A Tableau Calculus for Non-clausal Maximum Satisfiability

Chu Min Li[1], Felip Manyà[2](✉), and Joan Ramon Soler[2]

[1] MIS, Université de Picardie Jules Verne, Amiens, France
[2] Artificial Intelligence Research Institute (IIIA, CSIC), Bellaterra, Spain
felip@iiia.csic.es

Abstract. We define a non-clausal MaxSAT tableau calculus. Given a multiset of propositional formulas ϕ, we prove that the calculus is sound in the sense that if the minimum number of contradictions derived among the branches of a completed tableau for ϕ is m, then the minimum number of unsatisfied formulas in ϕ is m. We also prove that it is complete in the sense that if the minimum number of unsatisfied formulas in ϕ is m, then the minimum number of contradictions among the branches of any completed tableau for ϕ is m. Moreover, we describe how to extend the proposed calculus to deal with hard and weighted soft formulas.

1 Introduction

We can distinguish between clausal MaxSAT and non-clausal MaxSAT. Clausal MaxSAT, usually known simply as MaxSAT, is to find an assignment that minimizes the number of unsatisfied clauses in a given multiset of clauses, and non-clausal MaxSAT is to find an assignment that minimizes the number of unsatisfied formulas in a given multiset of propositional formulas that are not necessarily in clausal form.

Inference systems for SAT are unsound for MaxSAT, because they preserve satisfiability but not the minimum number of unsatisfied formulas. Thus, we need to define logical calculi meeting that condition and show that they allow one to derive as many contradictions as the minimum number of unsatisfied formulas in the input multiset.

We count with complete resolution, natural deduction and tableau calculi for clausal MaxSAT [4,7,8,18]. Restrictions of MaxSAT resolution are routinely used to propagate information in branch-and-bound MaxSAT solvers [1,2,13,16,17]; and MaxSAT resolution was used to show that there exist polynomial-size MaxSAT resolution proofs of the pigeon hole principle (PHP) if PHP is encoded as a Partial MaxSAT instance using the dual rail encoding [14]. Indeed, the combination of the dual rail encoding and MaxSAT resolution is a stronger proof system than general resolution [6].

This work was supported by Project LOGISTAR from the EU H2020 Research and Innovation Programme under Grant Agreement No. 769142, MINECO-FEDER projects RASO (TIN2015-71799-C2-1-P) and Generalitat de Catalunya SGR 172.

In this paper we address the problem of defining a complete calculus for non-clausal MaxSAT. As far as we know, non-clausal MaxSAT has not yet been considered in the community. Thus, inspired by the work on clausal MaxSAT tableaux [4,18], we define the first sound and complete non-clausal MaxSAT tableau calculus, and describe how it can be extended to deal with hard and weighted soft formulas.

The paper is mainly theoretical, but it is important to highlight that MaxSAT solving has been applied to solve problems in a range of real-world domains as diverse as bioinformatics [11,22], circuit design and debugging [23], community detection in complex networks [15], diagnosis [10], FPGA routing [25], planning [26], scheduling [5] and team formation [21], among many others.

The paper is structured as follows. Section 2 reviews how tableaux solve non-clausal SAT and clausal MaxSAT. Section 3 defines a complete non-clausal MaxSAT tableau calculus. Section 4 describes how to extend the proposed calculus to deal with hard and weighted soft formulas. Section 5 gives the conclusions.

2 Background

A propositional formula is an expression constructed from propositional variables by means of the propositional connectives $\wedge, \vee, \rightarrow$ and $\neg$ in accordance with the following rules: (i) each propositional variable is a propositional formula; and (ii) if A and B are propositional formulas, then so are $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, and $(\neg A)$. A non-clausal MaxSAT instance is a multiset of propositional formulas.[1] A truth assignment is a mapping that assigns 0 (false) or 1 (true) to each propositional variable. A propositional formula is satisfied by an assignment if it is true under the usual truth-functional interpretation of the connectives and the truth values assigned to the variables.

Given a non-clausal MaxSAT instance ϕ, non-clausal MaxSAT is the problem of finding an assignment of ϕ that minimizes the number of unsatisfied formulas.

Clauses are a particular type of propositional formulas defined as follows. A clause is a disjunction of literals, where a literal l_i is a variable x_i or its negation $\neg x_i$. A clausal MaxSAT instance is a multiset of clauses. Given a clausal MaxSAT instance ϕ, clausal MaxSAT is the problem of finding an assignment of ϕ that minimizes the number of unsatisfied clauses.

A weighted formula is a pair (A, w), where A is a propositional formula and w, its weight, is a positive number. A non-clausal weighted MaxSAT instance is a multiset of weighted formulas. Given a non-clausal weighted MaxSAT instance ϕ, non-clausal weighted MaxSAT is the problem of finding an assignment of ϕ that minimizes the sum of weights of unsatisfied formulas.

A weighted clause is a pair (C, w), where C is a clause and w, its weight, is a positive number. A clausal weighted MaxSAT instance is a multiset of weighted clauses. Given a clausal weighted MaxSAT instance ϕ, clausal weighted MaxSAT

[1] We use multisets of formulas instead of sets of formulas because duplicated formulas cannot be collapsed into one formula because then the minimum number of unsatisfied formulas might not be preserved.

is the problem of finding an assignment of ϕ that minimizes the sum of weights of unsatisfied clauses.

A non-clausal partial MaxSAT instance is a multiset of formulas in which some formulas are declared to be relaxable or soft and the rest are declared to be non-relaxable or hard. Given a non-clausal partial MaxSAT instance ϕ, non-clausal partial MaxSAT is the problem of finding an assignment of ϕ that satisfies all the hard formulas and minimizes the number of unsatisfied soft formulas.

A clausal partial MaxSAT instance is a multiset of clauses in which some clauses are declared to be relaxable or soft and the rest are declared to be non-relaxable or hard. Given a clausal partial MaxSAT instance ϕ, clausal partial MaxSAT is the problem of finding an assignment of ϕ that satisfies all the hard clauses and minimizes the number of unsatisfied soft clauses.

The weighted partial MaxSAT problem is the combination of partial MaxSAT and weighted MaxSAT. Given a multiset ϕ composed of hard formulas (clauses) and soft weighted formulas (clauses), non-clausal (clausal) weighted partial MaxSAT is the problem of finding an assignment of ϕ that satisfies all the hard formulas (clauses) and minimizes the sum of weights of unsatisfied soft formulas (clauses).

We can group all propositional formulas of the form $(A \circ B)$ and $\neg(A \circ B)$, where A and B denote propositional formulas and $\circ \in \{\vee, \wedge, \rightarrow\}$, into two categories so that the presentation and proofs are simplified. Those that act conjunctively, which are called α-formulas, and those that act disjunctively, which are called β-formulas. The different formulas in each category are displayed in Table 1. To complete a taxonomy of propositional formulas, excluding literals, we also need the propositional formulas of the form $\neg\neg A$. This notation is known as uniform notation.

Table 1. α-formulas and β-formulas.

α	α_1	α_2
$A \wedge B$	A	B
$\neg(A \vee B)$	$\neg A$	$\neg B$
$\neg(A \rightarrow B)$	A	$\neg B$

β	β_1	β_2
$A \vee B$	A	B
$\neg(A \wedge B)$	$\neg A$	$\neg B$
$A \rightarrow B$	$\neg A$	B

Note that α is logically equivalent to $\alpha_1 \wedge \alpha_2$, β is logically equivalent to $\beta_1 \vee \beta_2$ and $\neg\neg A$ is logically equivalent to A. In SAT tableaux, these equivalences are used to reduce the problem of finding a satisfying assignment of α to that of finding a satisfying assignment of both α_1 and α_2, of β to that of finding a satisfying assignment of β_1 or β_2 and of $\neg\neg A$ to that of finding a satisfying assignment of A. Thus, using the expansion rules of Table 2 we obtain a complete tableau calculus for non-clausal SAT. We introduced the contradiction rule ($\Box$-rule), where l denotes a literal, because it will be necessary in MaxSAT; in the literature, applying this rule is usually referred to as closing the branch. Note that uniform notation allows to define tableau rules for arbitrary propositional formulas in a concise way.

The tableau method is used to determine the satisfiability of a given set of propositional formulas [9,12,24]. It starts creating an initial tableau composed of a single branch that has a node for each formula in the input set of formulas. Then, it applies the expansion rules of Table 2 until a contradiction is derived in each branch (in this case, the input set of formulas is unsatisfiable) or a branch is saturated without deriving a contradiction (in this case, the input set of formula is satisfiable). A branch is saturated in a SAT tableau when all the possible applications of the expansion rules have been applied in that branch.

Table 2. Tableau expansion rules for SAT

$\dfrac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}}$	$\dfrac{\beta}{\beta_1 \mid \beta_2}$	$\dfrac{\neg\neg A}{\mathrm{A}}$	$\dfrac{\begin{array}{c}l\\ \neg l\end{array}}{\Box}$
α-rule	β-rule	$\neg$-rule	$\Box$-rule

The single tableau calculus for MaxSAT [18] defined in the literature limits the input to multisets of clauses; i.e., it is a clausal tableau calculus that cannot solve non-clausal MaxSAT. This calculus does not contain the α- and $\neg$-rule. It consists of the β- and $\Box$-rule. In fact, as all the formulas in the tableau are clauses and the formulas of type β are always disjunctions of literals of the form $l_1 \vee l_2 \vee \cdots \vee l_n$, the previous β-rule is replaced with the following n-ary β-rule:

$$\frac{l_1 \vee l_2 \vee \cdots \vee l_n}{l_1 \mid l_2 \mid \cdots \mid l_n}$$

n-ary β-rule

Note that the n-ary β-rule collapses $n-1$ applications of the β-rule over the clause $l_1 \vee l_2 \vee \cdots \vee l_n$.

In clausal MaxSAT tableaux, all the clauses in the initial tableau are declared to be active. Clauses become inactive in a branch once they have been used as premises of the β- or $\Box$-rule, and then the added conclusions become active. The application of expansion rules is restricted to active clauses. In this way, the preservation of the minimum number of unsatisfied clauses is guaranteed. Active and inactive clauses are not needed in SAT because the goal is to preserve satisfiability and the application of rules in a branch stops once a contradiction is detected. The application of rules in MaxSAT continues until no more tableau rules can be applied to the formulas in the branch, because the aim is to derive all the possible contradictions. Thus, the saturation of branches is also different in SAT and MaxSAT.

Figure 1 shows the differences between clausal SAT and clausal MaxSAT tableaux using the multiset of clauses is $\phi = \{\neg x_1, \neg x_2, \neg x_3, x_1 \vee x_2, \neg x_1 \vee x_3\}$. In the SAT case, it is enough with applying the β-rule to $x_1 \vee x_2$. Since a contradiction is detected in each branch, the input multiset of formulas is declared unsatisfiable. However, in the MaxSAT case, the β-rule must also applied to $\neg x_1 \vee x_3$ and all the possible contradictions must be detected to complete the tableau. Note that in the leftmost branch of the clausal MaxSAT tableau there is just one contradiction because we have just one occurrence of x_1, which became inactive after detecting the first contradiction.

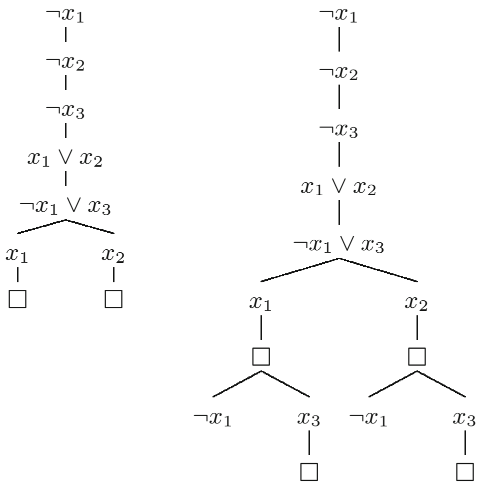


Fig. 1. Completed clausal SAT tableau (left) and completed clausal MaxSAT tableau (right) when the input multiset of clauses is $\phi = \{\neg x_1, \neg x_2, \neg x_3, x_1 \vee x_2, \neg x_1 \vee x_3\}$. The left tableau proves that ϕ is unsatisfiable and the right tableau proves that the minimum number of unsatisfied clauses in ϕ is 1.

The soundness of the previous clausal MaxSAT tableau calculus states that the β- and $\Box$-rule preserve the minimum number of unsatisfied clauses between a tableau and its extension; in particular, the β-rule preserves that number in at least one branch and does not decrease it in the rest of branches. So, once all branches have been saturated, the minimum number of contradictions derived among the branches of a completed tableau is the minimum number of unsatisfied clauses in the input multiset of clauses. The completeness states that any completed tableau for a multiset of clauses ϕ, whose minimum number of clauses that can be unsatisfied in it is k, has a branch with k contradictions and the rest of branches contain at least k contradictions [18].

If we move to deal with arbitrary propositional formulas (i.e., non-clausal MaxSAT), the first problem we encounter is that the α-rule does not preserve the minimum number of unsatisfied formulas as the β-rule does for clauses. Assume that we want to solve the non-clausal MaxSAT instance $\{x_1, x_2, \neg x_1 \wedge \neg x_2\}$, whose single optimal assignment is the one that sets x_1 and x_2 to true, and only falsifies $\neg x_1 \wedge \neg x_2$. If we apply the α-rule to $\neg x_1 \wedge \neg x_2$, we add two nodes, labelled with $\neg x_1$ and $\neg x_2$, to the initial tableau. Then, we can derive two contradictions by applying the $\Box$-rule to $\{x_1, \neg x_1\}$ and $\{x_2, \neg x_2\}$, but the minimum number of formulas unsatisfied by the optimal assignment is just one. Figure 2 displays the resulting tableau. This counterexample shows that the α-rule is unsound in MaxSAT. So, we need to define a new and sound α-rule as a first step towards getting a sound and complete non-clausal MaxSAT calculus.

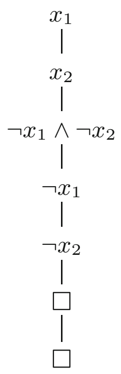


Fig. 2. Counterexample that shows that the α-rule is unsound for non-clausal MaxSAT. The input multiset is $\phi = \{x_1, x_2, \neg x_1 \wedge \neg x_2\}$.

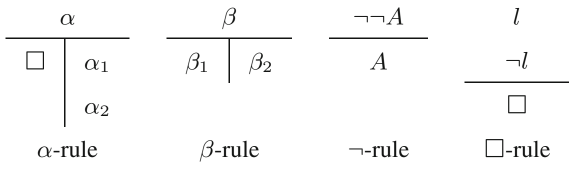
The previous example also illustrates that the standard conversion to clausal form is not valid in MaxSAT because it does not preserve the number of unsatisfied clauses. The clausal form of $\{x_1, x_2, \neg x_1 \wedge \neg x_2\}$ is $\{x_1, x_2, \neg x_1, \neg x_2\}$. However, the MaxSAT solution of $\{x_1, x_2, \neg x_1 \wedge \neg x_2\}$ is 1 and the MaxSAT solution of $\{x_1, x_2, \neg x_1, \neg x_2\}$ is 2. Thus, it is not possible to solve non-clausal MaxSAT by first translating to clausal form and then using clausal MaxSAT tableaux. We refer the reader to [19] for a recent paper on clausal form transformations for MaxSAT.

3 A Non-clausal MaxSAT Tableau Calculus

We formally define a non-clausal MaxSAT tableau calculus and prove its soundness and completeness. In the rest of the section, unless otherwise stated, when we say tableau we refer to a non-clausal MaxSAT tableau.

Definition 1. *A tableau is a tree with a finite number of branches whose nodes are labelled by either a propositional formula or a box ($\Box$). A box in a tableau denotes a contradiction. A branch is a maximal path in a tree, and we assume that branches have a finite number of nodes.*

Table 3. Tableau expansion rules for non-clausal MaxSAT



Definition 2. *Let $\phi = \{\phi_1, \ldots, \phi_m\}$ be a multiset of propositional formulas. A tableau for ϕ is constructed by a sequence of applications of the following rules:*

Initialize *A tree with a single branch with m nodes such that each node is labelled with a formula of ϕ is a tableau for ϕ. Such a tableau is called initial tableau and its formulas are declared active.*
Given a tableau T for ϕ, a branch b of T, and a node of b labelled with an active formula F,

- **α-rule** *If F is of type α, the tableau obtained by appending a new left node below b labelled with $\Box$ and a new right branch with two nodes below b labelled with α_1 and α_2 is a tableau for ϕ. Formula F becomes inactive in b and α_1 and α_2 are declared active.*
- **β-rule** *If F is of type β, the tableau obtained by appending a new left node below b labelled with β_1 and a new right node below b labelled with β_2 is a tableau for ϕ. Formula F becomes inactive in b and β_1 and β_2 are declared active.*
- **$\neg$-rule** *If F is of type $\neg\neg A$, the tableau obtained by appending a new node below b labelled with A is a tableau for ϕ. Formula $\neg\neg A$ becomes inactive in b and A is declared active.*

$\Box$-rule *Given a tableau T for ϕ, a branch b of T, and two nodes of b labelled with two active complementary literals l and $\neg l$, the tableau obtained by appending a node below b labelled with $\Box$ is a tableau for ϕ. Literals l and $\neg l$ become inactive in b.*

The expansion rules of the previous definition are summarized in Table 3. Note that all the rules preserve the number of premises falsified by an assignment I in at least one branch and do not decrease that number in the other branch (if any). In particular, in the α-rule, we have that if I falsifies α, the left branch contains one contradiction and α_1 and α_2 cannot be used to derive any other contradiction in that branch because they are not expanded; moreover, I falsifies α_1 or α_2 (or both) on the right branch. On the other hand, if I satisfies α, then I also satisfies α_1 and α_2 on the right branch.

Definition 3. *Let T be a tableau for a multiset of propositional formulas ϕ. A branch b of T is saturated when no further expansion rules can be applied on b, and T is completed when all its branches are saturated. The cost of a saturated branch is the number of boxes on the branch. The cost of a completed tableau is the minimum cost among all its branches.*

As we show below, the minimum number of formulas that can be unsatisfied in a multiset of propositional formulas ϕ is k iff the cost of a completed tableau for ϕ is k. Thus, the systematic construction of a completed tableau for ϕ provides an exact method for non-clausal MaxSAT.

Example 1. We can determine the minimum number of unsatisfied formulas in the multiset $\phi = \{x_1, x_2, \neg x_1 \wedge \neg x_2\}$ using the previous tableau calculus. Figure 3 displays how the tableau is constructed. We start by constructing the initial tableau (the leftmost tableau) and then apply the α-rule to $\neg x_1 \wedge \neg x_2$, getting as a result the second tableau in the figure. The leftmost branch is saturated and we apply the $\Box$-rule to $\{x_1, \neg x_1\}$ on the rightmost branch, getting as a result the third tableau. Finally, we apply the $\Box$-rule to $\{x_2, \neg x_2\}$ on the same branch and get the rightmost tableau in the figure. Since the minimum number of boxes among the branches of the last tableau is 1, the minimum number of formulas that can be unsatisfied in ϕ is 1.

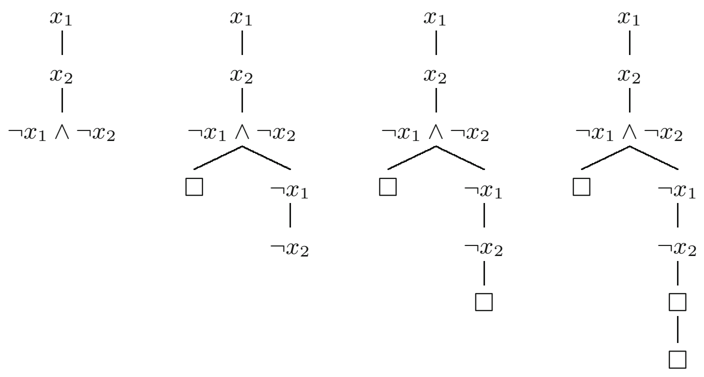


Fig. 3. A tableaux for the non-clausal MaxSAT instance $\{x_1, x_2, \neg x_1 \wedge \neg x_2\}$.

3.1 Soundness and Completeness

In this section we prove the soundness and completeness of the proposed tableau calculus for non-clausal MaxSAT. We start by proving two propositions needed later.

Proposition 1. *A tableau for a multiset of propositional formulas ϕ is completed in a finite number of steps.*

Proof. We start by creating an initial tableau and then apply rules in the newly created branches until they are saturated. The α-, β- and $\neg$-rule reduce the number of connectives. Since we began with a finite number of connectives, these rules can only be applied a finite number of times. The $\Box$-rule inactivates two literals and adds a box. Since we began with a finite number of literals and boxes cannot be premises of any expansion rule, this rule can only be applied a finite number of times. Hence, the construction of any completed tableau terminates in a finite number of steps. $\square$

Proposition 2. *An assignment I falsifies k premises of a α-, β-, $\neg$- and $\square$-rule iff assignment I falsifies k conclusions in one branch of the conclusions of the rule and at least k conclusions in the other branch (if any).*

Proof. We prove the result for each rule:

– $\square$-rule: Any assignment I always falsifies one premise and satisfies the other. Since the single conclusion is a box and denotes a contradiction, I falsifies the same number of formulas in the premises and the conclusion.
– α-rule: If I falsifies the premise of the rule, then I falsifies at least one conclusion in each branch. The left conclusion is a box and is falsified by any assignment, and I falsifies α_1 or α_2 (o both) of the right conclusion. On the other direction, if I falsifies at least one conclusion in each branch, then I falsifies α_1 or α_2 (o both) and therefore I falsifies the premise $\alpha_1 \wedge \alpha_2$.
– β-rule: If I falsifies the premise of the rule, then I falsifies β_1 and β_2, and so the left (β_1) and right (β_2) conclusions are falsified by I. On the other direction, if I falsifies both conclusions, then I falsifies $\beta_1 \vee \beta_2$.
– The $\neg$-rule: Since any assignment I that falsifies $\neg\neg A$ also falsifies A, and vice versa, I falsifies the premise iff I falsifies the conclusion.

$\square$

Theorem 1. Soundness and completeness. *The cost of a completed tableau for a multiset of formulas ϕ is k iff the minimum number of unsatisfied formulas in ϕ is k.*

Proof. (*Soundness:*) T was obtained by creating a sequence of tableaux $T_0, \ldots, T_n$ ($n \geq 0$) such that T_0 is an initial tableau for ϕ, $T_n = T$, and T_i was obtained by a single application of the α-, β-, $\neg$- or $\square$-rule on an branch of T_{i-1} for $i = 1, \ldots, n$. By Proposition 1, we know that such a sequence is finite. Since T has cost m, T_n contains one branch b with exactly m boxes and the rest of branches contain at least m boxes. Moreover, the active formulas in the branches of T_n are non-complementary literals; otherwise we could yet apply expansion rules and T_n could not be completed. The assignment that sets to true each active literal in b, only falsifies the m boxes and there cannot be any assignment satisfying less than m formulas in a branch of T_n because each branch contains at least m boxes. Therefore, the minimum number of active formulas than can be unsatisfied among the branches of T_n is m.

Proposition 2 guarantees that the minimum number of unsatisfied active formulas is preserved in the sequence of tableaux $T_0, \ldots, T_n$. Thus, the minimum number of unsatisfied active formulas in T_0 is also m. Since T_0 is formed by a single branch that only contains the formulas in ϕ and all these formulas are active, the minimum number of formulas that can be unsatisfied in ϕ is m.

(*Completeness:*) Assume that there is a completed tableau T for ϕ that does not have cost m. We distinguish two cases:

(i) T has a branch b of cost k, where $k < m$. Then, T has a branch with k boxes and a satisfiable multiset of non-complementary literals because T is completed. This implies that the minimum number of unsatisfied active formulas among the branches of T is at most k. By Proposition 2, this also holds for T_0, but this is in contradiction with m being the minimum number of unsatisfied formulas in ϕ because $k < m$. Thus, any branch of T has at least cost m.

(ii) T has no branch of cost m. This is in contradiction with m being the minimum number of unsatisfied formulas in ϕ. Since the tableau rules preserve the minimum number of unsatisfied formulas and the branches of any completed tableau only contain active formulas that are boxes or non-complementary literals, T must have a saturated branch with m boxes. Thus, T has a branch of cost m.

Hence, each completed tableau T for a multiset of formulas ϕ has cost m if the minimum number of formulas that can be unsatisfied in ϕ is m. $\square$

4 Extension to Hard and Weighted Formulas

We presented the tableau calculus for non-clausal unweighted MaxSAT (i.e,; non-clausal MaxSAT) for ease of presentation but tableaux can be extended to deal with hard and soft formulas, and soft formulas can be weighted as well.

In the case of non-clausal partial MaxSAT, there are three basic observations:

- The hard literals of the initial tableau, as well as any other literal derived by the application of an expansion rule to an input hard formula or a subformula derived from a hard formula, remain always active. In the rest of the section, we will refer to such literals as hard literals and to the subformulas derived from a hard formula as hard subformulas.
- If the $\square$-rule is applied to two hard literals, then the current branch is pruned. This means that we have found a contradiction among hard clauses. This corresponds to an unfeasible solution.
- When the premise of the α-rule is a hard formula or subformula, the α-rule of Table 2 can be used instead of the α-rule of Table 3. The calculus remains sound and complete but branching is reduced. This is so because hard formulas must be satisfied by any optimal assignment.

Example 2. Let $\phi = \mathcal{H} \cup \mathcal{S}$ be a non-clausal partial MaxSAT instance, where $\mathcal{H}$ is the multiset of hard formulas and $\mathcal{S}$ is the multiset of soft formulas. Given the multiset of propositional formulas $\{x_1 \land x_2 \land x_3, \neg x_1, \neg x_2, \neg x_3\}$, we analyze the different tableaux obtained when we vary the formulas declared as hard and soft.

The first tableau of Fig. 4 displays a completed tableau when all the formulas are soft; in this case $\phi = \mathcal{H} \cup \mathcal{S} = \emptyset \cup \{x_1 \land x_2 \land x_3, \neg x_1, \neg x_2, \neg x_3\}$.

The second tableau displays a completed tableau when $x_1 \wedge x_2 \wedge x_3$ is hard and the rest of formulas are soft; in this case $\phi = \mathcal{H} \cup \mathcal{S} = \{x_1 \wedge x_2 \wedge x_3\} \cup \{\neg x_1, \neg x_2, \neg x_3\}$. Notice that the input hard formulas and derived hard subformulas are in bold. We applied the α-rule of Table 2 because the premise is hard.

The third tableau displays a completed tableau when $\neg x_1$, $\neg x_2$ and $\neg x_3$ are hard, and $x_1 \wedge x_2 \wedge x_3$ is soft; in this case $\phi = \mathcal{H} \cup \mathcal{S} = \{\neg x_1, \neg x_2, \neg x_3\} \cup \{x_1 \wedge x_2 \wedge x_3\}$. We applied the α-rule of Table 3 because the premise is soft.

The fourth tableau displays a completed tableau when $x_1 \wedge x_2 \wedge x_3$ and $\neg x_1$ are hard, and $\neg x_2$ and $\neg x_3$ are soft; in this case $\phi = \mathcal{H} \cup \mathcal{S} = \{x_1 \wedge x_2 \wedge x_3, \neg x_1\} \cup \{\neg x_2, \neg x_3\}$. Notice that the single branch of the tableau is pruned as soon as the $\Box$-rule has two hard premises ($\neg x_1$ and x_1). We use a filled box to denote that there is no feasible solution.

In the first case, the minimum number of unsatisfied soft formulas is 1. In the second case, the minimum number of unsatisfied soft formulas among the assignments that satisfy the hard formulas is 3. In the third case, the minimum number of unsatisfied soft formulas among the assignments that satisfy the hard formulas is 1. In the fourth case, there is no optimal solution because the subset of hard formulas is unsatisfiable.

Table 4 displays the expansion rules for weighted formulas. The α-, β- and $\neg$-rule have just one premise and the weight associated to the premise is transferred to the conclusions. The $\Box$-rule has two premises and so the contradiction takes as weight the minimum of the weights associated to the premises

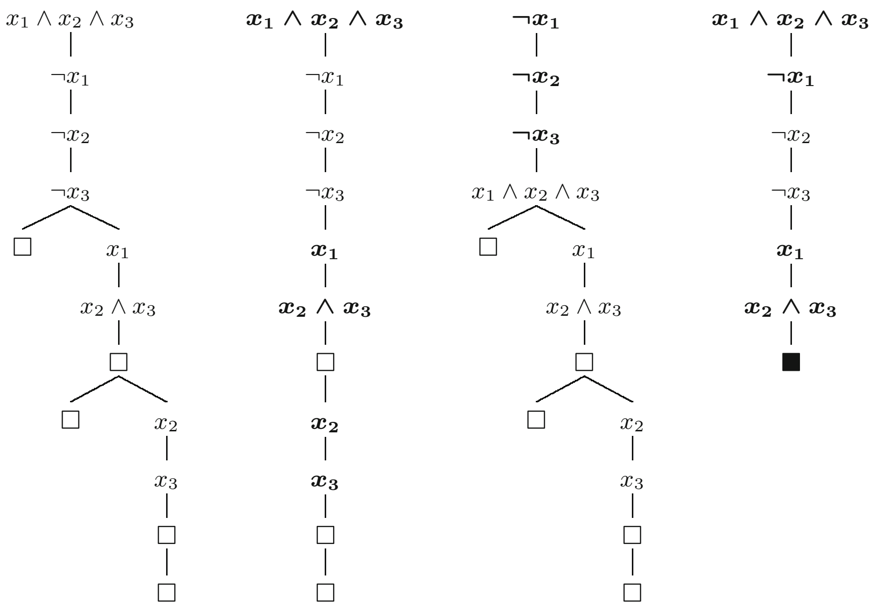


Fig. 4. Examples of non-clausal partial MaxSAT tableaux. Input hard formulas and derived hard subformulas are in bold.

Table 4. Tableau expansion rules for non-clausal weighted MaxSAT

$$
\begin{array}{cccc}
\dfrac{(\alpha, w)}{\begin{array}{c|c} (\square, w) & \begin{array}{c}(\alpha_1, w)\\ (\alpha_2, w)\end{array} \end{array}}
&
\dfrac{(\beta, w)}{\begin{array}{c|c} (\beta_1, w) & (\beta_2, w) \end{array}}
&
\dfrac{(\neg\neg A, w)}{(A, w)}
&
\dfrac{\begin{array}{c}(l, w_1)\\ (\neg l, w_2)\end{array}}{\begin{array}{c}(\square, \min(w_1, w_2))\\ (l, w_1 - \min(w_1, w_2))\\ (\neg l, w_2 - \min(w_1, w_2))\end{array}}
\\
\alpha\text{-rule} & \beta\text{-rule} & \neg\text{-rule} & \square\text{-rule}
\end{array}
$$

$(\min(w_1, w_2))$. If the premises have different weights, the remaining weight in the premise with the greatest weight can be used to detect further contradictions. The compensation weight of the other premise is 0, and formulas with weight 0 are removed. In the weighted case, when a branch has repeated occurrences of a formula A, say $(A, w_1), \ldots, (A, w_s)$, such occurrences can be replaced with the single formula $(A, w_1 + \cdots + w_s)$. Moreover, the cost of a saturated weighted branch is the sum of weights of the boxes that appear in the branch, and the cost of a completed weighted tableau is the minimum cost among all its branches.

The expansion rules of Table 4 provide a sound and complete calculus for non-clausal weighted MaxSAT. The correctness of such rules follows from the correctness of the unweighted tableau rules and the fact that having a weighted formula (A, w) is equivalent to having w copies of the unweighted formula A.

Example 3. Let $\phi = \{(\neg x_1 \rightarrow x_2, 3), (x_1 \wedge x_3, 2), (\neg x_1, 5), (\neg x_1, 5), (\neg x_3, 2)\}$ be a non-clausal weighted MaxSAT instance. Figure 5 displays a completed tableau for ϕ. This tableau has been obtained by applying the expansion rules of Table 4. The costs of the branches, from left to right, are 5, 7, 5 and 7. So, the minimum sum of weights of unsatisfied formulas is 5.

Finally, we show how to solve non-clausal weighted partial MaxSAT instances with tableaux. The first observation is that hard formulas can be considered as weighted formulas with infinity weight, and this observation is important to understand the $\square$-rule in weighted partial MaxSAT. Notice that the $\square$-rule is the only rule with two premises; in the rest os cases, if the premise is hard, we proceed as in partial MaxSAT, and if it is soft, we proceed as in weighted MaxSAT. If the two premises of the $\square$-rule are hard, then the branch is pruned because we are in front of an unfeasible solution. If the two premises are soft, then the $\square$-rule of Table 4 is applied. If there is a hard premise l and a soft premise $(\neg l, w)$, then $(\square, w)$ is derived, $(\neg l, w)$ becomes inactive and l remains active.

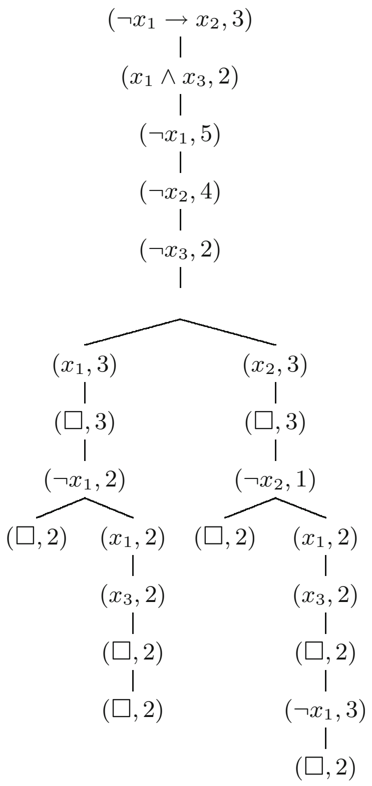


Fig. 5. Examples of non-clausal weighted MaxSAT tableaux.

Example 4. Let $\phi = \{(x_1 \wedge x_3, (\neg x_1 \rightarrow x_2, 3), (\neg x_1, 5), (\neg x_2, 1), (\neg x_3, 2)\}$ be a non-clausal weighted partial MaxSAT instance, where the first formula is hard and the rest of formulas are soft. Figure 6 displays a completed tableau for ϕ. This tableau has been obtained by applying the expansion rules for non-clausal weighted partial MaxSAT explained above. The cost of the left branch is 10 and the cost of the right branch is 8. Thus, the minimum sum of weights of unsatisfied soft formulas among the assignments that satisfy the hard formula is 8.

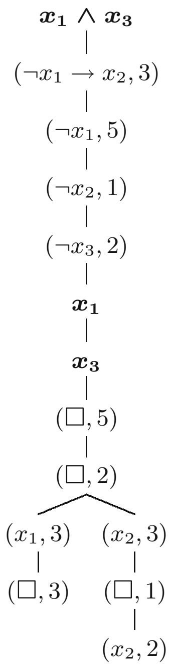


Fig. 6. Example of non-clausal weighted partial MaxSAT tableau. Input hard formulas and derived hard subformulas are in bold.

5 Conclusions

The main contributions of this paper are a non-clausal MaxSAT tableau calculus, the corresponding proofs of soundness and completeness, and its extension to deal with hard and weighted soft formulas. We claim that improvements defined for SAT, like detection of contradictory subformulas instead of contradictory literals, are also valid in our framework or can be easily adapted.

Tableaux have played a central role in automated deduction in first-order logic, as well as in other non-classical logics [9,12], and this work might be a first step towards dealing with optimization problems in those logics. From the propositional perspective, tableaux might be used to find new proof complexity results as the ones found for MaxSAT resolution [6,14], as well as to better understand MaxSAT and the logic behind. An interesting open problem is to find out how to define a complete tableau calculus for non-clausal MinSAT [3,20].

References

1. Abramé, A., Habet, D.: Efficient application of max-sat resolution on inconsistent subsets. In: Proceedings of the 20th International Conference on Principles and Practice of Constraint Programming, CP, Lyon, France, pp. 92–107 (2014)

2. Abramé, A., Habet, D.: On the resiliency of unit propagation to Max-Resolution. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI, Buenos Aires, Argentina, pp. 268–274 (2015)
3. Argelich, J., Li, C.M., Manyà, F., Soler, J.R.: Clause branching in MaxSAT and MinSAT. In: Proceedings of the 21st International Conference of the Catalan Association for Artificial Intelligence, Roses, Spain. Frontiers in Artificial Intelligence and Applications, vol. 308, pp. 17–26. IOS Press (2018)
4. Argelich, J., Li, C.M., Manyà, F., Soler, J.R.: Clause tableaux for maximum and minimum satisfiability. Logic J. IGPL (2019). https://doi.org/10.1093/jigpal/jzz025
5. Bofill, M., Garcia, M., Suy, J., Villaret, M.: MaxSAT-based scheduling of B2B meetings. In: Proceedings of the12th International Conference on Integration of AI and OR Techniques in Constraint Programming, CPAIOR, Barcelona, Spain, pp. 65–73 (2015)
6. Bonet, M.L., Buss, S., Ignatiev, A., Marques-Silva, J., Morgado, A.: MaxSAT resolution with the dual rail encoding. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, AAAI, New Orleans, Louisiana, USA, pp. 6565–6572 (2018)
7. Bonet, M.L., Levy, J., Manyà, F.: Resolution for Max-SAT. Artif. Intell. **171**(8–9), 240–251 (2007)
8. Casas-Roma, J., Huertas, A., Manyà, F.: Solving MaxSAT with natural deduction. In: Proceedings of the 20th International Conference of the Catalan Association for Artificial Intelligence, Deltebre, Spain. Frontiers in Artificial Intelligence and Applications, vol. 300, pp. 186–195. IOS Press (2017)
9. D'Agostino, M.: Tableaux methods for classical propositional logic. In: D'Agostino, M., Gabbay, D., Hähnle, R., Posegga, J. (eds.) Handbook of Tableau Methods, pp. 45–123. Springer, Dordrecht (1999). https://doi.org/10.1007/978-94-017-1754-0_2
10. D'Almeida, D., Grégoire, É.: Model-based diagnosis with default information implemented through MAX-SAT technology. In: Proceedings of the IEEE 13th International Conference on Information Reuse & Integration, IRI, Las Vegas, NV, USA, pp. 33–36 (2012)
11. Guerra, J., Lynce, I.: Reasoning over biological networks using maximum satisfiability. In: Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming, CP, Québec City, QC, Canada, pp. 941–956 (2012)
12. Hähnle, R.: Tableaux and related methods. In: Robinson, J.A., Voronkov, A. (eds.) Handbook of Automated Reasoning, pp. 100–178. Elsevier and MIT Press (2001)
13. Heras, F., Larrosa, J.: New inference rules for efficient Max-SAT solving. In: Proceedings of the National Conference on Artificial Intelligence, AAAI-2006, Boston/MA, USA, pp. 68–73 (2006)
14. Ignatiev, A., Morgado, A., Marques-Silva, J.: On tackling the limits of resolution in SAT solving. In: Gaspers, S., Walsh, T. (eds.) SAT 2017. LNCS, vol. 10491, pp. 164–183. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66263-3_11
15. Jabbour, S., Mhadhbi, N., Raddaoui, B., Sais, L.: A SAT-based framework for overlapping community detection in networks. In: Proceedings of the 21st Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, Part II, PAKDD, Jeju, South Korea, pp. 786–798 (2017)
16. Larrosa, J., Heras, F.: Resolution in Max-SAT and its relation to local consistency in weighted CSPs. In: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-2005, Edinburgh, Scotland, pp. 193–198. Morgan Kaufmann (2005)

17. Li, C.M., Manyà, F., Planes, J.: New inference rules for Max-SAT. J. Artif. Intell. Res. **30**, 321–359 (2007)
18. Li, C.M., Manyà, F., Soler, J.R.: A clause tableaux calculus for MaxSAT. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI-2016, New York, USA, pp. 766–772 (2016)
19. Li, C.M., Manyà, F., Soler, J.R.: Clausal form transformation in MaxSAT. In: Proceedings of the 49th IEEE International Symposium on Multiple-Valued Logic, ISMVL, Fredericton, Canada, pp. 132–137 (2019)
20. Li, C.M., Zhu, Z., Manyà, F., Simon, L.: Optimizing with minimum satisfiability. Artif. Intell. **190**, 32–44 (2012)
21. Manyà, F., Negrete, S., Roig, C., Soler, J.R.: A MaxSAT-based approach to the team composition problem in a classroom. In: Sukthankar, G., Rodriguez-Aguilar, J.A. (eds.) AAMAS 2017. LNCS (LNAI), vol. 10643, pp. 164–173. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71679-4_11
22. Marques-Silva, J., Argelich, J., Graça, A., Lynce, I.: Boolean lexicographic optimization: algorithms & applications. Ann. Math. Artif. Intell. **62**(3–4), 317–343 (2011)
23. Safarpour, S., Mangassarian, H., Veneris, A.G., Liffiton, M.H., Sakallah, K.A.: Improved design debugging using maximum satisfiability. In: Proceedings of 7th International Conference on Formal Methods in Computer-Aided Design, FMCAD, Austin, Texas, USA, pp. 13–19 (2007)
24. Smullyan, R.: First-Order Logic. Dover Publications, New York (1995). Second corrected edition, First published 1968 by Springer-Verlag
25. Xu, H., Rutenbar, R.A., Sakallah, K.A.: Sub-sat: a formulation for relaxed boolean satisfiability with applications in routing. IEEE Trans. CAD Integr. Circuits Syst. **22**(6), 814–820 (2003)
26. Zhang, L., Bacchus, F.: MAXSAT heuristics for cost optimal planning. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence, Toronto, Ontario, Canada, pp. 1846–1852 (2012)