# A Tableau Calculus for a Nonmonotonic Extension of the Description Logic *DL-Lite*$_{core}$

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

[1] Dip. di Informatica, U. Piemonte O., Alessandria, Italy
`laura@mfn.unipmn.it`
[2] Dip. Informatica, Univ. di Torino, Italy
`{gliozzi,pozzato}@di.unito.it`
[3] LSIS-UMR CNRS 6168, Marseille, France
`nicola.olivetti@univ-cezanne.fr`

**Abstract.** In this paper we introduce a tableau calculus for a nonmonotonic extension of the low complexity Description Logic *DL-Lite*$_{core}$ of the *DL-Lite* family. The extension, called *DL-Lite*$_c\mathbf{T}_{min}$, can be used to reason about typicality and defeasible properties. The calculus performs a two-phase computation to check whether a query is minimally entailed from the initial knowledge base. It is sound, complete and terminating. Furthermore, it is a decision procedure for *DL-Lite*$_c\mathbf{T}_{min}$ knowledge bases, whose complexity matches the known results for the logic, namely that entailment is in $\Pi^p_2$.

## 1 Introduction

The interest for efficient reasoning in Description Logics (DLs) has greatly increased in the last years. This is motivated by the fact that several applications are supposed to handle ontologies comprising thousands of elements, as it happens for instance in the Semantic Web, where ontologies are used to formalize concepts of very large Web repositories. For these applications, it is crucial to dispose of efficient reasoning algorithms. The current reasoning systems (capable of handling sometimes very expressive DLs) are inadequate, as they offer good performances in practice only for relatively small ontologies, but they cannot handle very large data sets. This has led to the study of *low complexity* DLs, such as the logics of the $\mathcal{EL}$ family and the ones of the *DL-Lite* family. The logics of the $\mathcal{EL}$ family [1] are relevant in the bio-medical domain: for instance, medical terminologies, such as the GALEN Medical Knowledge Base, the Systemized Nomenclature of Medicine (SNOMED), and the Gene Ontology used in bioinformatics, can be formalized in small extensions of $\mathcal{EL}$. The logics of the *DL-Lite* family [7] are specifically tailored for effective query answering over DL knowledge bases containing a large amount of data.

Since DLs are used to represent classes and their properties, a nonmonotonic mechanism is wished to express defeasible inheritance of prototypical properties. Nonmonotonic extensions of DLs have been actively investigated since the early 90s, [15, 5, 2, 3, 9, 13, 10, 8]. A simple but powerful nonmonotonic extension of DL is proposed in [13, 12, 10]: in this approach "typical" or "normal" properties can be directly specified by means of a "typicality" operator $\mathbf{T}$ enriching the underlying DL.

In *DL-Lite*$_c\mathbf{T}_{min}$ [12], one can consistently express defeasible inclusions and exceptions such as: typically elephants live in the Savannah, but elephants with a teacher normally do not live in the Savannah. In *DL-Lite*$_c\mathbf{T}_{min}$ the previous inclusions can be formalized as follows: $\mathbf{T}(\mathit{Elephant}) \sqsubseteq \mathit{LiveInTheSavannah}$, $\mathit{TrainedElephant} \sqsubseteq \mathit{Elephant}$, $\mathit{TrainedElephant} \sqsubseteq \exists \mathit{HasTeacher}.\top$, $\mathbf{T}(\mathit{TrainedElephant}) \sqsubseteq \neg \mathit{LiveInTheSavannah}$. The operator $\mathbf{T}$ is nonmonotonic, in the sense that from $\mathit{TrainedElephant} \sqsubseteq \mathit{Elephant}$ it does not follow that $\mathbf{T}(\mathit{TrainedElephant}) \sqsubseteq \mathbf{T}(\mathit{Elephant})$, and indeed $\mathbf{T}(\mathit{TrainedElephant})$ and $\mathbf{T}(\mathit{Elephant})$ can have different properties. Notice that without the $\mathbf{T}$ operator we would not be able to say that $\mathit{Elephant} \sqsubseteq \mathit{LiveInTheSavannah}$ and $\mathit{TrainedElephant} \sqsubseteq \neg\mathit{LiveInTheSavannah}$ without saying that $\mathit{TrainedElephant}$ is empty.

The typicality operator $\mathbf{T}$ is characterised by the core properties of nonmonotonic reasoning axiomatized by *preferential logic* [14]. $\mathbf{T}$ allows to consistently express inclusions as the ones above, but it is not sufficient to perform nonmonotonic inferences. This is why, *DL-Lite*$_c\mathbf{T}_{min}$ comprises a nonmonotonic mechanism based on a minimal model semantics. In the absence of information to the contrary, this mechanism allows to infer that a given individual is a typical instance of the most specific concept it belongs to. For instance, from the KB above, if we only know that $\mathit{dumbo}$ is an elephant, then we would conclude that it is a typical elephant and lives in the Savannah, whereas if we knew that it is a trained elephant, then we would conclude that he is a typical trained elephant, and it does not live in the Savannah. Notice that we would not be able to do these inferences without the nonmonotonic mechanism that has been added.

While it has been shown that adding the typicality operator with its minimal-model semantics to other standard DLs, such as $\mathcal{ALC}$ and $\mathcal{EL}^{\bot}$, leads to a very high complexity (in particular, query entailment in $\mathcal{EL}^{\bot}\mathbf{T}_{min}$ is EXPTIME hard [12]), it has been recently shown that query entailment in *DL-Lite*$_c\mathbf{T}_{min}$ is in $\Pi^p_2$ [12]. This result is analogous to the one for *circumscribed DL-Lite*$_{core}$ KBs [3].

Proof methods for nonomonotonic extensions of $\mathcal{ALC}$ and $\mathcal{EL}^{\bot}$, called $\mathcal{ALC}+\mathbf{T}_{min}$ and $\mathcal{EL}^{\bot}\mathbf{T}_{min}$, have been introduced in [10] and [11], respectively, whereas no proof methods for the logic *DL-Lite*$_c\mathbf{T}_{min}$ have been proposed. In order to fill this gap, in this paper we present a tableau calculus for deciding minimal entailment in *DL-Lite*$_c\mathbf{T}_{min}$. Similarly to $\mathcal{ALC}+\mathbf{T}_{min}$ and $\mathcal{EL}^{\bot}\mathbf{T}_{min}$, we present a two-phase calculus: in the first phase, candidate models (complete open branches) falsifying the given query are generated, in the second phase the minimality of candidate models is checked by means of an auxiliary tableau construction. The latter tries to build a model which is "more preferred" than the candidate one: if it fails (being closed) the candidate model is minimal, otherwise it is not. As we will discuss in Section 3, there are several differences between the calculus for *DL-Lite*$_c\mathbf{T}_{min}$ and the ones for $\mathcal{ALC}+\mathbf{T}_{min}$ and $\mathcal{EL}^{\bot}\mathbf{T}_{min}$. As a result, the calculus that we propose here is very simple and does not require any blocking machinery in order to achieve termination. Furthermore, our calculus provides a constructive proof of the upper bound for minimal entailment in *DL-Lite*$_c\mathbf{T}_{min}$.

## 2 The Typicality Operator T and the Logic *DL-Lite*$_c\mathbf{T}_{min}$

The language of *DL-Lite*$_c\mathbf{T}_{min}$ is obtained by adding to *DL-Lite*$_{core}$ the typicality operator $\mathbf{T}$. The intuitive idea is that $\mathbf{T}(C)$ selects the *typical* instances of a concept $C$.

In *DL-Lite*$_c\mathbf{T}_{min}$ we can therefore distinguish between the properties that hold for all instances of concept $C$ ($C \sqsubseteq D$), and those that only hold for the normal or typical instances of $C$ ($\mathbf{T}(C) \sqsubseteq D$). Formally, the *DL-Lite*$_c\mathbf{T}_{min}$ language is defined as follows.

**Definition 1.** *We consider an alphabet of concept names $\mathcal{C}$, of role names $\mathcal{R}$, and of individuals $\mathcal{O}$. Given $A \in \mathcal{C}$ and $r \in \mathcal{R}$, we define*

$$C_L := A \mid \exists R.\top \mid \mathbf{T}(A) \qquad C_R := A \mid \neg A \mid \exists R.\top \mid \neg\exists R.\top \qquad R := r \mid r^-$$

*A* DL-Lite$_c\mathbf{T}_{min}$ *KB is a pair (TBox, ABox). TBox contains a finite set of concept inclusions of the form $C_L \sqsubseteq C_R$. ABox contains assertions of the form $C(a)$ and $r(a,b)$, where $C$ is a $C_L$ or $C_R$ concept, $r \in \mathcal{R}$, and $a, b \in \mathcal{O}$.*

The semantics of *DL-Lite*$_c\mathbf{T}_{min}$ is defined by enriching ordinary models of *DL-Lite*$_{core}$ by a *preference relation* $<$ on the domain, whose intuitive meaning is to compare the "typicality" of individuals: $x < y$, means that $x$ is more typical than $y$. Typical members of a concept $C$, i.e., members of $\mathbf{T}(C)$, are the members $x$ of $C$ that are minimal with respect to this preference relation.

**Definition 2 (Semantics of T).** *A model $\mathcal{M}$ is any structure $\langle \Delta, <, I\rangle$ where $\Delta$ is the domain; $<$ is an irreflexive and transitive relation over $\Delta$ that satisfies the following* Smoothness Condition*: for all $S \subseteq \Delta$, for all $x \in S$, either $x \in Min_<(S)$ or $\exists y \in Min_<(S)$ such that $y < x$, where $Min_<(S) = \{u : u \in S$ and $\nexists z \in S$ s.t. $z < u\}$. Furthermore, $<$ is multilinear: if $u < z$ and $v < z$, then either $u = v$ or $u < v$ or $v < u$. $I$ is the extension function that maps each concept $C$ to $C^I \subseteq \Delta$, and each role $r$ to $r^I \subseteq \Delta^I \times \Delta^I$. For concepts of* DL-Lite$_{core}$, *$C^I$ is defined in the usual way. For the $\mathbf{T}$ operator: $(\mathbf{T}(C))^I = Min_<(C^I)$.*

**Definition 3 (Model satisfying a Knowledge Base).** *Given a model $\mathcal{M}$, $I$ can be extended so that it assigns to each individual $a$ of $\mathcal{O}$ a distinct element $a^I$ of the domain $\Delta$. $\mathcal{M}$ satisfies a KB (TBox,ABox), if it satisfies both its TBox and its ABox, where:*

- *$\mathcal{M}$ satisfies TBox if $\mathcal{M}$ satisifes all its inclusions. An inclusion $C \sqsubseteq D$ is satisifed in $\mathcal{M}$ if $C^I \subseteq D^I$.*
- *$\mathcal{M}$ satisfies ABox if $\mathcal{M}$ satisifes all its formulas $C(a)$ and $r(a,b)$. $\mathcal{M}$ satisifes $C(a)$ if $a^I \in C^I$. $\mathcal{M}$ satisfies $r(a,b)$ if $(a^I, b^I) \in r^I$.*

*We assume the unique name assumption.*

The operator $\mathbf{T}$ [13] is characterized by a set of postulates that are essentially a reformulation of KLM [14] axioms of *preferential logic* **P**. $\mathbf{T}$ has therefore all the "core" properties of nonmonotonic reasoning as it is axiomatised by **P**.

The semantics of the typicality operator can be specified by modal logic. The interpretation of $\mathbf{T}$ can be split into two parts: for any $x$ of the domain $\Delta$, $x \in (\mathbf{T}(C))^I$ just in case (i) $x \in C^I$, and (ii) there is no $y \in C^I$ such that $y < x$. Condition (ii) can be represented by means of an additional modality $\Box$, whose semantics is given by the preference relation $<$ interpreted as an accessibility relation. Observe that by the Smoothness Condition, $\Box$ has the properties of Gödel-Löb modal logic of provability G. The interpretation of $\Box$ in $\mathcal{M}$ is as follows:

$$(\Box C)^I = \{x \in \Delta \mid \text{for every } y \in \Delta, \text{if } y < x \text{ then } y \in C^I\}$$

We immediately get that $x \in (\mathbf{T}(C))^I$ iff $x \in (C \sqcap \Box\neg C)^I$. From now on, we consider $\mathbf{T}(C)$ as an abbreviation for $C \sqcap \Box\neg C$.

In order to perform nonmonotonic inferences, the semantics of *DL-Lite*$_c\mathbf{T}_{min}$ is strenghtened by restricting entailment to a class of minimal (or preferred) models. Intuitively, the idea is to restrict our consideration to models that *minimize the non typical instances of a concept.*

Given a KB, we consider a finite set $\mathcal{L}_{\mathbf{T}}$ of concepts: these are the concepts whose non typical instances we want to minimize. We assume that the set $\mathcal{L}_{\mathbf{T}}$ contains at least all concepts $C$ such that $\mathbf{T}(C)$ occurs in the KB or in the query $F$, where a *query* $F$ is either an assertion $C(a)$, where $C$ is a $C_L$ or $C_R$, or an inclusion relation $C \sqsubseteq D$, where $C$ is a $C_L$ and $D$ is a $C_R$. As we have just said, $x \in C^I$ is typical if $x \in (\Box\neg C)^I$. Minimizing the non typical instances of $C$ therefore means to minimize the objects not satisfying $\Box\neg C$ for $C \in \mathcal{L}_{\mathbf{T}}$. Hence, for a given model $\mathcal{M} = \langle \Delta, <, I\rangle$, we define:

$$\mathcal{M}^{\Box^-}_{\mathcal{L}_{\mathbf{T}}} = \{(x, \neg\Box\neg C) \mid x \not\in (\Box\neg C)^I, \text{ with } x \in \Delta, C \in \mathcal{L}_{\mathbf{T}}\}.$$

**Definition 4 (Preferred and minimal models).** *Given a model $\mathcal{M} = \langle \Delta <, I\rangle$ of a knowledge base KB, and a model $\mathcal{M}' = \langle \Delta', <', I'\rangle$ of KB, we say that $\mathcal{M}$ is preferred to $\mathcal{M}'$ with respect to $\mathcal{L}_{\mathbf{T}}$, and we write $\mathcal{M} <_{\mathcal{L}_{\mathbf{T}}} \mathcal{M}'$, if (i) $\Delta = \Delta'$, (ii) $a^I = a^{I'}$ for all $a \in \mathcal{O}$, and (iii) $\mathcal{M}^{\Box^-}_{\mathcal{L}_{\mathbf{T}}} \subset \mathcal{M}'^{\Box^-}_{\mathcal{L}_{\mathbf{T}}}$. A model $\mathcal{M}$ is a* minimal model *for KB (with respect to $\mathcal{L}_{\mathbf{T}}$) if it is a model of KB and there is no other model $\mathcal{M}'$ of KB such that $\mathcal{M}' <_{\mathcal{L}_{\mathbf{T}}} \mathcal{M}$.*

**Definition 5 (Minimal Entailment in *DL-Lite*$_c\mathbf{T}_{min}$).** *A query $F$ is minimally entailed in* DL-Lite$_c\mathbf{T}_{min}$ *by KB with respect to $\mathcal{L}_{\mathbf{T}}$ if all models of KB, that are minimal with respect to $\mathcal{L}_{\mathbf{T}}$, satisfy $F$. We write KB $\models_{\text{DL-Lite}_c\mathbf{T}_{min}} F$.*

In [12], a small model construction allowed us to prove the following complexity result:

**Theorem 1 (Complexity of entailment in *DL-Lite*$_c\mathbf{T}_{min}$, Theorem 4.6 in [12]).** *The problem of deciding whether KB $\models_{\text{DL-Lite}_c\mathbf{T}_{min}} F$ is in $\Pi^p_2$.*

## 3 A Tableau Calculus for *DL-Lite*$_c\mathbf{T}_{min}$

In this section we present a tableau calculus $\mathcal{TAB}^{Lite_c\mathbf{T}}_{min}$ for deciding whether a query $F$ is minimally entailed from a KB in the logic *DL-Lite*$_c\mathbf{T}_{min}$. As mentioned in the Introduction, the calculus is inspired to the calculi for $\mathcal{ALC} + \mathbf{T}_{min}$ and $\mathcal{EL}^{\perp}\mathbf{T}_{min}$ introduced in [10] and [11] but it contains a few significant differences in order to obtain a calculus whose complexity matches the result of Theorem 1 above.

The calculus $\mathcal{TAB}^{Lite_c\mathbf{T}}_{min}$ performs a two-phase computation. In the first phase, a tableau calculus, called $\mathcal{TAB}^{Lite_c\mathbf{T}}_{PH1}$, simply verifies whether KB $\cup \{\neg F\}$ is satisfiable in a model, building candidate models. In the second phase another tableau calculus, called $\mathcal{TAB}^{Lite_c\mathbf{T}}_{PH2}$, checks whether the candidate models found in the first phase are *minimal* models of KB, i.e. for each open saturated branch of the first phase, $\mathcal{TAB}^{Lite_c\mathbf{T}}_{PH2}$ tries to build a model of KB which is preferred to the candidate model w.r.t. Definition 4. The whole procedure $\mathcal{TAB}^{Lite_c\mathbf{T}}_{min}$ is formally defined in Definition 8.

Before examining the calculi in detail, we provide some additional formal definitions. First, the negation of a query $\neg F$ is defined as follows: (i) if $F \equiv C(a)$, then $\neg F \equiv (\neg C)(a)$; (ii) if $F \equiv C \sqsubseteq D$, then $\neg F \equiv C(x), (\neg D)(x)$, where $x$ does not occur in KB.

$\mathcal{TAB}_{min}^{\textit{Lite}_c\mathbf{T}}$ makes use of labels, which are denoted with $x, y, z, \ldots$. Labels represent either a variable or an individual of the ABox. These labels occur in *constraints*. A *constraint* (or *labelled* formula) is a syntactic entity of the form either $x \xrightarrow{r} y$ or $x : C$, where $x, y$ are labels, $r$ is a role and $C$ is either a concept or the negation of a concept of *DL-Lite*$_c\mathbf{T}_{min}$ or it has the form $\Box\neg D$ or $\neg\Box\neg D$, where $D$ is a concept. Finally, given a set of constraints $S$ and a role $r \in \mathcal{R}$, we define $r(S) = \{x \xrightarrow{r} y \mid x \xrightarrow{r} y \in S\}$.

## 3.1 First Phase: The Tableaux Calculus $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$

Let us first define the basic notions of a tableau system in $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$. A tableau of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ is a tree whose nodes are pairs $\langle S \mid U\rangle$. $S$ is a set of constraints, whereas $U$ contains formulas of the form $C \sqsubseteq D^L$, representing inclusions $C \sqsubseteq D$ of the TBox. $L$ is a list of labels, used in order to ensure the termination of the tableau calculus. A branch is a sequence of nodes $\langle S_1 \mid U_1\rangle, \langle S_2 \mid U_2\rangle, \ldots, \langle S_n \mid U_n\rangle \ldots$, where each node $\langle S_i \mid U_i\rangle$ is obtained from its immediate predecessor $\langle S_{i-1} \mid U_{i-1}\rangle$ by applying a rule of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$, having $\langle S_{i-1} \mid U_{i-1}\rangle$ as the premise and $\langle S_i \mid U_i\rangle$ as one of its conclusions. A branch is closed if one of its nodes is an instance of a (Clash) axiom, otherwise it is open. A tableau is closed if all its branches are closed.

In order to check the satisfiability of a KB, we build its *corresponding constraint system* $\langle S \mid U\rangle$, and we check its satisfiability.

**Definition 6 (Corresponding constraint system).** *Given a knowledge base* KB=(*TBox,ABox*)*, we define its* corresponding constraint system $\langle S \mid U\rangle$ *as follows:*

- $S = \{a : C \mid C(a) \in ABox\} \cup \{a \xrightarrow{r} b \mid r(a, b) \in ABox\}$
- $U = \{C \sqsubseteq D^{\emptyset} \mid C \sqsubseteq D \in TBox\}$

**Definition 7 (Model satisfying a constraint system).** *Let $\mathcal{M} = \langle \Delta, I, <\rangle$ be a model as defined in Definition 2. We define a function $\alpha$ which assigns to each variable of $\mathcal{V}$ an element of $\Delta$, and assigns every individual $a \in \mathcal{O}$ to $a^I \in \Delta$. $\mathcal{M}$ satisfies a constraint $F$ under $\alpha$, written $\mathcal{M} \models_\alpha F$, as follows: (i) $\mathcal{M} \models_\alpha x : C$ iff $\alpha(x) \in C^I$; (ii) $\mathcal{M} \models_\alpha x \xrightarrow{r} y$ iff $(\alpha(x), \alpha(y)) \in r^I$. A constraint system $\langle S \mid U\rangle$ is satisfiable if there is a model $\mathcal{M}$ and a function $\alpha$ such that $\mathcal{M}$ satisfies every constraint in $S$ under $\alpha$ and that, for all $C \sqsubseteq D^L \in U$, we have that $C^I \subseteq D^I$.*

**Proposition 1.** *Given a KB=(TBox,ABox), it is satisfiable if and only if its corresponding constraint system $\langle S \mid U\rangle$ is satisfiable.*

To verify the satisfiability of KB $\cup \{\neg F\}$, we use $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ to check the satisfiability of the constraint system $\langle S \mid U\rangle$ obtained by adding the constraint corresponding to $\neg F$ to $S'$, where $\langle S' \mid U\rangle$ is the corresponding constraint system of KB. To this purpose, the rules of the calculus $\mathcal{TAB}_{PH1}^{\mathit{Lite}_c\mathbf{T}}$ are applied until either a contradiction is generated (Clash) or no other rule is applicable. Given a node $\langle S \mid U \rangle$, for each inclusion $C \sqsubseteq D^L \in U$ and for each label $x$ occurring in the tableau, we add to $S$ the constraint $x : \neg C \sqcup D$: we refer to this mechanism as *unfolding*. As mentioned, each inclusion $C \sqsubseteq D$ is equipped with a list $L$ of labels in which it has been unfolded in the current branch. This is needed to avoid multiple unfolding of the same inclusion by using the same label, generating infinite branches.

The calculus $\mathcal{TAB}_{PH1}^{\mathit{Lite}_c\mathbf{T}}$ is significantly different in four respects from the calculi for $\mathcal{ALC} + \mathbf{T}_{min}$ and $\mathcal{EL}^{\bot}\mathbf{T}_{min}$. We try to explain such differences in detail.

1. The rule $(\exists^+)$ is split in the following two rules:

$$\frac{\langle S, x : \exists r.\top \mid U \rangle}{\langle S, x \xrightarrow{r} y \mid U \rangle \quad \langle S, x \xrightarrow{r} y_1 \mid U \rangle \cdots \langle S, x \xrightarrow{r} y_m \mid U \rangle}\,(\exists^+)_1^r \qquad \begin{array}{r}\text{if } r(S) = \emptyset\\ y \text{ new}\\ \text{if } y_1, \ldots, y_m \text{ are all the labels occurring in } S\end{array}$$

$$\frac{\langle S, x : \exists r.\top \mid U \rangle}{\langle S, x \xrightarrow{r} y_1 \mid U \rangle \ \cdots \ \langle S, x \xrightarrow{r} y_m \mid U \rangle}\,(\exists^+)_2^r \qquad \begin{array}{r}\text{if } r(S) \neq \emptyset\\ \text{if } y_1, \ldots, y_m \text{ are all the labels occurring in } S\end{array}$$

The split of the $(\exists^+)$ in the two rules above reflects the main idea of the construction of a small model at the base of Theorem 1. Such small model theorem essentially shows that *DL-Lite*$_c\mathbf{T}_{min}$ KBs can have small models in which all existentials $\exists r.\top$ occurring in KB are satisfied in the model by a single witness $y$. In the calculus we use the same idea: when the rule $(\exists^+)_1^r$ is applied to a formula $x : \exists r.\top$, it introduces a new label $y$ and the constraint $x \xrightarrow{r} y$ only when there is no other previous constraint $u \xrightarrow{r} v$ in $S$. Otherwise, rule $(\exists^+)_2^r$ is applied and it introduces $x \xrightarrow{r} y$, with $y$ already occurring in the branch. As a consequence, $(\exists^+)_2^r$ does not introduce any new label in the branch whereas $(\exists^+)_1^r$ only introduces a new label $y$ for each role $r$ occurring in the initial KB in some $\exists r.\top$ or $\exists r^-.\top$, and no blocking machinery is needed to ensure termination.

2. In order to keep into account inverse roles, two further rules for existential formulas are introduced:

$$\frac{\langle S, x : \exists r^-.\top \mid U \rangle}{\langle S, y \xrightarrow{r} x \mid U \rangle \ \langle S, y_1 \xrightarrow{r} x \mid U \rangle \cdots \langle S, y_m \xrightarrow{r} x \mid U \rangle}\,(\exists^+)_1^{r^-} \qquad \begin{array}{r}\text{if } r(S) = \emptyset\\ y \text{ new}\\ \text{if } y_1, \ldots, y_m \text{ are all the labels occurring in } S\end{array}$$

$$\frac{\langle S, x : \exists r^-.\top \mid U \rangle}{\langle S, y_1 \xrightarrow{r} x \mid U \rangle \cdots \langle S, y_m \xrightarrow{r} x \mid U \rangle}\,(\exists^+)_2^{r^-} \qquad \begin{array}{r}\text{if } r(S) \neq \emptyset\\ \text{if } y_1, \ldots, y_m \text{ are all the labels occurring in } S\end{array}$$

These rules work similarly to $(\exists^+)_1^r$ and $(\exists^+)_2^r$ in order to build a branch representing a small model: when the rule $(\exists^+)_1^{r^-}$ is applied to a formula $x : \exists r^-.\top$, it introduces a new label $y$ and the constraint $y \xrightarrow{r} x$ only when there is no other constraint $u \xrightarrow{r} v$ in $S$. Otherwise, since a constraint $y \xrightarrow{r} u$ has been already introduced in that branch, $y \xrightarrow{r} x$ is added to the conclusion of the rule.

3. The calculus $\mathcal{TAB}_{PH1}^{\mathit{Lite}_c\mathbf{T}}$ does not need the rule $(\exists^-)$ of $\mathcal{TAB}_{min}^{\mathcal{ALC}+\mathbf{T}}$. Indeed, the only negated existential formulas that can occur in a branch have the form (i) $x : \neg\exists r.\top$ or (ii) $x : \neg\exists r^-.\top$. (i) means that $x$ has no relationships with other individuals via the role $r$, i.e. we need to detect a contradiction if both (i) and, for some $y$, $x \xrightarrow{r} y$ belong to the same branch, in order to mark the branch as closed. The clash condition (Clash)$_r$ is added to the calculus $\mathcal{TAB}_{PH1}^{\mathit{Lite}_c\mathbf{T}}$ in order to detect such a situation. Analogously, (ii) means that there is no $y$ such that $y$ is related to $x$ by means of $r$, then (Clash)$_{r^-}$ is

introduced in order to close a branch containing both (ii) and, for some $y$, a constraint $y \xrightarrow{r} x$. The clash conditions (Clash)$_r$ and (Clash)$_{r^-}$ are as follows:

$$\langle S, x \xrightarrow{r} y, x : \neg\exists r.\top \mid U\rangle \ (\text{Clash})_r \qquad \langle S, y \xrightarrow{r} x, x : \neg\exists r^-.\top \mid U\rangle \ (\text{Clash})_{r^-}$$

4. In order to build multilinear models of Definition 2, the calculus adopts a strengthened version of the rule $(\Box^-)$ used in $\mathcal{TAB}^{\mathcal{ALC}+\mathbf{T}}_{min}$ [10]. We write $\overline{S}$ as an abbreviation for $S, x : \neg\Box\neg C_1, \dots, x : \neg\Box\neg C_n$. Moreover, we define $S^M_{x\rightarrow y} = \{y : \neg D, y : \Box\neg D \mid x : \Box\neg D \in S\}$ and, for $k = 1, 2, \dots, n$, we define $\overline{S}^{\Box^{-k}}_{x\rightarrow y} = \{y : \neg\Box\neg C_j \sqcup C_j \mid x : \neg\Box\neg C_j \in \overline{S} \wedge j \neq k\}$. The strengthened rule $(\Box^-)$ is also adopted in the calculus for $\mathcal{EL}^{\bot}\mathbf{T}_{min}$ in [11] and is as follows:

$$\frac{\langle S, x : \neg\Box\neg C_1, \dots, \neg\Box\neg C_n \mid U\rangle}{\langle S, y : C_k, y : \Box\neg C_k, S^M_{x\rightarrow y}, \overline{S}^{\Box^{-k}}_{x\rightarrow y} \mid U\rangle \ \ \langle S, y_1 : C_k, y_1 : \Box\neg C_k, S^M_{x\rightarrow y_1}, \overline{S}^{\Box^{-k}}_{x\rightarrow y_1} \mid U\rangle \cdots \langle S, y_m : C_k, y_m : \Box\neg C_k, S^M_{x\rightarrow y_m}, \overline{S}^{\Box^{-k}}_{x\rightarrow y_m} \mid U\rangle} \ (\Box^-)$$
$$\begin{array}{r} y \text{ new} \\ \text{if } y_1, \dots, y_m \text{ are all the labels occurring in } S, y_1 \neq x, \dots, y_m \neq x \\ \forall k = 1, 2, \dots, n \end{array}$$

Rule $(\Box^-)$ contains: - $n$ branches, one for each $x : \neg\Box\neg C_k$ in $\overline{S}$; in each branch a *new* typical $C_k$ individual $y$ is introduced (i.e. $y : C_k$ and $y : \Box\neg C_k$ are added), and for all other $x : \neg\Box\neg C_j$, either $y : C_j$ holds or the formula $y : \neg\Box\neg C_j$ is recorded; - other $n \times m$ branches, where $m$ is the number of labels occurring in $S$, one for each label $y_i$ and for each $x : \neg\Box\neg C_k$ in $\overline{S}$; in these branches, a given $y_i$ is chosen as a typical instance of $C_k$, that is to say $y_i : C_k$ and $y_i : \Box\neg C_k$ are added, and for all other $x : \neg\Box\neg C_j$, either $y_i : C_j$ holds or the formula $y_i : \neg\Box\neg C_j$ is recorded. This rule is sound with respect to multilinear models. The advantage of this rule over the $(\Box^-)$ rule in the calculus $\mathcal{TAB}^{\mathcal{ALC}+\mathbf{T}}_{min}$ for $\mathcal{ALC} + \mathbf{T}_{min}$ is that all the negated box formulas labelled by $x$ are treated in one step, introducing at most one new label $y$ in

$$\langle S, x : C, x : \neg C \mid U\rangle \ (\text{Clash}) \qquad \langle S, x \xrightarrow{r} y, x : \neg\exists r.\top \mid U\rangle \ (\text{Clash})_r \qquad \langle S, y \xrightarrow{r} x, x : \neg\exists r^-.\top \mid U\rangle \ (\text{Clash})_{r^-}$$

$$\frac{\langle S, x : \exists r.\top \mid U\rangle}{\langle S, x \xrightarrow{r} y \mid U\rangle \ \langle S, x \xrightarrow{r} y_1 \mid U\rangle \cdots \langle S, x \xrightarrow{r} y_m \mid U\rangle} \ (\exists^+)^r_1 \qquad \begin{array}{r} \text{if } r(S) = \emptyset \\ y \text{ new} \\ \text{if } y_1, \dots, y_m \text{ are all the labels occurring in } S \end{array}$$

$$\frac{\langle S, x : \exists r.\top \mid U\rangle}{\langle S, x \xrightarrow{r} y_1 \mid U\rangle \cdots \langle S, x \xrightarrow{r} y_m \mid U\rangle} \ (\exists^+)^r_2 \qquad \begin{array}{r} \text{if } r(S) \neq \emptyset \\ \text{if } y_1, \dots, y_m \text{ are all the labels occurring in } S \end{array}$$

$$\frac{\langle S, x : \mathbf{T}(C) \mid U\rangle}{\langle S, x : C, x : \Box\neg C \mid U\rangle} \ (\mathbf{T}^+)$$

$$\frac{\langle S, x : \neg\mathbf{T}(C) \mid U\rangle}{\langle S, x : \neg C \mid U\rangle \ \langle S, x : \neg\Box\neg C \mid U\rangle} \ (\mathbf{T}^-)$$

$$\frac{\langle S \mid U\rangle}{\langle S, x : \Box\neg C \mid U\rangle \ \ \langle S, x : \neg\Box\neg C \mid U\rangle} \ (cut) \qquad \begin{array}{r} \text{if } x : \neg\Box\neg C \notin S \text{ and } x : \Box\neg C \notin S \\ C \in \mathcal{L}_{\mathbf{T}} \\ x \text{ occurs in } S \end{array}$$

$$\frac{\langle S \mid U, C \sqsubseteq D^L\rangle}{\langle S, x : \neg C \sqcup D \mid U, C \sqsubseteq D^{L,x}\rangle} \ (\text{Unfold}) \qquad \text{if } x \text{ occurs in } S \text{ and } x \notin L$$

$$\frac{\langle S, x : \exists r^-.\top \mid U\rangle}{\langle S, y \xrightarrow{r} x \mid U\rangle \ \langle S, y_1 \xrightarrow{r} x \mid U\rangle \dots \langle S, y_m \xrightarrow{r} x \mid U\rangle} \ (\exists^+)^{r^-}_1 \qquad \begin{array}{r} \text{if } r(S) = \emptyset \\ y \text{ new} \\ \text{if } y_1, \dots, y_m \text{ are all the labels occurring in } S \end{array}$$

$$\frac{\langle S, x : \exists r^-.\top \mid U\rangle}{\langle S, y_1 \xrightarrow{r} x \mid U\rangle \cdots \langle S, y_m \xrightarrow{r} x \mid U\rangle} \ (\exists^+)^{r^-}_2 \qquad \begin{array}{r} \text{if } r(S) \neq \emptyset \\ \text{if } y_1, \dots, y_m \text{ are all the labels occurring in } S \end{array}$$

$$\frac{\langle S, x : C \sqcup D \mid U\rangle}{\langle S, x : C \mid U\rangle \ \ \langle S, x : D \mid U\rangle} \ (\sqcup^+)$$

$$\frac{\langle S, x : \neg\Box\neg C_1, \dots, \neg\Box\neg C_n \mid U\rangle}{\langle S, y : C_k, y : \Box\neg C_k, S^M_{x\rightarrow y}, \overline{S}^{\Box^{-k}}_{x\rightarrow y} \mid U\rangle \ \ \langle S, y_1 : C_k, y_1 : \Box\neg C_k, S^M_{x\rightarrow y_1}, \overline{S}^{\Box^{-k}}_{x\rightarrow y_1} \mid U\rangle \cdots \langle S, y_m : C_k, y_m : \Box\neg C_k, S^M_{x\rightarrow y_m}, \overline{S}^{\Box^{-k}}_{x\rightarrow y_m} \mid U\rangle} \ (\Box^-)$$
$$\begin{array}{r} y \text{ new} \\ \text{if } y_1, \dots, y_m \text{ are all the labels occurring in } S, y_1 \neq x, \dots, y_m \neq x \\ \forall k = 1, 2, \dots, n \end{array}$$

**Fig. 1.** The calculus $\mathcal{TAB}^{\mathit{Lite}_c\mathbf{T}}_{PH1}$

the conclusions. Notice that in order to keep $\overline{S}$ readable, we have used $\sqcup$. This is the reason why our calculi contain the rule $(\sqcup^+)$, even if this constructor does not belong to *DL-Lite*$_c\mathbf{T}_{min}$.

The rules of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ are presented in Figure 1. Rules $(\exists^+)_1^r$, $(\exists^+)_1^{r^-}$ and $(\Box^-)$ are called *dynamic* since they can introduce a new variable in their conclusions. The other rules are called *static*. We do not need any extra rule for the positive occurrences of the $\Box$ operator, since these are taken into account by the computation of $S^M_{x\to y}$ of $(\Box^-)$. The $(cut)$ rule ensures that, given any concept $C \in \mathcal{L}_\mathbf{T}$, an open saturated branch built by $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ contains either $x : \Box\neg C$ or $x : \neg\Box\neg C$ for each label $x$: this is needed in order to allow $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ to check the minimality of the model corresponding to the open branch.

The rules of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ are applied with the following standard strategy that takes into account the order $\prec$ of insertion of the labels in the branch: as in [6], if $y$ is introduced in the tableau, then $x \prec y$ for all labels $x$ that are already in the tableau.

*Standard strategy*: 1. apply a rule to a label $x$ only if no rule is applicable to a label $y$ such that $y \prec x$; 2. apply dynamic rules only if no static rule is applicable.

The above strategy imposes that the static rule $(cut)$ is applied to a label $x$ before $(\Box^-)$ is applied to the same $x$. This has two benefits. First, it guarantees that $(\Box^-)$ can be applied only once to each $x$. As it will become clear from the proof of Theorem 5, this has an impact on the overall complexity of the calculus. Indeed, no other $x : \neg\Box\neg C$ can be introduced after the application of $(\Box^-)$ to $x$, and no further application of $(\Box^-)$ is needed to take into account the newly introduced negated box formula. This is a consequence of the fact that each $x : \neg\Box\neg C$ that will ever occur on the branch has been introduced by $(cut)$ (indeed if $x : \neg\Box\neg C$ has not been introduced by $(cut)$, then $x : \Box\neg C$ has been introduced, which prevents $x : \neg\Box\neg C$ to be introduced at a later stage). Second, since $(cut)$ introduces all possible positive boxed $x : \Box\neg C$ that will ever appear on the branch, the strategy guarantees that for all these formulas $y : \neg C$ and $y : \Box\neg C$ hold for each $y$ introduced by an application of $(\Box^-)$ to $x$. In this way, we do not need an extra rule $(\Box^+)$.

The calculus $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ is sound and complete.

**Theorem 2 (Soundness of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$).** *If KB* $\not\models_{\text{DL-Lite}_c\mathbf{T}_{min}} F$*, then the tableau for the constraint system corresponding to KB* $\cup \{\neg F\}$ *contains an open saturated branch, which is satisfiable (via an injective assignment from labels to domain elements) in a minimal model of KB.*

**Theorem 3 (Completeness of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$).** *Given a constraint system* $\langle S \mid U\rangle$*, if it is unsatisfiable, then it has a closed tableau in* $\mathcal{TAB}_{PH1}^{\text{Lite}_c\mathbf{T}}$.

Let us now analyze termination of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$.

**Theorem 4 (Termination of $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$).** *Let* $\langle S \mid U\rangle$ *be the corresponding constraint system of a KB. Any tableau generated by* $\mathcal{TAB}_{PH1}^{\text{Lite}_c\mathbf{T}}$ *for* $\langle S \mid U\rangle$ *is finite.*

*Proof.* (Sketch) The following facts allow us to prove termination:

– Rules cannot be reapplied over the same formula without any control. Indeed, the only rule copying its principal formula in its conclusion is (Unfold), but this rule can be applied to $\langle S \mid U, C \sqsubseteq D^L\rangle$ by using the label $x$ only if it has not yet been applied to $x$ in the current branch (i.e., $x$ does not belong to $L$).

– Only finitely-many labels can be introduced on a branch. Roughly speaking, the $(\exists^+)^r_1$ rule introduces at most one new label $y$ for each role $r$ belonging to the initial KB. The same holds for the rule $(\exists^+)^{r^-}_1$. Moreover, thanks to the properties of $\Box$, it can be shown that the interplay between rules $(\mathbf{T}^-)$ and $(\Box^-)$ does not generate branches containing infinitely-many labels. Intuitively, the application of $(\Box^-)$ to $x : \neg\Box\neg C, x : \neg\Box\neg C_1, \ldots, x : \neg\Box\neg C_k$ adds $y : \Box\neg C$ to the conclusion, so that $(\mathbf{T}^-)$ can no longer consistently introduce $y : \neg\Box\neg C$.
– The $(cut)$ rule does not affect termination, since it is applied only to the finitely many formulas belonging to $\mathcal{L}_{\mathbf{T}}$. ■

Let us conclude this section by estimating the complexity of $\mathcal{TAB}^{\mathit{Lite}_c\mathbf{T}}_{PH1}$. Let $n$ be the size of the initial KB, i.e. the length of the string representing KB, and let $\langle S \mid U\rangle$ its corresponding constraint system. We assume that the size of $F$ and $\mathcal{L}_{\mathbf{T}}$ is $O(n)$.

**Theorem 5 (Complexity of Phase 1).** *Given a KB and a query F, the problem of checking whether KB* $\cup \{\neg F\}$ *is satisfiable is in* NP.

*Proof.* (Sketch) The calculus builds a tableau for $\langle S \mid U\rangle$ whose branches's size is $O(n)$. This immediately follows from the fact that dynamic rules generate at most $O(n)$ labels in a branch. This is obvious for rules $(\exists^+)^r_1$ and $(\exists^+)^{r^-}_1$. Concerning $(\Box^-)$, consider a branch generated by its application to a constraint system $\langle S, x : \neg\Box\neg C_1 \ldots, x : \neg\Box\neg C_n \mid U\rangle$. In the worst case, a new label $y_1$ is introduced. Suppose also that the branch under consideration is the one containing $y_1 : C_1$ and $y_1 : \Box\neg C_1$. The $(\Box^-)$ rule can then be applied to formulas $y_1 : \neg\Box\neg C_k$, introducing also a further new label $y_2$. However, by the presence of $y_1 : \Box\neg C_1$, the rule $(\Box^-)$ can no longer consistently introduce $y_2 : \neg\Box\neg C_1$, since $y_2 : \Box\neg C_1 \in S^M_{y_1\rightarrow y_2}$. Therefore, once $(\Box^-)$ is applied to $\neg\Box\neg C_1 \ldots \neg\Box\neg C_n$ in $x$, this application generates (at most) one new world $y_1$ that labels (at most) $n-1$ negated boxed formulas. A further application of $(\Box^-)$ to $\neg\Box\neg C_1 \ldots \neg\Box\neg C_{n-1}$ in $y_1$ generates (at most) one new world $y_2$ that labels (at most) $n-2$ negated boxed formulas, and so on. Overall, at most $O(n)$ new labels are introduced by $(\Box^-)$ in each branch. For each of these labels, static rules apply at most $O(n)$ times. Therefore, the length of the tableau branch built by the strategy is $O(n^2)$. Last, to test that a node is an instance of a (Clash) axiom has at most complexity polynomial in $n$. The same for detecting (Clash)$_r$ and (Clash)$_{r^-}$. Indeed, a node contains a polynomial number of labels, $O(n)$ roles, (hence) a polynomial number of formulas $x : \neg\exists r.\top$ (or $x : \neg\exists r^-.\top$), as well as a polynomial number of formulas $x \xrightarrow{r} y$. ■

## 3.2 The Tableaux Calculus $\mathcal{TAB}^{\mathit{Lite}_c\mathbf{T}}_{PH2}$

Let us now introduce the calculus $\mathcal{TAB}^{\mathit{Lite}_c\mathbf{T}}_{PH2}$ which, for each open saturated branch **B** built by $\mathcal{TAB}^{\mathit{Lite}_c\mathbf{T}}_{PH1}$, verifies whether it represents a minimal model of the KB. First, given an open saturated branch **B** of a tableau built from $\mathcal{TAB}^{\mathit{Lite}_c\mathbf{T}}_{PH1}$, we define $\mathcal{D}(\mathbf{B})$ as the set of labels occurring on **B** and $\mathbf{B}^{\Box^-}$ as the set of formulas $x : \neg\Box\neg C$ occurring in **B**, i.e. $\mathbf{B}^{\Box^-} = \{x : \neg\Box\neg C \mid x : \neg\Box\neg C$ occurs in $\mathbf{B}\}$.

A tableau of $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$ is a tree whose nodes are tuples of the form $\langle S \mid U \mid K\rangle$, where $S$ and $U$ are defined as in $\mathcal{TAB}_{PH1}^{\mathit{Lite}_c\mathbf{T}}$, whereas $K$ contains formulas of the form $x : \neg\Box\neg C$, with $C \in \mathcal{L}_{\mathbf{T}}$. The basic idea of $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$ is as follows. Given an open saturated branch **B** built by $\mathcal{TAB}_{PH1}^{\mathit{Lite}_c\mathbf{T}}$ and corresponding to a model $\mathcal{M}^{\mathbf{B}}$ of KB $\cup \{\neg F\}$, $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$ checks whether $\mathcal{M}^{\mathbf{B}}$ is a minimal model of KB by trying to build a model of KB which is preferred to $\mathcal{M}^{\mathbf{B}}$. To this purpose, it keeps track (in $K$) of the negated box used in **B** ($\mathbf{B}^{\Box^-}$) in order to check whether it is possible to build a model of KB containing less negated box formulas. The tableau built by $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$ closes if it is not possible to build a model smaller than $\mathcal{M}^{B}$, it remains open otherwise. Since by Definition 4 two models can be compared only if they have the same domain, $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$ tries to build an open saturated branch containing all the labels appearing on **B**, i.e. those in $\mathcal{D}(\mathbf{B})$. To this aim, the dynamic rules use labels in $\mathcal{D}(\mathbf{B})$ instead of introducing new ones in their conclusions.

The rules of $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$ are shown in Figure 2. The rules $(\exists^+)^r$ and $(\exists^+)^{r^-}$ introduce $x \xrightarrow{r} y$ and $y \xrightarrow{r} x$, respectively, where $y \in \mathcal{D}(\mathbf{B})$, instead of $y$ being a new label. The choice of the label $y$ introduces a branching in the tableau construction. The rule (Unfold) is applied to *all the labels of* $\mathcal{D}(\mathbf{B})$ (and not only to those appearing in the branch). The rule $(\Box^-)$ is applied to a node $\langle S, x : \neg\Box\neg C_1, \ldots, x : \neg\Box\neg C_n \mid U \mid K\rangle$, when $\{x : \neg\Box\neg C_1, \ldots, x : \neg\Box\neg C_n\} \subseteq K$, i.e. when the negated box formulas $x : \neg\Box\neg C_i$ also belong to the open branch **B**. Even in this case, the rule introduces a branch on the choice of the individual $y_i \in \mathcal{D}(\mathbf{B})$ to be used in the conclusion. In case a tableau node has the form $\langle S, x : \neg\Box\neg C \mid U \mid K\rangle$, and $x : \neg\Box\neg C \notin K$, then

$$\langle S, x : C, x : \neg C \mid U \mid K\rangle \text{ (Clash)} \qquad \langle S, x \xrightarrow{r} y, x : \neg\exists r.\top \mid U \mid K\rangle \ (\text{Clash})_r \qquad \langle S, y \xrightarrow{r} x, x : \neg\exists r^-.\top \mid U \mid K\rangle \ (\text{Clash})_{r^-}$$

$$\langle S \mid U \mid \emptyset\rangle\ (\text{Clash})_\emptyset \qquad \begin{array}{r}\langle S, x : \neg\Box\neg C \mid U \mid K\rangle\ (\text{Clash})_{\Box^-}\\ \text{if } x : \neg\Box\neg C \notin K\end{array}$$

$$\dfrac{\langle S \mid U \mid K\rangle}{\langle S, x : \Box\neg C \mid U \mid K\rangle \quad \langle S, x : \neg\Box\neg C \mid U \mid K\rangle}\ (\mathit{cut}) \quad \begin{array}{l}\text{if } x : \neg\Box\neg C \notin S \text{ and } x : \Box\neg C \notin S\\ x \in \mathcal{D}(\mathbf{B}) \qquad C \in \mathcal{L}_{\mathbf{T}}\end{array}$$

$$\dfrac{\langle S, x : \mathbf{T}(C) \mid U \mid K\rangle}{\langle S, x : C, x : \Box\neg C \mid U \mid K\rangle}\ (\mathbf{T}^+) \qquad \dfrac{\langle S, x : \neg\mathbf{T}(C) \mid U \mid K\rangle}{\langle S, x : \neg C \mid U \mid K\rangle \quad \langle S, x : \neg\Box\neg C \mid U \mid K\rangle}\ (\mathbf{T}^-)$$

$$\dfrac{\langle S, x : \neg\Box\neg C_1, \ldots, x : \neg\Box\neg C_n \mid U \mid K, x : \neg\Box\neg C_1, \ldots, x : \neg\Box\neg C_n\rangle}{\langle S, y_1 : C_k, y_1 : \Box\neg C_k, S^M_{x\rightarrow y_1}, \overline{S}^{\Box^{-k}}_{x\rightarrow y_1} \mid U \mid K\rangle \ \cdots\ \langle S, y_m : C_k, y_m : \Box\neg C_k, S^M_{x\rightarrow y_m}, \overline{S}^{\Box^{-k}}_{x\rightarrow y_m} \mid U \mid K\rangle}\ (\Box^-)$$
$$\begin{array}{r}\text{if } \mathcal{D}(\mathbf{B}) = \{y_1, \ldots, y_m\} \text{ and } y_1 \neq x, \ldots, y_m \neq x\\ \forall\, k = 1, 2, \ldots, n\end{array}$$

$$\dfrac{\langle S \mid U, C \sqsubseteq D^L \mid K\rangle}{\langle S, x : \neg C \sqcup D \mid U, C \sqsubseteq D^{L,x} \mid K\rangle}\ (\text{Unfold}) \quad x \in \mathcal{D}(\mathbf{B}) \text{ and } x \notin L \qquad \dfrac{\langle S, x : C \sqcup D \mid U \mid K\rangle}{\langle S, x : C \mid U \mid K\rangle \ \langle S, x : D \mid U \mid K\rangle}(\sqcup^+)$$

$$\dfrac{\langle S, x : \exists r.\top \mid U \mid K\rangle}{\langle S, x \xrightarrow{r} y_1 \mid U \mid K\rangle \ldots \langle S, x \xrightarrow{r} y_m \mid U \mid K\rangle}\ (\exists^+)^r \quad \text{if } \mathcal{D}(\mathbf{B}) = \{y_1, \ldots, y_m\}$$

$$\dfrac{\langle S, x : \exists r^-.\top \mid U \mid K\rangle}{\langle S, y_1 \xrightarrow{r} x \mid U \mid K\rangle \ldots \langle S, y_m \xrightarrow{r} x \mid U \mid K\rangle}\ (\exists^+)^{r^-} \quad \text{if } \mathcal{D}(\mathbf{B}) = \{y_1, \ldots, y_m\}$$

**Fig. 2.** The calculus $\mathcal{TAB}_{PH2}^{\mathit{Lite}_c\mathbf{T}}$

$\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ detects a clash, called (Clash)$_{\Box^-}$: this corresponds to the situation in which $x : \neg\Box\neg C$ does not belong to $\mathbf{B}$, while the model corresponding to the branch being built contains $x : \neg\Box\neg C$, and hence is *not* preferred to the model represented by $\mathbf{B}$.

The calculus $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ contains also the clash condition (Clash)$_\emptyset$. Since each application of $(\Box^-)$ removes the negated box formulas $x : \neg\Box\neg C_i$ from the set $K$, when $K$ is empty all the negated boxed formulas occurring in $\mathbf{B}$ also belong to the current branch. In this case, the model built by $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ satisfies the same set of $x : \neg\Box\neg C_i$ (for all individuals) as $\mathbf{B}$ and, thus, it is not preferred to the one represented by $\mathbf{B}$. $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ is sound and complete:

**Theorem 6 (Soundness and completeness of $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$).** *Given a* KB *and a query $F$, let $\langle S' \mid U\rangle$ be the corresponding constraint system of* KB*, and $\langle S \mid U\rangle$ the corresponding constraint system of* KB $\cup$ $\{\neg F\}$*. An open saturated branch* $\mathbf{B}$ *built by $\mathcal{TAB}_{PH1}^{\text{Lite}_c\mathbf{T}}$ for $\langle S \mid U\rangle$ is satisfiable by an injective mapping in a minimal model of* KB *iff the tableau in $\mathcal{TAB}_{PH2}^{\text{Lite}_c\mathbf{T}}$ for $\langle S' \mid U \mid \mathbf{B}^{\Box^-}\rangle$ is closed.*

$\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ always terminates. Termination is ensured by the fact that dynamic rules make use of labels belonging to $\mathcal{D}(\mathbf{B})$, which is finite, rather than introducing "new" labels in the tableau.

**Theorem 7 (Termination of $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$).** *Let $\langle S' \mid U \mid \mathbf{B}^{\Box^-}\rangle$ be a constraint system starting from an open saturated branch* $\mathbf{B}$ *built by $\mathcal{TAB}_{PH1}^{\text{Lite}_c\mathbf{T}}$, then any tableau generated by $\mathcal{TAB}_{PH2}^{\text{Lite}_c\mathbf{T}}$ is finite.*

It is possible to show that the problem of verifying that a branch $\mathbf{B}$ represents a minimal model for KB in $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ is in NP in the size of $\mathbf{B}$.

The overall procedure $\mathcal{TAB}_{min}^{\textit{Lite}_c\mathbf{T}}$ is defined as follows:

**Definition 8.** *Let KB be a knowledge base whose corresponding constraint system is $\langle S \mid U\rangle$. Let $F$ be a query and let $S'$ be the set of constraints obtained by adding to $S$ the constraint corresponding to $\neg F$. The calculus $\mathcal{TAB}_{min}^{\text{Lite}_c\mathbf{T}}$ says that* KB $\models_{\text{DL-Lite}_c\mathbf{T}_{min}}$ *$F$ iff for each branch* $\mathbf{B}$ *built by $\mathcal{TAB}_{PH1}^{\text{Lite}_c\mathbf{T}}$, either (i)* $\mathbf{B}$ *is closed or (ii) the tableau built in* (phase 2) *by the calculus $\mathcal{TAB}_{PH2}^{\text{Lite}_c\mathbf{T}}$ for $\langle S \mid U \mid \mathbf{B}^{\Box^-}\rangle$ is open.*

**Theorem 8 (Soundness and completeness of $\mathcal{TAB}_{min}^{\textit{Lite}_c\mathbf{T}}$).** *$\mathcal{TAB}_{min}^{\text{Lite}_c\mathbf{T}}$ is a sound and complete decision procedure for verifying whether KB $\models_{\text{DL-Lite}_c\mathbf{T}_{min}}$ $F$.*

*Proof.* (Soundness) If for all branches $\mathbf{B}$ (i) holds, then by Theorem 2, KB $\models_{\textit{DL-Lite}_c\mathbf{T}_{min}}$ $F$. Consider an open saturated branch $\mathbf{B}$ for which (ii) holds, by Theorem 6, $\mathbf{B}$ is not satisfiable via an injective mapping in a minimal model of KB, hence also in this case by Theorem 2, KB $\models_{\textit{DL-Lite}_c\mathbf{T}_{min}}$ $F$.

(Completeness) Let KB $\models_{\textit{DL-Lite}_c\mathbf{T}_{min}}$ $F$. For contraposition, let $\mathbf{B}$ be an open saturated branch (if any) generated by $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$. If this branch was satisfiable by an injective mapping in a minimal model of KB, then by Proposition 1, also KB $\cup \{\neg F\}$ would be, against the hypothesis that KB $\models_{\textit{DL-Lite}_c\mathbf{T}_{min}} F$. Hence, **B** is not satisfiable by an injective mapping in a minimal model of KB, and by Theorem 6 the tableau in $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ for $\langle S \mid U \mid \mathbf{B}^{\Box^-} \rangle$ is open. Hence (i) or (ii) hold. ■

We can also prove that the complexity of $\mathcal{TAB}_{min}^{\textit{Lite}_c\mathbf{T}}$ matches the known results for minimal entailment in *DL-Lite*$_c\mathbf{T}_{min}$ of Theorem 1:

**Theorem 9 (Complexity of $\mathcal{TAB}_{min}^{\textit{Lite}_c\mathbf{T}}$).** *The problem of deciding whether KB $\models_{\text{DL-Lite}_c\mathbf{T}_{min}} F$ by means of $\mathcal{TAB}_{min}^{\text{Lite}_c\mathbf{T}}$ is in $\Pi_2^p$.*

*Proof.* We first consider the complementary problem: KB $\not\models_{min}^{\mathcal{L}_\mathbf{T}} F$. This problem can be solved according to the procedure in Definition 8: by nondeterministically generating an open saturated branch of polynomial length in the size of KB in $\mathcal{TAB}_{PH1}^{\textit{Lite}_c\mathbf{T}}$ (a model $\mathcal{M}^\mathbf{B}$ of KB $\cup \{\neg F\}$), and then by calling an NP oracle which verifies that $\mathcal{M}^\mathbf{B}$ is a minimal model of KB. In fact, the verification that $\mathcal{M}^\mathbf{B}$ is not a minimal model of the KB can be done by an NP algorithm which nondeterministically generates a branch in $\mathcal{TAB}_{PH2}^{\textit{Lite}_c\mathbf{T}}$ of polynomial size in the size of $\mathcal{M}^\mathbf{B}$ (and of KB), representing a model $\mathcal{M}^{B'}$ of KB preferred to $\mathcal{M}^\mathbf{B}$. Hence, the problem of verifying that KB $\not\models_{min}^{\mathcal{L}_\mathbf{T}} F$ is in NP$^{\text{NP}}$, that is to say in $\Sigma_2^p$, and the problem of deciding whether KB $\models_{\textit{DL-Lite}_c\mathbf{T}_{min}}$ $F$ is in CO-NP$^{\text{NP}}$, that is to say in $\Pi_2^p$. ■

## 4 Related Works and Conclusions

Several nonmonotonic extensions of DLs have been proposed in the literature [15, 5, 2, 3, 9, 13, 10, 8]. Recently, much attention has been devoted to nonmonotonic extensions of low complexity DLs. The complexity of *circumscribed* fragments of the $\mathcal{EL}^\perp$ and DL-lite families have been studied in [3]. A fragment of $\mathcal{EL}^\perp$ for which the complexity of circumscribed KBs is polynomial has been identified in [4].

In this work we have provided a two-phase tableau calculus $\mathcal{TAB}_{min}^{\textit{Lite}_c\mathbf{T}}$ for checking minimal entailment in a nonmonotonic extension of the Description Logic *DL-Lite*$_{core}$, described in [12]. This fills the gap due to the lack of a calculus for this logic. The proposed calculus matches the known complexity results for *DL-Lite*$_c\mathbf{T}_{min}$, namely that entailment is in $\Pi_2^p$ [12].

**Acknowledgements.** The work has been partially supported by the project “MIUR PRIN08 LoDeN: Logiche Descrittive Nonmonotone: Complessitá e implementazioni”.

## References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: IJCAI, pp. 364–369 (2005)
2. Baader, F., Hollunder, B.: Priorities on defaults with prerequisites, and their application in treating specificity in terminological default logic. J. of Automated Reasoning (JAR) 15(1), 41–68 (1995)

3. Bonatti, P., Faella, M., Sauro, L.: Defeasible inclusions in low-complexity dls: Preliminary notes. In: IJCAI, pp. 696–701 (2009)
4. Bonatti, P., Faella, M., Sauro, L.: $\mathcal{EL}$ with default attributes and overriding. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 64–79. Springer, Heidelberg (2010)
5. Bonatti, P., Lutz, C., Wolter, F.: Description logics with circumscription. In: KR, pp. 400–410 (2006)
6. Buchheit, M., Donini, F.M., Schaerf, A.: Decidable reasoning in terminological knowledge representation systems. J. Artif. Int. Research (JAIR) 1, 109–138 (1993)
7. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family. J. Autom. Reasoning (JAR) 39(3), 385–429 (2007)
8. Casini, G., Straccia, U.: Rational closure for defeasible description logics. In: Janhunen, T., Niemelä, I. (eds.) JELIA 2010. LNCS, vol. 6341, pp. 77–90. Springer, Heidelberg (2010)
9. Donini, F.M., Nardi, D., Rosati, R.: Description logics of minimal knowledge and negation as failure. ACM Trans. Comput. Log. 3(2), 177–225 (2002)
10. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: Reasoning About Typicality in Preferential Description Logics. In: Hölldobler, S., Lutz, C., Wansing, H. (eds.) JELIA 2008. LNCS (LNAI), vol. 5293, pp. 192–205. Springer, Heidelberg (2008)
11. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: A tableau calculus for a nonmonotonic extension of $\mathcal{EL}^{\perp}$. In: Brünnler, K., Metcalfe, G. (eds.) TABLEAUX 2011. LNCS (LNAI), vol. 6793, pp. 180–195. Springer, Heidelberg (2011)
12. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: Reasoning about typicality in low complexity DLs: the logics $\mathcal{EL}^{\perp}\mathbf{T}_{min}$ and *DL-Lite*$_c\mathbf{T}_{min}$. In: IJCAI, pp. 894–899 (2011)
13. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: $\mathcal{ALC} + \mathbf{T}_{min}$: a preferential extension of description logics. Fundamenta Informaticae 96, 1–32 (2009)
14. Kraus, S., Lehmann, D., Magidor, M.: Nonmonotonic reasoning, preferential models and cumulative logics. Artificial Intelligence 44(1-2), 167–207 (1990)
15. Straccia, U.: Default inheritance reasoning in hybrid kl-one-style logics. In: IJCAI, pp. 676–681 (1993)