KLM Logics of Nonmonotonic Reasoning: Calculi and Implementations

Laura Giordano[1], Valentina Gliozzi[2], Nicola Olivetti[3], and Gian Luca Pozzato[2]

[1] Dipartimento di Informatica - Università del Piemonte Orientale "A. Avogadro" - via Bellini 25/G - 15100 Alessandria, Italy
laura@mfn.unipmn.it
[2] Dipartimento di Informatica - Università degli Studi di Torino, corso Svizzera 185 - 10149 Turin - Italy
{gliozzi,pozzato}@di.unito.it
[3] LSIS - UMR CNRS 6168 Université Paul Cézanne (Aix-Marseille 3), Avenue Escadrille Normandie-Niemen 13397 Marseille Cedex 20 - France
nicola.olivetti@univ-cezanne.fr

Abstract. We present proof methods for the logics of nonmonotonic reasoning defined by Kraus, Lehmann and Magidor (KLM). We introduce tableaux calculi (called $\mathcal{T}\mathbf{S}^{\mathbf{T}}$) for all KLM logics. We provide decision procedures for KLM logics based on these calculi, and we study their complexity. Finally, we describe KLMLean 2.0, a theorem prover implementing the calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ inspired by the "lean" methodology. KLMLean 2.0 is implemented in SICStus Prolog and it also comprises a graphical interface written in Java[1].

1 Introduction

In the early 90' [13, 14] Kraus, Lehmann and Magidor (from now on KLM) proposed a formalization of non-monotonic reasoning that was early recognized as a landmark. Their work stemmed from two sources: the theory of nonmonotonic consequence relations initiated by Gabbay [8] and the preferential semantics proposed by Shoham [16] as a generalization of Circumscription. Their works lead to a classification of nonmonotonic consequence relations, determining a hierarchy of stronger and stronger systems.

According to the KLM framework, defeasible knowledge is represented by a (finite) set of nonmonotonic conditionals or assertions of the form

$$A \mathrel{|\!\sim} B$$

whose reading is *normally (or typically) the A's are B's*. The operator "$\mathrel{|\!\sim}$" is nonmonotonic, in the sense that $A \mathrel{|\!\sim} B$ does not imply $A \wedge C \mathrel{|\!\sim} B$. For instance, a

[1] This research has been partially supported by "*Progetto Lagrange - Fondazione CRT*" and by the projects "*MIUR PRIN05: Specification and verification of agent interaction protocols*" and *"GALILEO 2006: Interazione e coordinazione nei sistemi multi-agenti"*.

knowledge base K may contain $\mathit{football\_lover} \mathrel{|\!\sim} \mathit{bet}$, $\mathit{football\_player} \mathrel{|\!\sim} \mathit{football\_lover}$, $\mathit{football\_player} \mathrel{|\!\sim} \neg \mathit{bet}$, whose meaning is that people loving football typically bet on the result of a match, football players typically love football but they typically do not bet (especially on matches they are going to play...). If $\mathrel{|\!\sim}$ were interpreted as classical (or intuitionistic) implication, one would get $\mathit{football\_player} \mathrel{|\!\sim} \bot$, i.e. typically there are not football players, thereby obtaining a trivial knowledge base.

In KLM framework, one can derive new conditional assertions from the knowledge base by means of a set of inference rules. The set of adopted inference rules defines some fundamental types of inference systems, namely, from the strongest to the weakest: Rational (**R**), Preferential (**P**), Loop-Cumulative (**CL**), and Cumulative (**C**) logic. In all these systems one can infer new assertions without incurring the trivializing conclusions of classical logic: in the above example, in none of the systems, one can infer $\mathit{football\_player} \mathrel{|\!\sim} \mathit{bet}$. In cumulative logics (both **C** and **CL**) one can infer $\mathit{football\_lover} \wedge \mathit{football\_player} \mathrel{|\!\sim} \neg \mathit{bet}$, giving preference to more specific information; in Preferential logic **P** one can also infer that $\mathit{football\_lover} \mathrel{|\!\sim} \neg \mathit{football\_player}$; in the rational case **R**, if one further knows that $\neg(\mathit{football\_lover} \mathrel{|\!\sim} \mathit{rich})$, that is to say it is not the case that football lovers are typically rich persons, one can also infer that $\mathit{football\_lover} \wedge \neg \mathit{rich} \mathrel{|\!\sim} \mathit{bet}$.

From a semantic point of view, to the each logic (**R**, **P**, **CL**, **C**) corresponds one kind of models, namely, possible-world structures equipped with a preference relation among worlds or states. More precisely, for **P** we have models with a preference relation (an irreflexive and transitive relation) on worlds. For the stronger **R** the preference relation is further assumed to be *modular*. For the weaker logic **CL**, the preference relation is defined on *states*, where a state can be identified, intuitively, with a set of worlds. In the weakest case of **C**, the preference relation is on states, as for **CL**, but it is no longer assumed to be transitive. In all cases, the meaning of a conditional assertion $A \mathrel{|\!\sim} B$ is that B holds in the *most preferred* worlds/states where A holds.

A recent result by Halpern and Friedman [6] has shown that preferential and rational logic are quite natural and general systems: surprisingly enough, the axiom system of preferential (likewise of rational logic) is complete with respect to a wide spectrum of semantics, from ranked models, to parametrized probabilistic structures, ϵ-semantics and possibilistic structures. The reason is that all these structures are examples of *plausibility structures* and the truth in them is captured by the axioms of preferential (or rational) logic. These results, and their extensions to the first order setting [7] are the source of a renewed interest in KLM framework.

Even if KLM was born as an inferential approach to nonmonotonic reasoning, curiously enough, there has not been much investigation on deductive mechanisms for these logics. In short, the state of the art is as follows:

- Lehmann and Magidor [14] have proved that validity in **P** is **coNP**-complete. Their decision procedure for **P** is more a theoretical tool than a practical algorithm, as it requires to guess sets of indexes and propositional evaluations. They have also provided another procedure for **P** that exploits its reduction

to **R**. However, the reduction of **P** to **R** breaks down if boolean combinations of conditionals are allowed, indeed it is exactly when such combinations are allowed that the difference between **P** and **R** arises.
- A tableau proof procedure for **C** has been given in [1]. Their tableau procedure is fairly complicated; it uses labels and it contains a cut-rule. Moreover, it is not clear how it can be adapted to **CL** and **P**.
- In [11] it is defined a labelled tableau calculus for the conditional logic **CE** whose flat fragment (i.e. without nested conditionals) corresponds to **P**. That calculus needs a fairly complicated loop-checking mechanism to ensure termination. It is not clear if it matches complexity bounds and if it can be adapted in a simple way to **CL**.
- Finally, decidability of **P** and **R** has also been obtained by interpreting them into standard modal logics, as it is done by Boutilier [4]. However, his mapping is not very direct and natural, as we discuss below.

In this work we present tableau procedures for all KLM logics called $\mathcal{T}\mathbf{S}^{\mathbf{T}}$, where **S** stands for $\{\mathbf{R}, \mathbf{P}, \mathbf{CL}, \mathbf{C}\}$. Our approach is based on a novel interpretation of **P** into modal logics. As a difference with previous approaches (e.g. Crocco and Lamarre [5] and Boutillier [4]), that take S4 as the modal counterpart of **P**, we consider here Gödel-Löb modal logic of provability G (see for instance [12]). Our tableau method provides a sort of run-time translation of **P** into modal logic G. The idea is simply to interpret the preference relation as an accessibility relation: a conditional $A \mathrel{|\!\sim} B$ holds in a model if B is true in all minimal A-worlds w (i.e. worlds in which A holds and that are minimal). An A-world w is a minimal A-world if all smaller worlds are not A-worlds. The relation with modal logic G is motivated by the fact that we assume, following KLM, the so-called *smoothness condition*, which is related to the well-known *limit assumption*. This condition ensures that minimal A-worlds exist whenever there are A-worlds, by preventing infinitely descending chains of worlds. This condition therefore corresponds to the finite-chain condition on the accessibility relation (as in modal logic G). Therefore, our interpretation of conditionals is different from the one proposed by Boutilier, who rejects the smoothness condition and then gives a less natural (and more complicated) interpretation of **P** into modal logic S4. We are able to extend our approach to the cases of **CL** and **C** by using a second modality which takes care of states. As a difference with **P** and **CL**, the calculus for **C** requires a sort of (analytic) cut rule to account for the smoothness condition. We also consider the case of the strongest logic **R**; as for the other weaker systems, our approach is based on an interpretation of **R** into an extension of modal logic G, including modularity of the preference relation (previous approaches [5, 4] take S4.3 as the modal counterpart of **R**). As a difference with the tableau calculi introduced for **P**, **CL**, and **C**, here we present a *labelled* tableau system, which seems to be the most natural approach in order to capture the modularity of the preference relation.

The tableau calculi mentioned above can be used to define a systematic procedure which allows the satisfiability problem for **R**, **P**, and **CL** to be decided in nondeterministic polynomial time; moreover, we obtain a complexity bound

for these logics, namely that they are **coNP**-complete. For **R** and **P**, this is in accordance with the known complexity results, whereas for **CL** this bound is new, to the best of our knowledge. The calculus $\mathcal{T}\mathbf{C}^{\mathbf{T}}$ gives nonetheless a decision procedure for this logic too.

Finally, we present KLMLean 2.0, a theorem prover implementing $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ calculi in SICStus Prolog. KLMLean 2.0 is inspired by the "lean" methodology [3], and it also comprises a graphical interface written in Java. For the rational logic **R**, KLMLean 2.0 offers two different versions: 1. a simple version, where Prolog *constants* are used to represent $\mathcal{T}\mathbf{R}^{\mathbf{T}}$'s labels; 2. a more efficient one, where labels are represented by Prolog *variables*, inspired by the free-variable tableau presented in [2]. To the best of our knowledge, KLMLean 2.0 is the first theorem prover for KLM logics.

The plan of the paper is as follows: in section 2, we recall KLM logics, and we show how their semantics can be represented by standard Kripke models. In section 3, we present the tableaux calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ and we elaborate them in order to obtain a terminating procedure. Also, we summarize complexity results obtained by analyzing $\mathcal{T}\mathbf{S}^{\mathbf{T}}$. In section 4 we present KLMLean 2.0, a theorem prover written in SICStus Prolog implementing the $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ calculi.

2 KLM Logics

We briefly recall the axiomatizations and semantics of the KLM systems. For the sake of exposition, we present the systems in the order from the strongest to the weakest: **R**, **P**, **CL**, and **C**. For a complete picture of KLM systems, see [13, 14]. The language of KLM logics consists just of conditional assertions $A \mathrel{|\!\sim} B$. We consider a richer language allowing boolean combinations of assertions and propositional formulas. Our language $\mathcal{L}$ is defined from a set of propositional variables ATM, the boolean connectives and the conditional operator $\mathrel{|\!\sim}$. We use $A, B, C, ...$ to denote propositional formulas (that do not contain conditional formulas), whereas $F, G, ...$ are used to denote all formulas (including conditionals); $\Gamma, \Delta, ...$ represent sets of formulas, whereas $X, Y, ...$ denote sets of sets of formulas. The formulas of $\mathcal{L}$ are defined as follows: if A is a propositional formula, $A \in \mathcal{L}$; if A and B are propositional formulas, $A \mathrel{|\!\sim} B \in \mathcal{L}$; if F is a boolean combination of formulas of $\mathcal{L}$, $F \in \mathcal{L}$.

The axiomatization of **R** consists of all axioms and rules of propositional calculus together with the following axioms and rules. We use $\vdash_{PC}$ to denote provability in the propositional calculus, whereas $\vdash$ is used to denote provability in **R**:

- REF. $A \mathrel{|\!\sim} A$ (reflexivity)
- LLE. If $\vdash_{PC} A \leftrightarrow B$, then $\vdash (A \mathrel{|\!\sim} C) \rightarrow (B \mathrel{|\!\sim} C)$ (left logical equivalence)
- RW. If $\vdash_{PC} A \rightarrow B$, then $\vdash (C \mathrel{|\!\sim} A) \rightarrow (C \mathrel{|\!\sim} B)$ (right weakening)
- CM. $((A \mathrel{|\!\sim} B) \wedge (A \mathrel{|\!\sim} C)) \rightarrow (A \wedge B \mathrel{|\!\sim} C)$ (cautious monotonicity)
- AND. $((A \mathrel{|\!\sim} B) \wedge (A \mathrel{|\!\sim} C)) \rightarrow (A \mathrel{|\!\sim} B \wedge C)$
- OR. $((A \mathrel{|\!\sim} C) \wedge (B \mathrel{|\!\sim} C)) \rightarrow (A \vee B \mathrel{|\!\sim} C)$

– RM. $((A \mathrel{|\!\sim} B) \wedge \neg(A \mathrel{|\!\sim} \neg C)) \rightarrow ((A \wedge C) \mathrel{|\!\sim} B)$ (rational monotonicity)

REF states that A is always a default conclusion of A. LLE states that the syntactic form of the antecedent of a conditional formula is irrelevant. RW describes a similar property of the consequent. This allows to combine default and logical reasoning [6]. CM states that if B and C are two default conclusions of A, then adding one of the two conclusions to A will not cause the retraction of the other conclusion. AND states that it is possible to combine two default conclusions. OR states that it is allowed to reason by cases: if C is the default conclusion of two premises A and B, then it is also the default conclusion of their disjunction. RM is the rule of *rational monotonicity*, which characterizes the logic **R**: if $A \mathrel{|\!\sim} B$ and $\neg(A \mathrel{|\!\sim} \neg C)$ hold, then one can infer $A \wedge C \mathrel{|\!\sim} B$. This rule allows a conditional to be inferred from a set of conditionals in absence of other information. More precisely, "it says that an agent should not have to retract any previous defeasible conclusion when learning about a new fact the negation of which was not previously derivable" [14].

The axiomatization of **P** can be obtained from the axiomatization of **R** by removing the axiom RM.

The axiomatization of **CL** can be obtained from the axiomatization of **P** by removing the axiom OR and by adding the following infinite set of LOOP axioms:

LOOP. $(A_0 \mathrel{|\!\sim} A_1) \wedge (A_1 \mathrel{|\!\sim} A_2)...(A_{n-1} \mathrel{|\!\sim} A_n) \wedge (A_n \mathrel{|\!\sim} A_0) \rightarrow (A_0 \mathrel{|\!\sim} A_n)$

and the following axiom CUT:

$$\text{CUT. } ((A \mathrel{|\!\sim} B) \wedge (A \wedge B \mathrel{|\!\sim} C)) \rightarrow (A \mathrel{|\!\sim} C)$$

Notice that these axioms are derivable in **P** (and therefore in **R**). The weakest logical system considered by KLM is Cumulative Logic **C**, whose axiom system can be obtained from the one for **CL** by removing the set of (LOOP) axioms.

The semantics of KLM logics is defined by considering possible world (or possible states) structures with a *preference relation* $w < w'$ among worlds (or states), whose meaning is that w is preferred to w'. $A \mathrel{|\!\sim} B$ holds in a model $\mathcal{M}$ if B holds in all *minimal worlds* (*states*) where A holds. This definition makes sense provided minimal worlds for A exist whenever there are A-worlds (A-states): this is ensured by the *smoothness condition* defined below.

Definition 1 (Rational and Preferential models). *A* rational *model is a triple*

$$\mathcal{M} = \langle \mathcal{W}, <, V \rangle$$

where $\mathcal{W}$ is a non-empty set of items called worlds, $<$ is an irreflexive, transitive and modular[2] *relation on $\mathcal{W}$, and V is a function $V : \mathcal{W} \longmapsto 2^{ATM}$, which assigns to every world w the set of atoms holding in that world. The truth conditions for a formula F are as follows:*

[2] A relation $<$ is *modular* if the following condition holds: for each u, v, w, if $u < v$, then either $w < v$ or $u < w$.

- *if F is a boolean combination of formulas, $\mathcal{M}, w \models F$ is defined as for propositional logic, namely:*
  - *if F is an atom $P \in ATM$, then $\mathcal{M}, w \models P$ iff $P \in V(w)$;*
  - *if F is a negation $\neg G$, then $\mathcal{M}, w \models \neg G$ iff $\mathcal{M}, w \not\models G$;*
  - *if F is a conjunction $G_1 \wedge G_2$, then $\mathcal{M}, w \models G_1 \wedge G_2$ iff $\mathcal{M}, w \models G_1$ and $\mathcal{M}, w \models G_2$;*
  - *if F is a disjunction $G_1 \vee G_2$, then $\mathcal{M}, w \models G_1 \vee G_2$ iff $\mathcal{M}, w \models G_1$ or $\mathcal{M}, w \models G_2$;*
  - *if F is an implication $G_1 \rightarrow G_2$, then $\mathcal{M}, w \models G_1 \rightarrow G_2$ iff $\mathcal{M}, w \not\models G_1$ or $\mathcal{M}, w \models G_2$.*

Let A be a propositional formula; we define $Min_<(A) = \{w \in \mathcal{W} \mid \mathcal{M}, w \models A$ and $\forall w'$, $w' < w$ implies $\mathcal{M}, w' \not\models A\}$. We define:

- *$\mathcal{M}, w \models A \mathrel{|\!\sim} B$ if for all w', if $w' \in Min_<(A)$ then $\mathcal{M}, w' \models B$.*

We also define the smoothness condition *on the preference relation: if $\mathcal{M}, w \models A$, then $w \in Min_<(A)$ or $\exists w' \in Min_<(A)$ s.t. $w' < w$. Validity and satisfiability of a formula are defined as usual.*

A preferential *model is defined as the rational/preferential model, with the only difference that the preference relation $<$ is no longer assumed to be modular.*

Observe that the above definition of rational model extends the one given by KLM to boolean combinations of formulas. Notice also that the truth conditions for conditional formulas are given with respect to single possible worlds for uniformity sake. Since the truth value of a conditional only depends on global properties of $\mathcal{M}$, we have that: $\mathcal{M}, w \models A \mathrel{|\!\sim} B$ iff $\mathcal{M} \models A \mathrel{|\!\sim} B$.

Models for (loop-)cumulative logics also comprise states:

Definition 2 (Loop-Cumulative and Cumulative models). *A* (loop-)cumulative *model is a tuple*

$$\mathcal{M} = \langle S, \mathcal{W}, l, <, V \rangle$$

where S is a set of states and $l : S \mapsto 2^{\mathcal{W}}$ is a function that labels every state with a nonempty set of worlds; $<$ is defined on S, it satisfies the smoothness condition and it is irreflexive and transitive in **CL**, *whereas it is only irreflexive in* **C**.

For $s \in S$ and A propositional, we let $\mathcal{M}, s \mid\!\equiv A$ if $\forall w \in l(s)$, then $\mathcal{M}, w \models A$, where $\mathcal{M}, w \models A$ is defined as for propositional logic. Let $Min_<(A)$ be the set of minimal states s such that $\mathcal{M}, s \mid\!\equiv A$. We define $\mathcal{M}, s \mid\!\equiv A \mathrel{|\!\sim} B$ if $\forall s' \in Min_<(A)$, $\mathcal{M}, s' \mid\!\equiv B$. The relation $\mid\!\equiv$ can be extended to boolean combinations of conditionals in the standard way. We assume that $<$ satisfies the smoothness condition.

The language of the tableau calculi that we will introduce in the next section extends $\mathcal{L}$ by formulas of the form $\Box A$, where A is propositional, whose intuitive meaning is that $\Box A$ holds in a world/state w if A holds in all the worlds/states preferred to w (i.e. in all w' such that $w' < w$). We extend the notion of KLM model to provide an evaluation of boxed formulas as follows:

Definition 3 (Truth condition of modality $\Box$). *We define the truth condition of a boxed formula as follows:*

- (**R** *and* **P**)*:* $\mathcal{M}, w \models \Box A$ *if, for every* $w' \in \mathcal{W}$*, if* $w' < w$ *then* $\mathcal{M}, w' \models A$
- (**CL** *and* **C**)*:* $\mathcal{M}, s \models\!\!\mid \Box A$ *if, for every* $s' \in S$*, if* $s' < s$ *then* $\mathcal{M}, s' \models\!\!\mid A$

From definition of $Min_<(A)$ in Definitions 1 and 2 above, and Definition 3, it follows that for any formula A, $w \in Min_<(A)$ iff $\mathcal{M}, w \models A \wedge \Box\neg A$ (resp. $s \in Min_<(A)$ iff $\mathcal{M}, s \models\!\!\mid A \wedge \Box\neg A$).

3 Tableaux Calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$

In this section we present analytic tableaux calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ for KLM logics, where **S** stands for $\{\mathbf{R}, \mathbf{P}, \mathbf{CL}, \mathbf{C}\}$. The basic idea is simply to interpret the preference relation as an accessibility relation. The calculi for **R** and **P** implement a sort of run-time translation into (extensions of) Gödel-Löb modal logic of provability G. This is motivated by the fact that we assume the smoothness condition, which ensures that minimal A-worlds exist whenever there are A-worlds, by preventing infinitely descending chains of worlds. This condition therefore corresponds to the finite-chain condition on the accessibility relation (as in modal logic G). This approach is extended to the cases of **CL** and **C** by using a second modality L which takes care of states. For a broader discussion on those calculi, see [9, 15, 10].

The rules of the calculi manipulate sets of formulas Γ. We write Γ, F as a shorthand for $\Gamma \cup \{F\}$. Moreover, given Γ we define the following sets:

- $\Gamma^{\Box} = \{\Box\neg A \mid \Box\neg A \in \Gamma\}$
- $\Gamma^{\Box^{\downarrow}} = \{\neg A \mid \Box\neg A \in \Gamma\}$
- $\Gamma^{\mid\!\sim^{\pm}} = \{A \mid\sim B \mid A \mid\sim B \in \Gamma\} \cup \{\neg(A \mid\sim B) \mid \neg(A \mid\sim B) \in \Gamma\}$
- $\Gamma^{L^{\downarrow}} = \{A \mid LA \in \Gamma\}$.

As mentioned, the calculus for rational logic **R** makes use of *labelled* formulas (see Figure 1), where the labels are drawn from a denumerable set $\mathcal{A}$; there are two kinds of formulas: 1. *world formulas*, denoted by $x : F$, where $x \in \mathcal{A}$ and $F \in \mathcal{L}$; 2. *relation formulas*, denoted by $x < y$, where $x, y \in \mathcal{A}$, representing the preference relation. Rules of the calculus $\mathcal{T}\mathbf{R}^{\mathbf{T}}$ also manipulate the following set of formulas: $\Gamma^{M}_{x \to y} = \{y : \neg A, y : \Box\neg A \mid x : \Box\neg A \in \Gamma\}$. The following definition states the truth conditions for labelled formulas:

Definition 4 (Truth conditions of formulas of $\mathcal{T}\mathbf{R}$). *Given a model* $\mathcal{M} = \langle \mathcal{W}, <, V\rangle$ *and a label alphabet* $\mathcal{A}$*, we consider a mapping* $I : \mathcal{A} \mapsto \mathcal{W}$*. Given a formula* α *of the calculus* $\mathcal{T}\mathbf{R}$*, we define* $\mathcal{M} \models_I \alpha$ *as follows:* $\mathcal{M} \models_I x : F$ *iff* $\mathcal{M}, I(x) \models F$*;* $\mathcal{M} \models_I x < y$ *iff* $I(x) < I(y)$*.*
We say that a set Γ *of formulas of* $\mathcal{T}\mathbf{R}$ *is satisfiable if, for all formulas* $\alpha \in \Gamma$*, we have that* $\mathcal{M} \models_I \alpha$*, for some model* $\mathcal{M}$ *and some mapping* I*.*

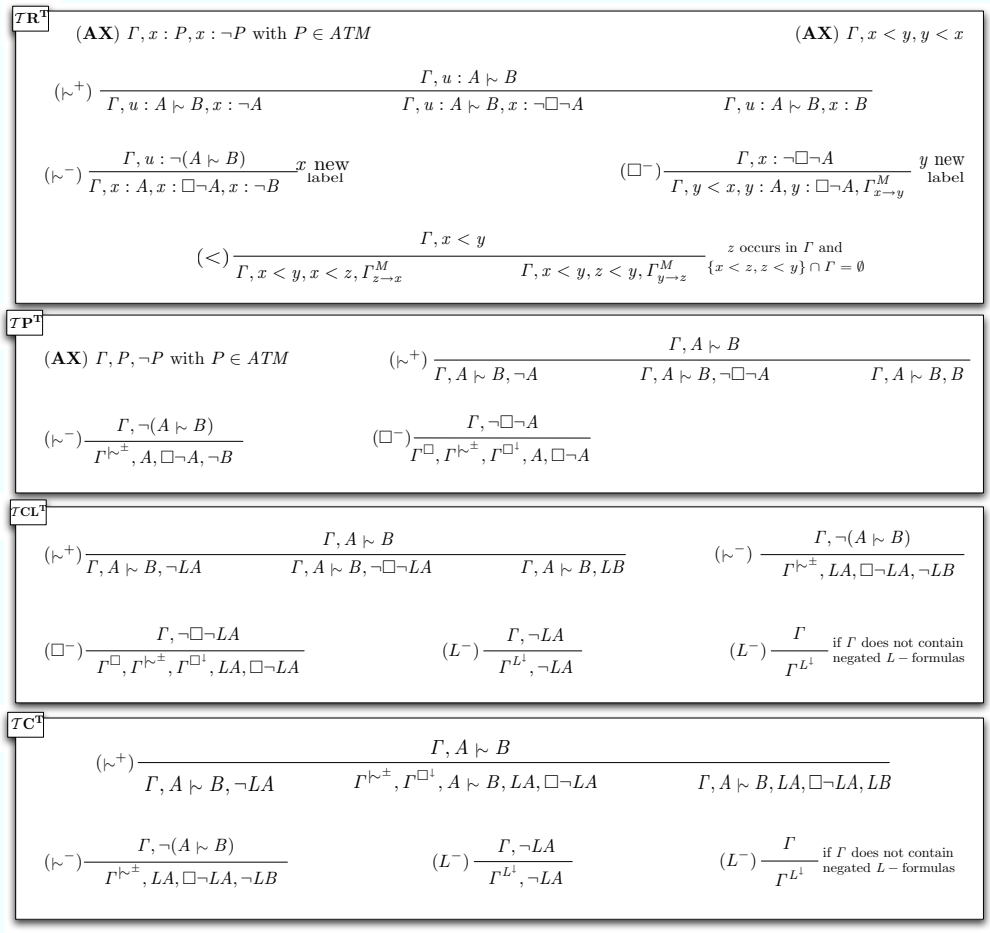

Fig. 1. Tableau systems $\mathcal{T}\mathbf{S}^{\mathbf{T}}$. To save space, we omit the standard rules for boolean connectives. For $\mathcal{T}\mathbf{CL}^{\mathbf{T}}$ and $\mathcal{T}\mathbf{C}^{\mathbf{T}}$ the axiom (**AX**) is as in $\mathcal{T}\mathbf{P}^{\mathbf{T}}$.

While the calculus for **R** makes use of labelled formulas, the calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ for all other KLM logics, presented in Figure 1, do not make use of labels. A tableau is a tree whose nodes are sets of formulas Γ. A branch is a sequence of sets of formulas $\Gamma_1, \Gamma_2, \ldots, \Gamma_n, \ldots$, where each node Γ_i is obtained by its immediate predecessor Γ_{i-1} by applying a rule of $\mathcal{T}\mathbf{S}^{\mathbf{T}}$, having Γ_{i-1} as the premise and Γ_i as one of its conclusions. A branch is closed if one of its nodes is an instance of (**AX**), otherwise it is open. We say that a tableau is closed if all its branches are closed.

The rules $(\mathrel{|\!\sim}^{-})$ and $(\Box^{-})$ are called *dynamic*, since they introduce a new world in their conclusions. The other rules are called *static*. In order to prove that *football_lover* $\mathrel{|\!\sim} \neg$*football_player* can be derived in preferential logic **P** from the knowledge base {*football_lover* $\mathrel{|\!\sim}$ *bet*, *football_player* $\mathrel{|\!\sim}$ *football_lover*, *football_player*

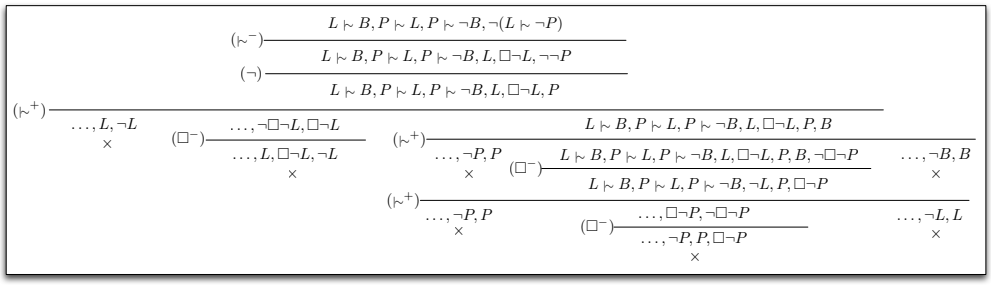

Fig. 2. A derivation in $\mathcal{T}\mathbf{P}^{\mathbf{T}}$ for $\{$*football_lover* $\mathrel{|\!\sim}$ *bet*, *football_player* $\mathrel{|\!\sim}$ *football_lover*, *football_player* $\mathrel{|\!\sim}$ $\neg$*bet*, $\neg$(*football_lover* $\mathrel{|\!\sim}$ $\neg$*football_player*)$\}$. To increase readability, we use P to denote *football_player*, L to denote *football_lover* and B for *bet*.

$\mathrel{|\!\sim}$ $\neg$*bet*$\}$, one needs to search a closed tableau in $\mathcal{T}\mathbf{P}^{\mathbf{T}}$ for the set of formulas $\{$*football_lover* $\mathrel{|\!\sim}$ *bet*, *football_player* $\mathrel{|\!\sim}$ *football_lover*, *football_player* $\mathrel{|\!\sim}$ $\neg$*bet*, $\neg$(*football_lover* $\mathrel{|\!\sim}$ $\neg$*football_player*)$\}$. In Figure 2 a closed tableau in $\mathcal{T}\mathbf{P}^{\mathbf{T}}$ for the above unsatisfiable set of formulas is presented.

The calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ are sound and complete wrt the semantics:

Theorem 1 (Soundness and completeness of $\mathcal{T}\mathbf{S}^{\mathbf{T}}$, [9, 15, 10]). *Given a set of formulas Γ of $\mathcal{L}$, it is unsatisfiable if and only if there is a closed tableau in $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ having Γ as a root.*

3.1 Termination and Complexity Results

The calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ do not ensure termination. However, we are able to turn them into terminating calculi. In general, non-termination in tableau calculi can be caused by two different reasons: 1. dynamic rules can generate infinitely-many worlds, creating infinite branches; 2. some rules copy their principal formula in the conclusion, thus can be reapplied over the same formula without any control.

Concerning point 1, we are able to show that the generation of infinite branches due to the interplay between rules $(\mid\!\sim^+)$ and $(\Box^-)$ cannot occur. Intuitively, the application of $(\Box^-)$ to a formula $\neg\Box\neg A$ (introduced by $(\mid\!\sim^+)$) adds the formula $\Box\neg A$ to the conclusion, so that $(\mid\!\sim^+)$ can no longer consistently introduce $\neg\Box\neg A$. This is due to the properties of $\Box$ in G. Furthermore, the $(\mid\!\sim^-)$ rule can be applied only once to a given negated conditional on a branch, thus infinitely-many worlds cannot be generated on a branch.

Concerning point 2, we have to control the application of the $(\mid\!\sim^+)$ rule, which can otherwise be applied without any control since it copies its principal formula $A \mathrel{|\!\sim} B$ in all its conclusions, then the conclusions have a higher complexity than the premise. We can show that it is useless to apply $(\mid\!\sim^+)$ *more than once in the same world/state*, therefore the calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ are modified to keep track of positive conditionals already considered in a world/state by moving them in an additional set Σ in the conclusions of $(\mid\!\sim^+)$, and restrict the application of this rule to unused conditionals only. The dynamic rules $(\mid\!\sim^-)$ and $(\Box^-)$, whose

conclusions represent a *different* world/state wrt the corresponding premise, reintroduce formulas from Σ in order to allow further applications of $(\mid\!\sim^+)$ in the other worlds/states. This machinery is standard. Concerning the labelled calculus $\mathcal{T}\mathbf{R}^\mathbf{T}$, the same mechanism is applied by equipping each positive conditional with the list L of worlds-labels in which $(\mid\!\sim^+)$ has already been applied, and restricting its application by using worlds not belonging to L. In [15, 9, 10] it is shown that no other machinery is needed to ensure termination, except for $\mathcal{T}\mathbf{C}^\mathbf{T}$, which needs a further standard loop-checking machinery.

Theorem 2 (Termination of $\mathcal{T}\mathbf{S}^\mathbf{T}$, [9, 15, 10]). *For all KLM logic* $\mathbf{S}$*,* $\mathcal{T}\mathbf{S}^\mathbf{T}$ *ensures a terminating proof search.*

We conclude this section by summarizing complexity results for $\mathcal{T}\mathbf{S}^\mathbf{T}$. In [9, 15, 10], we have shown that for the logics **R**, **P**, and **CL**, the tableaux calculi above can be used to define a **coNP** decision procedure to check the validity of a set of formulas of $\mathcal{L}$. The same cannot be done for the weakest logic **C**.

We can prove the following Theorem:

Theorem 3 (Complexity of R, P, and CL). *The problem of deciding validity for KLM logics* **R***,* **P***, and* **CL** *is* **coNP***-complete.*

Concerning **R** and **P**, this result matches the known complexity results for these logics. Concerning the logic **CL**, this complexity bound is new, to the best of our knowledge.

4 KLMLean 2.0: a Theorem Prover for KLM Logics

In this section we describe an implementation of $\mathcal{T}\mathbf{S}^\mathbf{T}$ calculi in SICStus Prolog. The program, called KLMLean 2.0, is inspired by the "lean" methodology [3] (even if it does not fit its style in a rigorous manner): the Prolog program consists in a set of clauses, each one representing a tableau rule or axiom; the proof search is provided for free by the mere depth-first search mechanism of Prolog, without any additional ad hoc mechanism. KLMLean 2.0 can be downloaded at `http://www.di.unito.it/~pozzato/klmlean 2.0`.

Let us first describe the implementation of non-labelled calculi for **P**, **CL**, and **C**. We represent each node of a proof tree (i.e. set of formulas) by a Prolog list. The tableaux calculi are implemented by the predicate

prove(Gamma,Sigma,Tree).

which succeeds if and only if the set of formulas Γ, represented by the list `Gamma`, is unsatisfiable. `Sigma` is the list representing the set Σ of *used conditionals*, and it is used in order to control the application of the $(\mid\!\sim^+)$ rule, as described in the previous section. When `prove` succeeds, `Tree` contains a representation of a

closed tableau[3]. For instance, to prove that $A \mathrel{|\!\sim} B \wedge C, \neg(A \mathrel{|\!\sim} C)$ is unsatisfiable in **P**, one queries KLMLean 2.0 with the following goal: `prove([a => (b and c), neg (a => c)],[ ],Tree).` The string "`=>`" is used to represent the conditional operator $\mathrel{|\!\sim}$, "`and`" is used to denote $\wedge$, and so on. Each clause of `prove` implements one axiom or rule of the tableaux calculi; for example, the clauses implementing (**AX**) and ($\mathrel{|\!\sim}^-$) are as follows:

```
prove(Gamma,_,tree(...)):-member(F,Gamma),member(neg F,Gamma),!.
prove(Gamma,Sigma,tree(...)):-select(neg (A => B),Gamma,NewGamma),
   conditionals(NewGamma,Cond),append(Cond,Sigma,DefGamma),
   prove([neg B|[box neg A|[A|DefGamma]]],[],...).
```

The clause for **(AX)** is applied when a formula F and its negation $\neg F$ belong to Γ. Notice that F is a formula of the language $\mathcal{L}$, even complex; KLMLean 2.0 extends **(AX)** to a generic formula F in order to increase its performances, without losing the soundness of the calculi. The clause for ($\mathrel{|\!\sim}^-$) is applied when a formula $\neg(A \mathrel{|\!\sim} B)$ belongs to Γ. The predicate `select` removes $\neg(A \mathrel{|\!\sim} B)$ from `Gamma`, then the auxiliary predicate `conditionals` is invoked to compute the set $\Gamma^{\mid\sim\pm}$ on the resulting list `NewGamma`; finally, the predicate `prove` is recursively invoked on the only conclusion of the rule. Notice that, since ($\mathrel{|\!\sim}^-$) is a dynamic rule, the conditionals belonging to Σ move to Γ in the conclusion (execution of `append`), in order to allow further applications of ($\mathrel{|\!\sim}^+$). To search for a derivation of a set of formulas Γ, KLMLean 2.0 proceeds as follows: first of all, if Γ is an instance of (**AX**), the goal will succeed immediately by using the clauses for the axioms. If it is not, then the first applicable rule will be chosen, e.g. if `Gamma` contains a formula `neg(neg F)`, then the clause for ($\neg$) rule will be used, invoking `prove` on its unique conclusion. KLMLean 2.0 proceeds in a similar way for the other rules. The ordering of the clauses is such that the boolean rules are applied before the other ones. In the case of cumulative logic **C**, KLMLean 2.0 implements a loop-checking machinery by equipping the `prove` predicate with an additional argument, called `Analyzed`, representing the list of sets of formulas already considered in the current branch. Clauses implementing $\mathcal{T}\mathbf{C}^{\mathbf{T}}$ are invoked only if the current set of formulas has not yet been considered, i.e. if it does not belong to `Analyzed`.

The theorem prover for rational logic **R** implements *labelled* tableau calculi $\mathcal{T}\mathbf{R}^{\mathbf{T}}$. It makes use of Prolog constants to represent labels: world formulas $x : A$ are represented by a Prolog list `[x,a]`, and relation formulas $x < y$ are represented by a list `[x,<,y]`. As for the other systems, each clause of the predicate `prove` implements a tableau rule or axiom. As an example, here is the clause implementing the rule ($<$), capturing the modularity of the preference relation:

[3] More in detail, the argument `Tree` is instantiated by a functor `tree(Rule,F,G,SubTree1,[SubTree2])`, tracing that the rule `Rule` has been applied to the principal formula `F Rule G` in the current set of formulas. `SubTree1` and `SubTree2` are functors `tree` representing the closed tableau for each conclusion of the rule.

```
prove(Gamma,Labels,Cond,tree(...)):-
   member([X,<,Y],Gamma),member(Z,Labels),X\=Z, Y\=Z,
   \+member([X,<,Z],Gamma),\+member([Z,<,Y],Gamma),!,
   gammaM(Gamma,Y,Z,ResLeft),gammaM(Gamma,Z,X,ResRight),
   append(ResLeft,Gamma,LeftConcl),append(ResRight,Gamma,RightConcl),
   prove([[Z,<,Y]|LeftConcl],Labels,Cond,...),!,
   prove([[X,<,Z]|RightConcl],Labels,Cond,...).
```

The predicate `gammaM(Gamma,X,Y,...)` computes the set $\Gamma^M_{x\to y}$ defined in the previous section. In this system, `Cond` is used in order to control the application of the $(\mid\!\sim^+)$ rule: it is a list whose elements have the form `[x,a => b]`, representing that $(\mid\!\sim^+)$ has been applied to $A \mid\!\sim B$ in the current branch by using the label x. In order to increase its performances, KLMLean for **R** adopts a heuristic approach (not very "lean") to implement the crucial $(\mid\!\sim^+)$ rule: the predicate `prove` *chooses the "best" positive conditional* to which apply the rule, and the *"best" label to use*. Roughly speaking, an application of $(\mid\!\sim^+)$ is considered to be better than another one if it leads to an immediate closure of more conclusions. Even if $(\mid\!\sim^+)$ is invertible, choosing the right label in the application of $(\mid\!\sim^+)$ is highly critical for the performances of the thorem prover. To postpone this choice as much as possible, for the logic **R** we have defined a more efficient version of the prover, inspired by the free-variable tableaux introduced in [2]. It makes use of *Prolog variables* to represent all the labels that can be used in a single application of the $(\mid\!\sim^+)$ rule. This version represents labels by integers starting from 1; by using integers we can easily express constraints on the range of the variable-labels. To this regard, the library `clpfd` is used to manage free-variables domains. In order to prove $\Gamma, u : A \mid\!\sim B$, KLMLean 2.0 will call `prove` on the following conclusions: $\Gamma, u : A \mid\!\sim B, Y : \neg A$; $\Gamma, u : A \mid\!\sim B, Y : \neg\Box\neg A$; $\Gamma, u : A \mid\!\sim B, Y : B$, where Y is a Prolog variable. Y will then be instantiated by Prolog's pattern matching to close a branch with an axiom. Predicate `Y in 1..Max` is used to define the domain of Y, where `Max` is the maximal integer occurring in the branch (i.e. the last label introduced). The list `Cond` here contains elements of the form `[a => b,Used]`, where `Used` is the list of free variables already introduced to apply $(\mid\!\sim^+)$ in the current branch. In order to ensure termination, the clause implementing $(\mid\!\sim^+)$ is applied *only if* |`Used`|< `Max`; the predicate `all_different([Y|Used])` is then invoked to ensure that all variables used to apply $(\mid\!\sim^+)$ on the same conditional will assume different values. On the unsatisfiable set $\neg(A \vee D \mid\!\sim F \vee \neg C \vee \neg B \vee A), (A \wedge B) \vee (C \wedge D) \mid\!\sim E \wedge F, \neg(P \mid\!\sim E \wedge F), \neg((A \wedge B) \vee (C \wedge D) \mid\!\sim G)$, the free-variables version succeeds in less than 2 ms, whereas the "standard" version requires 1.9 s.

The performances of KLMLean 2.0 are promising. We have tested the implementation for **R** over 300 sets of formulas: it terminates its computation in 236 cases in less than 2.5 s (204 in less than 100 ms). The results for **R** and for the other KLM logics are reported in Table 1:

KLMLean 2.0 has also a graphical user interface (GUI) implemented in Java. The GUI interacts with the SICStus Prolog implementation by means of the package `se.sics.jasper`. Thanks to the GUI, one does not need to know how to call

KLM logic	1 ms	10 ms	100 ms	1 s	2.5 s
R (with free-variables)	176	178	204	223	236
P	166	164	185	206	211
CL	119	118	136	150	159
C	76	77	92	110	123

Table 1. Some statistics for KLMLean 2.0.

the predicate `prove`, or if the program implements a labelled or an unlabelled deductive system; one just introduces the formulas of a knowledge base K (or the set of formulas to prove to be unsatisfiable) and a formula F, in order to prove if one can infer F from K, in a text box and searches a derivation by clicking a button. Moreover, one can choose the intended system of KLM logic, namely **R**, **P**, **CL** or **C**. When the analyzed set $K \cup \{\neg F\}$ of formulas is unsatisfiable, KLMLean offers these options:

- display a proof tree of the set in a special window;
- build a LaTeX file containing the same proof tree: compiling this file with LaTeX, one can obtain the closed tree in a pdf file, or ps, or dvi, and then can print it.

The theorem prover for rational logic **R** offers another functionality. If the initial set of formulas $K \cup \{\neg F\}$ is satisfiable, i.e. the predicate `prove` answers `no`, then KLMLean 2.0 also displays a model satisfying $K \cup \{\neg F\}$.

Some pictures of KLMLean 2.0 are presented in Figure 3.

5 Conclusions and Future Work

In this paper, we have presented tableau calculi $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ for all the logics of the KLM framework for nonmonotonic reasoning. We have shown a tableau calculus for rational logic **R**, preferential logic **P**, loop-cumulative logic **CL**, and cumulative logic **C**. The calculi presented give a decision procedure for the respective logics. Moreover, for **R**, **P** and **CL** we have shown that we can obtain **coNP** decision procedures by refining the rules of the respective calculi. In case of **C**, we obtain a decision procedure by adding a suitable loop-checking mechanism. In this case, our procedure gives an hyper exponential upper bound. Further investigation is needed to get a more efficient procedure. On the other hand, we are not aware of any tighter complexity bound for this logic.

All the calculi presented in this paper have been implemented by a theorem prover called KLMLean 2.0, which is a SICStus Prolog implementation of $\mathcal{T}\mathbf{S}^{\mathbf{T}}$ inspired to the "lean" methodology, whose basic idea is to write short programs and exploit the power of Prolog's engine as much as possible. To the best of our knowledge, KLMLean 2.0 is the first theorem prover for KLM logics.

We plan to extend our calculi to first order case. The starting point will be the analysis of first order preferential and rational logics by Friedman, Halpern

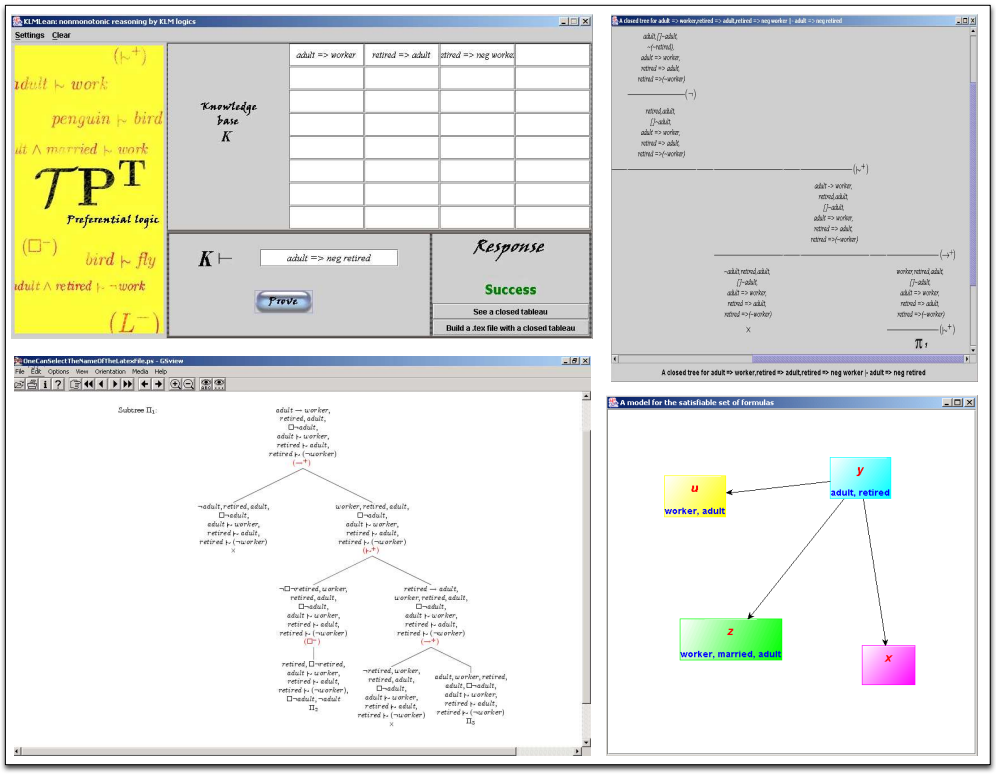

Fig. 3. Some pictures of KLMLean 2.0

and Koller in [7]. Moreover, we intend to increase the performances of KLMLean 2.0 by experimenting standard refinements and heuristics.

References

1. A. Artosi, G. Governatori, and A. Rotolo. Labelled tableaux for non-monotonic reasoning: Cumulative consequence relations. *J. of Logic and Computation*, 12(6):1027–1060, 2002.
2. B. Beckert and R. Goré. Free variable tableaux for propositional modal logics. *In Proc. of TABLEAUX 1997 (Automated Reasoning with Analytic Tableaux and Related Methods), volume 1227 of LNAI, Springer-Verlag*, pages 91–106, 1997.
3. B. Beckert and J. Posegga. leantap: Lean tableau-based deduction. *Journal of Automated Reasoning*, 15(3):339–358, 1995.
4. C. Boutilier. Conditional logics of normality: a modal approach. *Art. Int.*, 68(1):87–154, 1994.
5. G. Crocco and P. Lamarre. On the connection between non-monotonic inference systems and conditional logics. In *Proc. of KR 92*, pages 565–571, 1992.
6. N. Friedman and J. Y. Halpern. Plausibility measures and default reasoning. *Journal of the ACM*, 48(4):648–685, 2001.

7. N. Friedman, J. Y. Halpern, and D. Koller. First-order conditional logic for default reasoning revisited. *ACM TOCL*, 1(2):175–207, 2000.
8. D. Gabbay. Theoretical foundations for non-monotonic reasoning in expert systems. *Logics and models of concurrent systems, Springer*, pages 439–457, 1985.
9. L. Giordano, V. Gliozzi, N. Olivetti, and G. L. Pozzato. Analytic Tableaux for KLM Preferential and Cumulative Logics. In *Proc. of LPAR 2005, LNAI 3835*, pages 666–681. Springer, 2005.
10. L. Giordano, V. Gliozzi, N. Olivetti, and G. L. Pozzato. Analytic Tableaux Calculi for KLM Rational Logic R. In *Proc. of JELIA 2006*, volume 4160 of *LNAI*, pages 190–202. Springer, 2006.
11. L. Giordano, V. Gliozzi, N. Olivetti, and C. Schwind. Tableau calculi for preference-based conditional logics. In *Proc. of TABLEAUX 2003, LNAI 2796*, pages 81–101. Springer, 2003.
12. G.E. Hughes and M.J. Cresswell. *A Companion to Modal Logic*. Methuen, 1984.
13. S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, 44(1-2):167–207, 1990.
14. D. Lehmann and M. Magidor. What does a conditional knowledge base entail? *Artificial Intelligence*, 55(1):1–60, 1992.
15. G. L. Pozzato. *Proof Methods for Conditional and Preferential Logics*. PhD thesis, 2007.
16. Y. Shoham. A semantical approach to nonmonotonic logics. *In Proc. of Logics in Computer Science*, pages 275–279, 1987.