the Max-SAT refutation. Then, there is a (classical) resolution refutation of* $\mathcal{C}$ *in S steps.*

Proof. Let

$$\mathcal{C} = \mathcal{C}_0 \vdash \mathcal{C}_1 \vdash \cdots \vdash \mathcal{C}_S = \underbrace{\Box, \ldots, \Box}_{k}, \mathcal{D}$$

be the sequence of multisets that we generate in the S steps. We will define by induction on S a sequence of multisets

$$\mathcal{C} = \mathcal{D}_0 \vdash \mathcal{D}_1 \vdash \cdots \vdash \mathcal{D}_S$$

such that for every step i and for every clause C in $\mathcal{C}_i$, there is a clause C' in $\mathcal{D}_i$ such that $C' \subset C$.

Since $\mathcal{D}_0 = \mathcal{C}_0$, obviously the property holds for 0.

Assume now by the induction hypothesis that for every $j \leqslant i$ the corresponding $\mathcal{D}_i$'s are defined fulfilling the property. Let $\mathcal{C}_i = \{x \vee A, \bar{x} \vee B\} \cup \mathcal{C}'$, and $\mathcal{C}_{i+1} = \{A \vee B \ldots\} \cup \mathcal{C}'$. Let $D_1, D_2 \in \mathcal{D}_i$ such that $D_1 \subset x \vee A$ and $D_2 \subset \bar{x} \vee B$. If $x \notin D_1$ or $\bar{x} \notin D_2$ then $\mathcal{D}_{i+1} = \mathcal{D}_i$. Else, we apply the resolution rule to D_1 and D_2 cutting x to obtain a clause D that will be a subset of $A \vee B$. Then we define $\mathcal{D}_{i+1} = \mathcal{D}_i \cup \{D\}$. Notice that the other clauses of $\mathcal{C}_{i+1}$ have a subclause in $\mathcal{D}_i$ and therefore in $\mathcal{D}_{i+1}$.

Now since $\mathcal{C}_S$ has at least one $\Box$, and for every clause in $\mathcal{C}_S$ there is a clause in $\mathcal{D}_S$ that is a subclause of it, $\mathcal{D}_S$ must contain $\Box$. $\Box$

Now we can give the lower bound theorem, which is an immediate corollary of Theorem 17 together with Theorem 16.

Corollary 18. *For sufficiently large n, any Max-SAT resolution derivation of* $\overline{PHP_n^{n+1}} \vdash \Box, \mathcal{D}$*, where* $\mathcal{D}$ *is satisfiable, requires* $2^{n/20}$ *inference steps.*

There are resolution lower bounds (as in Theorem 16) for other combinatorial principles like Tseitin formulas [11, 31], Random formulas [9,11], Mutilated chessboard [1], Reflexion principle [6,7], Planar Tautologies [14], the Clique-Coloring principle [19,26] and the Weak pigeon-hole principle [27,28]. Also Theorem 17 is very general. Therefore we can obtain the same results (as in Corollary 18) for all these combinatorial principles.

9. Conclusions

In this paper we have presented several contributions to the Max-SAT problem: (i) a new sound and complete resolution rule in which both conclusions and premises are in clausal form; (ii) an original exact algorithm; (iii) a complete ordered resolution refinement; (iv) an extension of our rule which is complete for Weighted Max-SAT, and (v) examples of classes of multisets of clauses that require an exponential number of Max-SAT resolution steps (in terms of the size of the initial multiset) to obtain the minimal number of unsatisfied clauses.

One feature of our logical framework is that the inference rules implemented in modern Max-SAT solvers either are particular cases of our rule or can be obtained as derived rules. So, we believe that our framework is a good starting point for designers of future Max-SAT solvers to devise more powerful inference techniques that could be applied efficiently at every node of the proof tree.

The exact algorithm we have obtained from the completeness proof can be described as an extension to Max-SAT of the original Davis–Putnam algorithm [15]. We believe that, as it happens in SAT, an implementation of that algorithm could rarely outperform a DPLL-style Max-SAT solver. Nevertheless, it is an open question to know if there are Max-SAT instances that can be solved in polynomial time with our algorithm but require exponential time with DPLL-style solvers.

The contributions of this paper can also be applied beyond Max-SAT. As an example, we would like to point out that we have recently extended our results to many-valued CNF formulas. The complete resolution rule for Many-Valued Max-SAT we have obtained has allowed us to define derived rules that capture the most relevant soft local consistencies defined in the Weighted CSP community, as well as to provide a logical framework for Weighted CSPs. Such results were presented at the workshop Soft Constraints at CP'06, and also published in [5].

This paper is an extended version of our paper "A complete calculus for Max-SAT" presented at SAT'06 [12] containing more results. Sections 7 and 8 are new. Independently of our work, [17] have recently presented a clausal form translation of their inference rule [21] at AAAI-2006.

References

[1] M. Alekhnovich, Mutilated chessboard problem is exponentially hard for resolution, Theoretical Computer Science 310 (1–3) (2004) 513–525.

[2] T. Alsinet, F. Manyà, J. Planes, Improved branch and bound algorithms for Max-SAT, in: Proc. of the 6th Int. Conf. on the Theory and Applications of Satisfiability Testing, SAT'03, 2003.

[3] T. Alsinet, F. Manyà, J. Planes, A Max-SAT solver with lazy data structures, in: Proc. of the 9th Ibero–American Conference on Artificial Intelligence, IBERAMIA'04, Puebla, México, in: Lecture Notes Comput. Sci., vol. 3315, Springer, 2004, pp. 334–342.

[4] T. Alsinet, F. Manyà, J. Planes, Improved exact solver for weighted Max-SAT, in: Proc. of the 8th Int. Conf. on Theory and Applications of Satisfiability Testing, SAT'05, St. Andrews, Scotland, in: Lecture Notes Comput. Sci., vol. 3569, Springer, 2005, pp. 371–377.

[5] C. Ansótegui, M.L. Bonet, J. Levy, F. Manyà, The logic behind weighted CSP, in: Proc. of the 20th Int. Joint Conference on Artificial Intelligence, IJCAI'07, AAAI Press, 2007, pp. 32–37.

[6] A. Atserias, M.L. Bonet, On the automatizability of resolution and related propositional proof systems, in: Proc. of the 16th Int. Conf. on Computer Science Logic, CSL'02, Edinburgh, Scotland, in: Lecture Notes Comput. Sci., vol. 2471, Springer, 2002, pp. 569–583.

[7] A. Atserias, M.L. Bonet, On the automatizability of resolution and related propositional proof systems, Information and Computation 189 (2) (2004).

[8] N. Bansal, V. Raman, Upper bounds for MaxSat: Further improved, in: Proc. of the 10th Int. Symposium on Algorithms and Computation, ISAAC'99, Chennai, India, in: Lecture Notes Comput. Sci., vol. 1741, Springer, 1999, pp. 247–260.

[9] P. Beame, R. Karp, T. Pitassi, M. Saks, The efficiency of resolution and Davis–Putnam procedures, SIAM Journal on Computing 31 (4) (2002) 1048–1075.

[10] P. Beame, T. Pitassi, Simplified and improved resolution lower bounds, in: Proc. of the 37th Annual Symposium on Foundations of Computer Science, FOCS'96, Burlington, VT, USA, 1996, pp. 274–282.

[11] E. Ben-Sasson, A. Wigderson, Short proofs are narrow—resolution made simple, Journal of the ACM 48 (2) (2001) 149–169.

[12] M.L. Bonet, J. Levy, F. Manyà, A complete calculus for Max-SAT, in: Proc. of the 9th Int. Conf. on Theory and Applications of Satisfiability Testing, SAT'06, Seattle, WA, USA, in: Lecture Notes Comput. Sci., vol. 4121, Springer, 2006, pp. 240–251.

[13] B. Borchers, J. Furman, A two-phase exact algorithm for Max-SAT and weighted Max-SAT problems, Journal of Combinatorial Optimization 2 (1999) 299–306.

[14] S. Dantchev, S. Riis, Planar tautologies, hard for resolution, in: Proc. of the 42nd Annual Symposium on Foundations of Computer Science, FOCS'01, Los Alamitos, CA, USA, 2001, pp. 220–231.

[15] M. Davis, H. Putnam, A computing procedure for quantification theory, Journal of the ACM 7 (3) (1960) 201–215.

[16] A. Haken, The intractability of Resolution, Theoretical Computer Science 39 (2–3) (1985) 297–308.

[17] F. Heras, J. Larrosa, New inference rules for efficient Max-SAT solving, in: Proc. of the 21st National Conference on Artificial Intelligence, AAAI'06, Boston, MA, AAAI Press, 2006, pp. 68–73.

[18] S. Khanna, M. Sudan, L. Trevisan, D.P. Williamson, The approximability of constraint satisfaction problems, SIAM Journal on Computing 30 (6) (2001) 1863–1920.

[19] J. Krajícek, Interpolation theorems, lower bounds for proof systems and independence results for bounded arithmetic, Journal of Symbolic Logic 62 (2) (1997).

[20] A.S. Kulikov, Automated generation of simplification rules for SAT and MAXSAT, in: Proc. of the 8th Int. Conf. on Theory and Applications of Satisfiability Testing, SAT'05, St. Andrews, Scotland, in: Lecture Notes Comput. Sci., vol. 3569, Springer, 2005, pp. 430–436.

[21] J. Larrosa, F. Heras, Resolution in Max-SAT and its relation to local consistency in weighted CSPs, in: Proc. of the 19th Int. Joint Conference on Artificial Intelligence, IJCAI'05, Edinburgh, Scotland, 2005, pp. 193–198.

[22] C.M. Li, F. Manyà, J. Planes, Exploiting unit propagation to compute lower bounds in branch and bound Max-SAT solvers, in: Proc. of the 11th Int. Conf. on Principles and Practice of Constraint Programming, CP'05, Sitges, Spain, in: Lecture Notes Comput. Sci., vol. 3709, Springer, 2005, pp. 403–414.

[23] C.M. Li, F. Manyà, J. Planes, Detecting disjoint inconsistent subformulas for computing lower bounds for Max-SAT, in: Proc. of the 21st National Conference on Artificial Intelligence, AAAI'06, Boston, MA, AAAI Press, 2006, pp. 86–91.

[24] R. Niedermeier, P. Rossmanith, New upper bounds for maximum satisfiability, Journal of Algorithms 36 (1) (2000) 63–88.

[25] C.M. Papadimitriou, Computational Complexity, Addison-Wesley, Reading, MA, 1994.

[26] P. Pudlák, Lower bounds for resolution and cutting plane proofs and monotone computations, Journal of Symbolic Logic 62 (3) (1997).

[27] R. Raz, Resolution lower bounds for the weak pigeonhole principle, in: Proc. of the 34th Annual ACM Symposium on Theory of Computing, STOC'02, Montreal, Canada, 2002, pp. 553–562.
[28] A. Razborov, Improved resolution lower bounds for the weak pigeonhole principle, Electronic Colloquium on Computational Complexity 8 (55) (2001).
[29] E. Rollon, J. Larrosa, Bucket elimination for multiobjective optimization problems, Journal of Heuristics 12 (4–5) (2006).
[30] H. Shen, H. Zhang, Study of lower bound functions for MAX-2-SAT, in: Proc. of the 19th National Conference on Artificial Intelligence, AAAI'04, San Jose, CA, USA, AAAI Press, 2004, pp. 185–190.
[31] A. Urquhart, Hard examples for resolution, Journal of the ACM 34 (1) (1987) 209–219.
[32] Z. Xing, W. Zhang, Efficient strategies for (weighted) maximum satisfiability, in: Proc. of the 10th Int. Conf. on Principles and Practice of Constraint Programming, CP'04, Toronto, Canada, in: Lecture Notes Comput. Sci., vol. 3258, Springer, 2004, pp. 690–705.
[33] Z. Xing, W. Zhang, An efficient exact algorithm for (weighted) maximum satisfiability, Artificial Intelligence 164 (2) (2005) 47–80.
[34] H. Zhang, H. Shen, F. Manya, Exact algorithms for MAX-SAT, Electronic Notes in Theoretical Computer Science 86 (1) (2003).