A Complete Calculus for Max-SAT*

María Luisa Bonet[1], Jordi Levy[2], and Felip Manyà[2]

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

Abstract. Max-SAT is the problem of finding an assignment minimizing the number of unsatisfied clauses of a given 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.

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 counterpart of SAT and is NP-hard.

The most competitive exact Max-SAT solvers [1–3, 7, 9, 11–13] 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 clauses that will become unsatisfied if the current partial assignment is completed. If $LB \geq UB$ 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 limited compared with DPLL-style SAT solvers. Since unit propagation is unsound for Max-SAT,[1] when branching is applied on a literal l, BnB just removes

* This research has been partially founded by the CICYT research projects iDEAS (TIN2004-04343), Mulog (TIN2004-07933-C03-01/03) and SofSAT (TIC2003-00950).

[1] The multiset of clauses $\{a,\ \overline{a} \vee b,\ \overline{a} \vee \overline{b},\ \overline{a} \vee c,\ \overline{a} \vee \overline{c}\}$ has a minimum of one unsatisfied clause. However, setting a to true (by unit propagation) leads to a non-optimal assignment falsifying two clauses.

the clauses containing l and deletes the occurrences of $\overline{l}$. The new unit clauses derived as a consequence of deleting the occurrences of $\overline{l}$ are not propagated as in DPLL. To mitigate that problem some simple inference rules have been incorporated into state-of-the-art Max-SAT solvers: (i) the pure literal rule [1, 6, 11, 13, 14]; (ii) the dominating unit clause rule first proposed in [8], and applied in [2, 6, 8, 11]; (iii) the almost common clause rule, first proposed in [4] and extended to weighted Max-SAT in [2]; that rule was called neighborhood resolution in [5] and used as a preprocessing technique in [2, 6, 10]; and (iv) the complementary unit clause rule [8]. All these rules, which are sound but not complete, have proved to be useful in practice.

The main objective of this paper is to make a step forward in the study of resolution-like inference rules for Max-SAT by defining a sound and complete resolution rule. That rule should subsume the previous rules, and provide a general framework that should allow us to define complete refinements of resolution and devise faster Max-SAT solvers.

In the context of SAT, a *sound* rule has to preserve satisfiability, like resolution does. However, in Max-SAT this is not enough; rules have to preserve the number of unsatisfied clauses for every possible assignment. Therefore, the way we apply the rule is different. To obtain a sound calculus, instead of *adding* the conclusion, which would make the number of unsatisfied clauses increase, we *replace* the premises of the rule by its conclusion. Then, the resolution rule $x \vee A, \overline{x} \vee B \vdash A \vee B$ 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.

The most natural variant of a sound resolution rule for Max-SAT was defined in [5]:

$$\begin{array}{c} x \vee A \\ \overline{x} \vee B \\ \hline A \vee B \\ x \vee A \vee \overline{B} \\ \overline{x} \vee \overline{A} \vee B \end{array}$$

However, two of the conclusions of this rule are not in clausal form, and the application of distributivity:

$$\begin{array}{c} x \vee a_1 \vee \ldots \vee a_s \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \\ \hline\hline a_1 \vee \ldots \vee a_s \vee b_1 \vee \ldots \vee b_t \\ x \vee a_1 \vee \ldots \vee a_s \vee \overline{b_1} \\ \ldots \\ x \vee a_1 \vee \ldots \vee a_s \vee \overline{b_t} \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \vee \overline{a_1} \\ \ldots \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \vee \overline{a_s} \end{array}$$

results into an unsound rule. As we show in the next section, obtaining a sound rule requires a more sophisticated adaptation of the resolution rule.

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 widely to get a saturated set of clauses, as described in Section 3. In Section 4 we prove the completeness of the new rule, and in Section 5 we prove that this result extends to ordered resolution. Finally, in Section 6 we deduce an exact algorithm and give a worst-case time upper bound in Section 7.

2 The Max-SAT Resolution Rule and its Soundness

In Max-SAT we use multisets of clauses instead of just sets. For instance, the multiset $\{a, \overline{a}, \overline{a}, a \vee b, \overline{b}\}$, 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 sub-clause.

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

$$\frac{\begin{array}{l} x \vee a_1 \vee \ldots \vee a_s \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \end{array}}{\begin{array}{l} a_1 \vee \ldots \vee a_s \vee b_1 \vee \ldots \vee b_t \\ x \vee a_1 \vee \ldots \vee a_s \vee \overline{b_1} \\ x \vee a_1 \vee \ldots \vee a_s \vee b_1 \vee \overline{b_2} \\ \ldots \\ x \vee a_1 \vee \ldots \vee a_s \vee b_1 \vee \ldots \vee b_{t-1} \vee \overline{b_t} \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \vee \overline{a_1} \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \vee a_1 \vee \overline{a_2} \\ \ldots \\ \overline{x} \vee b_1 \vee \ldots \vee b_t \vee a_1 \vee \ldots \vee a_{s-1} \vee \overline{a_s} \end{array}}$$

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.*

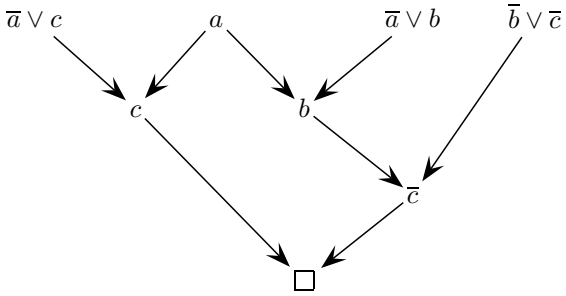
The tautologies concluded by the rule are removed from the resulting multiset. Similarly, repeated literals in a clause are also removed.

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 rule finitely many times. We write $\mathcal{C} \vdash_x \mathcal{D}$ when this sequence of applications only cuts the variable x.*

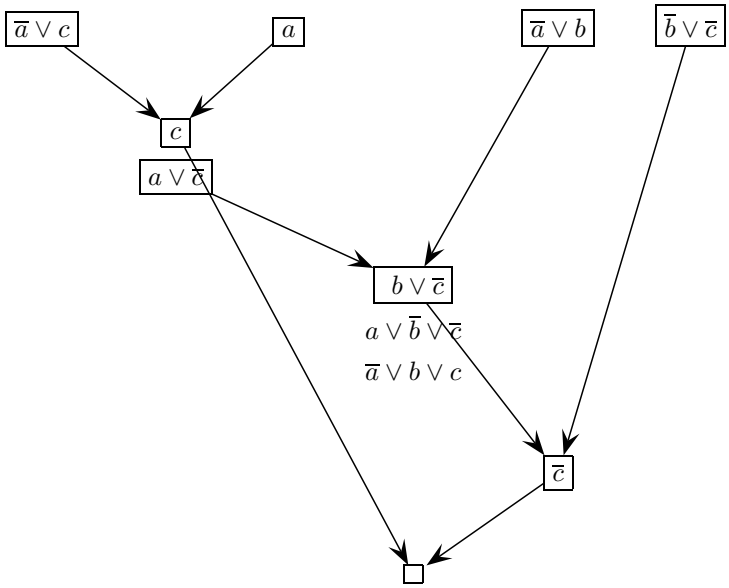
The Max-Sat resolution rule concludes many more clauses than the classical version. However, when the two premises share literals, some of the conclusions are tautologies, hence removed. In particular we have $x \vee A, \overline{x} \vee A \vdash A$. Moreover, as we will see when we study the completeness 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 the 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 1. 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:



However, this is not the case. We can derive the empty clause as follows (where already used clauses are put into boxes):



More precisely, we have derived a, $\overline{a} \vee b$, $\overline{a} \vee c$, $\overline{b} \vee \overline{c} \vdash \square$, $a \vee \overline{b} \vee \overline{c}$, $\overline{a} \vee b \vee c$, where any truth assignment satisfying $\{a \vee \overline{b} \vee \overline{c}, \overline{a} \vee b \vee c\}$ minimizes the number of falsified clauses in the original formula.

Notice that the structure of the classical 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.

Theorem 1 (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 upper clauses, since it satisfies either x or $\overline{x}$.

Suppose I satisfies $x \vee a_1 \vee \ldots \vee a_s$ but not $\overline{x} \vee b_1 \vee \ldots \vee b_t$. Then I falsifies all b_j's and sets x to true. Now, suppose that I satisfies some a_i. Say a_i is the first of such elements. Then I falsifies $\overline{x} \vee b_1 \vee \ldots \vee b_t \vee a_1 \vee \ldots \vee a_{i-1} \vee \overline{a_i}$ and it satisfies all the others in the set below. Suppose now that I falsifies all a_i's. Then, it falsifies $a_1 \vee \ldots a_s \vee b_1 \vee \ldots \vee b_t$ but satisfies all the others.

If I satisfies the second but not the first, then it is the same argument.

Finally, suppose that I satisfies both upper clauses. Suppose that I sets x to true. Then, for some j, b_j is true and I satisfies all the lower clauses since all of them have either b_j or 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 1). 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. Fortunately, we still keep 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 4).

Definition 3. *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 $C_2 = \overline{x} \vee B$, there is a literal l such that l is in A and $\overline{l}$ 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.*

The following is a trivial equivalent version of the definition.

Lemma 1. *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 4. *For every clause $C = x_1 \vee \ldots \vee x_s \vee \overline{x}_{s+1} \vee \ldots \vee \overline{x}_{s+t}$ we define its* characteristic function *as $P_C(\boldsymbol{x}) = (1 - x_1) \ldots (1 - x_s)\, x_{s+1} \ldots x_{s+t}$.*

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

Notice that the set of functions $\{0,1\}^n \to \mathbb{N}$, with the order relation: $f \leq g$ if for all x, $f(x) \leq g(x)$, defines a partial order between functions. The strict part of this relation, i.e. $f < g$ if for all x, $f(x) \leq g(x)$ and for some x, $f(x) < g(x)$, defines a strictly decreasing partial order.

Lemma 2. *Let $P_{\mathcal{C}}$ be the characteristic function of a multiset of clauses $\mathcal{C}$. For every assignment I, $P_{\mathcal{C}}(I)$ is the number of clauses of $\mathcal{C}$ falsified by I.*

The inference rule replaces a multiset of clauses by another with the same characteristic function.

Lemma 3. *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 applying the inference rule to any pair of clauses $x \vee A$ and $\overline{x} \vee B$ satisfying that $A \vee B$ is not a tautology, using any ordering of the literals, until we can not apply the inference rule any longer.

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 infinite sequences $\mathcal{C} = \mathcal{C}_0 \vdash \mathcal{C}_1 \vdash \ldots$, where at every inference 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: $\mathcal{D}_i$ with all the clauses that do not contain x, and $\mathcal{E}_i$ with the clauses that contain the variable x (in positive or negative form). When we apply the inference rule we replace two clauses of $\mathcal{E}_i$ 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 $\mathcal{D}_{i+1} = \mathcal{D}_i \cup \{A\}$. Since A is not a tautology the characteristic function P_A is not zero for some value.

Then, since $P_{\mathcal{C}_{i+1}} = P_{\mathcal{C}_i}$ and $P_{\mathcal{D}_{i+1}} = P_{\mathcal{D}_i} + P_A$, we obtain $P_{\mathcal{E}_{i+1}} = P_{\mathcal{E}_i} - P_A$. 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 strictly decreasing, this proves that we can not perform infinitely many inference steps. ■

Remark 1. Although every multiset of clauses is saturable, its saturation is not unique. For instance, the multiset $\{a,\ \overline{a} \vee b,\ \overline{a} \vee c\}$ has two possibles saturations w.r.t. a: the multiset $\{b,\ \overline{b} \vee c,\ a \vee \overline{b} \vee \overline{c},\ \overline{a} \vee b \vee c\}$ and the multiset $\{c,\ b \vee \overline{c},\ a \vee \overline{b} \vee \overline{c},\ \overline{a} \vee b \vee c\}$.

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 $\{\overline{a} \vee c,\ a \vee b \vee c\}$ w.r.t. a, we obtain $\{b \vee c,\ \overline{a} \vee \overline{b} \vee c\}$. 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 $\{\overline{a} \vee c,\ a \vee b \vee c\}$. Therefore, it is not possible to saturate this multiset of clauses w.r.t. a and b simultaneously.

Lemma 4. *Let $\mathcal{C}$ be a saturated multiset of clauses w.r.t. x. Let $\mathcal{D}$ be the subset of clauses of $\mathcal{C}$ not containing x. Then, any assignment I satisfying $\mathcal{D}$ (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 $\mathcal{C} \setminus \mathcal{D}$, 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 $\overline{x} \vee A$, 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 $\overline{x} \vee A$) 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 $\overline{x} \vee B$, since $\mathcal{C}$ is saturated, there exists a literal l such that $l \in A$ and $\overline{l} \in B$. 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. 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 4, 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 2 (Completeness). *For any multiset of clauses $\mathcal{C}$, we have*

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

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

1. $\mathcal{C} = \mathcal{C}_0$,
2. 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
3. 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 3, 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 $\{\square, \ldots, \square\}$.

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 $\mathcal{E}_i = \mathcal{D}_i \cup \ldots \cup \mathcal{D}_n$, and let $\mathcal{E}_{n+1} = \emptyset$. Notice that, for $i = 1, \ldots, n$,

1. the multiset $\mathcal{E}_i$ only contains the variables $\{x_i, \ldots, x_n\}$,
2. $\mathcal{E}_i$ is saturated w.r.t. x_i, and
3. $\mathcal{E}_i$ 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.

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 4, we can extend I_{i+1} with an assignment for x_i to obtain I_i satisfy $\mathcal{E}_i$. Iterating, we get that I_1 satisfies $\mathcal{E}_1 = \mathcal{D} = \bigcup_{i=1}^{n} \mathcal{D}_i$.

Concluding, since by the soundness Theorem 1 the inference preserves the number of falsified clauses for every assignment, $m = |\mathcal{C}_n|$ is the minimum number of unsatisfied clauses of $\mathcal{C}$. ■

5 Complete Refinements

In classical resolution we can assume a given total order on the variables $x_1 > x_2 > \ldots > x_n$ and restrict inferences $x \vee A, \overline{x} \vee B \vdash A \vee B$ to satisfy x is maximal in $x \vee A$ and in $\overline{x} \vee B$. 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:

Theorem 3 (Completeness of Ordered Max-SAT Resolution). *Max-SAT resolution with the restriction that the cut variable is maximal on the premises is complete.*

PROOF: The proof is similar to Theorem 2. First, given the ordering $x_1 > x_2 > \ldots > x_n$, 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. ■

Corollary 1. *For any multiset of clauses $\mathcal{C}$, and for every ordering $x_1 > \ldots > x_n$ of the variables, we have*

$$\mathcal{C} \vdash_{x_1} C_1 \vdash_{x_2} \cdots \vdash_{x_n} \underbrace{\square, \ldots, \square}_{m}, \mathcal{D}$$

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.

6 An Algorithm for Max-SAT

From the proof of Theorem 2, we can extract the following algorithm:

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

The function $partition(C, x)$ computes a partition of C into the subset of clauses containing x and the subset of clauses not containing x.

The 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 2 does not depends on which particular saturation we take. Therefore, this computation can be done with "don't care" nondeterminism.

The 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 $\overline{x}$, then returns true. According to Lemma 4 and the way the D_i's are computed, I evaluates to true all the clauses containing x or all the clauses containing $\overline{x}$.

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

7 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, but no formula requires more than 2^n inference steps to be refuted, being n the number of variables. We don't have a better situation in Max-SAT resolution. Moreover, since we can have repeated clauses, and need to generate more than one empty clause, the number of inference steps is not bounded by the number of variables. It also depends on the number of original clauses. The following theorem states an upper bound on the number of inference steps, using the strategy of saturating variable by variable:

Theorem 4. *For any multiset $\mathcal{C}$ of m clauses on n variables, we can deduce $\mathcal{C} \vdash \square, \ldots, \square, \mathcal{D}$, where $\mathcal{D}$ is satisfiable, in less than $n \cdot m \cdot 2^n$ inference steps.*

Moreover, the search of this proof can be also done in time $\mathcal{O}(m\, 2^n)$.

Proof: Let n be the number of variables, and m the number of original clauses. Instead of the characteristic function of a clause, we will 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 $w(C) \leq 2^n$, being $w(C) = 0$ true only for tautologies. Therefore, the weight of the original multiset is bounded by $m\, 2^n$.

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

A similar argument to Lemma 3 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 depends on which saturation we compute. Each inference step can be computed in time $\mathcal{O}(n)$. This gives the worst-case time upper bound. ■

8 Conclusions

We have defined a complete resolution rule for Max-SAT which subsumes the resolution-like rules defined so far. To the best of our knowledge, this is the first complete logical calculus defined for Max-SAT. We have also proved the completeness of the ordered resolution refinement, described an exact algorithm and computed a time upper bound.

In a longer version of this paper, we have extended the contributions to weighted Max-SAT and we have found formulas that require exponential refutations. There remain many interesting directions to follow both from a theoretical and practical perspective. For example, define further complete refinements, and use the rule to derive equivalent encodings of a given instance and study the impact on the performance of exact and non-exact Max-SAT solvers.

References

1. T. Alsinet, F. Manyà, and 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.
2. T. Alsinet, F. Manyà, and J. Planes. A Max-SAT solver with lazy data structures. In *Proc. of the 9th Ibero-American Conference on Artificial Intelligence, IBERAMIA'04*, number 3315 in LNCS, pages 334–342, Puebla, México, 2004. Springer.
3. T. Alsinet, F. Manyà, and 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*, number 3569 in LNCS, pages 371–377, St. Andrews, Scotland, 2005. Springer.
4. N. Bansal and V. Raman. Upper bounds for MaxSat: Further improved. In *Proc. of the 10th Int. Symposium on Algorithms and Computation, ISAAC'99*, number 1741 in LNCS, pages 247–260, Chennai, India, 1999. Springer.
5. J. Larrosa and 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*, pages 193–198, Edinburgh, Scotland, 2005. Morgan Kaufmann.
6. C. M. Li, F. Manyà, and 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*, number 3709 in LNCS, pages 403–414, Sitges, Spain, 2005. Springer.
7. C. M. Li, F. Manyà, and 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, USA, 2006.

8. R. Niedermeier and P. Rossmanith. New upper bounds for maximum satisfiability. *Journal of Algorithms*, 36(1):63–88, 2000.
9. H. Shen and H. Zhang. Study of lower bound functions for MAX-2-SAT. In *Proc. of the 19th National Conference on Artificial Intelligence, AAAI'04*, pages 185–190, San Jose, California, USA, 2004.
10. H. Shen and H. Zhang. Improving exact algorithms for MAX-2-SAT. *Annals of Mathematics and Artificial Intelligence*, 44(4):419–436, 2005.
11. Z. Xing and W. Zhang. Efficient strategies for (weighted) maximum satisfiability. In *Proc. of the 10th Int. Conf. on Principles and Practice of Constraint Programming, CP'04*, number 3258 in LNCS, pages 690–705, Toronto, Canada, 2004. Springer.
12. Z. Xing and W. Zhang. An efficient exact algorithm for (weighted) maximum satisfiability. *Artificial Intelligence*, 164(2):47–80, 2005.
13. H. Zhang, H. Shen, and F. Manya. Exact algorithms for Max-SAT. In *4th Int. Workshop on First-Order Theorem Proving, FTP'03*, Valencia, Spain, 2003.
14. H. Zhang, H. Shen, and F. Manya. Exact algorithms for Max-SAT. *Electronic Notes in Theoretical Computer Science*, 86(1), 2003.