

On Extending Description Logics for Reasoning About Typicality: a First Step

Laura Giordano^{1,2,3}

*Dipartimento di Informatica
Università del Piemonte Orientale “A. Avogadro”
Alessandria, Italy*

Valentina Gliozzi^{1,2,4}

*Dipartimento di Informatica
Università degli Studi di Torino
Torino, Italy*

Nicola Olivetti^{1,2,5}

*LSIS-UMR CNRS 6168
Université “Paul Cézanne”
Aix en Provence, France*

Gian Luca Pozzato^{1,2,6}

*Dipartimento di Informatica
Università degli Studi di Torino
Torino, Italy*

Abstract

We extend the Description Logic \mathcal{ALC} with a “typicality” operator \mathbf{T} that allows us to reason about the prototypical properties and inheritance with exceptions. The resulting logic is called $\mathcal{ALC} + \mathbf{T}$. The typicality operator is intended to select the “most normal” or “most typical” instances of a concept. In our framework, knowledge bases may then contain, in addition to ordinary ABoxes and TBoxes, subsumption relations of the form “ $\mathbf{T}(C)$ is subsumed by P ”, expressing that typical C -members have the property P . The semantics of a typicality operator is defined by a set of postulates that are strongly related to Kraus-Lehmann-Magidor axioms of preferential logic \mathbf{P} . We first show that \mathbf{T} enjoys a simple semantics provided by ordinary structures equipped by a preference relation. This allows us to obtain a modal interpretation of the typicality operator. Using such a modal interpretation, we present a tableau calculus for deciding satisfiability of $\mathcal{ALC} + \mathbf{T}$ knowledge bases. Our calculus gives a nondeterministic-exponential time decision procedure for satisfiability of $\mathcal{ALC} + \mathbf{T}$. We then extend $\mathcal{ALC} + \mathbf{T}$ knowledge bases by a nonmonotonic completion that allows inferring defeasible properties of specific concept instances.

Keywords: Description Logics, Nonmonotonic Reasoning, Preferential Logics, Tableaux Calculi.

1 Introduction

The family of description logics (DLs) is one of the most important formalisms of knowledge representation. DLs are reminiscent of the early semantic networks and of frame-based systems. They offer two key advantages: a well-defined semantics based on first-order logic and a good trade-off between expressivity and complexity. DLs have been successfully implemented by a range of systems and they are at the base of languages for the semantic web such as OWL. A DL knowledge base (KB) comprises two components: (i) the TBox, containing the definition of concepts (and possibly roles), and a specification of inclusions relations among them, and (ii) the ABox containing instances of concepts and roles, in other words, properties and relations of individuals. Since the very objective of the TBox is to build a taxonomy of concepts, the need of representing prototypical properties and of reasoning about defeasible inheritance of such properties easily arises. The traditional approach is to handle defeasible inheritance by integrating some kind of nonmonotonic reasoning mechanism. This has led to study nonmonotonic extensions of DLs [1,2,3,5,6,7,13]. However, finding a suitable nonmonotonic extension for inheritance reasoning with exceptions is far from obvious. Let us put forward some desiderata for such an extension:

- (i) The (nonmonotonic) extension must have a clear semantics and should be based on the same semantics as the underlying monotonic DL.
- (ii) The extension should allow to specify prototypical properties in a natural and direct way.
- (iii) The extension must be decidable, if so is the underlying monotonic DL and, possibly, computationally effective.

The nonmonotonic extensions proposed in the literature do not seem to fully satisfy all the above desiderata.

[1] proposes the extension of DL with Reiter's default logic. However, the same authors have pointed out that this integration may lead to both semantical and computational difficulties. Indeed, the unsatisfactory treatment of open defaults via Skolemization may lead to an undecidable default consequence relation. For this reason, [1] proposes a restricted semantics for open default theories, in which default rules are only applied to individuals explicitly mentioned in the ABox. Furthermore, Reiter's default logic does not provide a direct way of modeling inheritance with exceptions. This has motivated the study of extensions of DLs with prioritized defaults [13,2].

A more general approach is undertaken in [6], where it is proposed an extension of DL with two epistemic operators. This extension allows one to encode Reiter's default logic as well as to express epistemic concepts and procedural rules. However,

¹ This paper has been also presented at the 13th Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2007) [10].

² This research has been partially supported by "MIUR PRIN05: Specification and verification of agent interaction protocols".

³ Email: laura@mfn.unipmn.it

⁴ Email: gliozzi@di.unito.it

⁵ Email: nicola.olivetti@univ-cezanne.fr

⁶ Email: pozzato@di.unito.it

this extension has a rather complicated modal semantics, so that the integration with the existing systems requires significant changes to the standard semantics of DLs.

In [3] the authors propose an extension of DL with circumscription. One of motivating applications of circumscription is indeed to express prototypical properties with exceptions, and this is done by introducing “abnormality” predicates, whose extension is minimized. The authors provide decidability and complexity results based on theoretical analysis. However, they do not provide a calculus for their logic. Moreover, the use of circumscription to model inheritance with exceptions is not that straightforward, as we remark below.

In this work, we propose a novel approach to defeasible inheritance reasoning based on the typicality operator \mathbf{T} . The intended meaning is that, for any concept C , $\mathbf{T}(C)$ singles out the instances of C that are considered as “typical” or “normal”. Thus assertions as “normally students do not pay taxes”, or “typically users do not have access to confidential files” [3] are represented by $\mathbf{T}(\textit{Student}) \sqsubseteq \neg \textit{TaxPayer}$ and $\mathbf{T}(\textit{User}) \sqsubseteq \neg \exists \textit{hasAccess}.\textit{ConfidentialFile}$.

Before entering in the technical details, let us sketch how we intend to use the typicality operator and what kind of inferential services we expect to profit. We assume that a KB comprises, in addition to the standard TBox and ABox, a set of assertions of the type $\mathbf{T}(C) \sqsubseteq D$ where D is a concept not mentioning \mathbf{T} . The reasoning system should be able to infer or propagate prototypical properties of the concepts specified in the TBox, then to ascribe defeasible properties to individuals. For instance, let the KB contain:

$$\begin{aligned} \mathbf{T}(\textit{Student}) &\sqsubseteq \neg \textit{TaxPayer} \\ \mathbf{T}(\textit{Student} \sqcap \textit{Worker}) &\sqsubseteq \textit{TaxPayer} \\ \mathbf{T}(\textit{Student} \sqcap \textit{Worker} \sqcap \exists \textit{HasChild}.\top) &\sqsubseteq \neg \textit{TaxPayer} \\ \mathbf{T}(\textit{Unemployed}) &\sqsubseteq \neg \textit{TaxPayer} \end{aligned}$$

corresponding to the assertions: normally a student does not pay taxes, normally a working student pays taxes, but normally a working student having children does not pay taxes (because he is discharged by the government) etc...Observe that, if the same properties were expressed by ordinary inclusions, such as $\textit{Student} \sqsubseteq \neg \textit{TaxPayer}$ etc...we would simply get that there are not working students and so on, thus the KB would collapse. This collapse is avoided as we do not assume that \mathbf{T} is monotonic, that is to say $C \sqsubseteq D$ does not imply $\mathbf{T}(C) \sqsubseteq \mathbf{T}(D)$. To continue with the example, if the TBox contains $\textit{PersonWithNoIncome} \equiv \textit{Student} \sqcup \textit{Unemployed}$, then the system should be able to infer $\mathbf{T}(\textit{PersonWithNoIncome}) \sqsubseteq \neg \textit{TaxPayer}$. Suppose next that the ABox contains alternatively the following facts about *john*:

1. $\textit{student}(\textit{john})$
2. $\textit{student}(\textit{john}), \textit{worker}(\textit{john})$
3. $\textit{student}(\textit{john}), \textit{worker}(\textit{john}), \exists \textit{HasChild}.\top(\textit{john})$

Then the reasoning system should be able to infer the expected (defeasible) conclusion about *john* in each case:

1. $\neg \textit{TaxPayer}(\textit{john})$
2. $\textit{TaxPayer}(\textit{john})$

3. $\neg \text{TaxPayer}(\text{john})$

Observe that setting up a similar specification of the KB by using default logic or circumscription is not that simple: with default logic [1,2,5,6,7,13], one has to specify a priority on default application (or one has to find a smart encoding of defaults giving priority to more specific information); with circumscription [3], one has to introduce abnormality predicates, and then establish which predicates are minimized, fixed, or variable, and finally for the minimized ones what is their priority with respect to minimization (a total or a partial order). As a further step, the system should be able to infer (defeasible) properties also of individuals implicitly introduced by existential restrictions, for instance, if the ABox further contains $\exists \text{HasChild}.\text{Student}(\text{jack})$, it should conclude (defeasibly) $\exists \text{HasChild}.\neg \text{TaxPayer}(\text{jack})$.

Given the nonmonotonic character of the \mathbf{T} operator, there is a difficulty with handling irrelevant information, for instance, given the KB as above, one should be able to infer as well:

$$\begin{aligned} \mathbf{T}(\text{Student} \sqcap \text{SportLover}) &\sqsubseteq \neg \text{TaxPayer} \\ \mathbf{T}(\text{Student} \sqcap \text{Worker} \sqcap \text{SportLover}) &\sqsubseteq \text{TaxPayer} \end{aligned}$$

as *SportLover* is irrelevant with respect to being a TaxPayer or not. For the same reason, the conclusion about *john* being a TaxPayer or not should not be influenced by the addition of *SportLover(john)* to the ABox. We refer to this problem as the problem of Irrelevance.

In this paper we lay down the base of an extension of DL with a typicality operator. Our starting point is a monotonic extension of the basic \mathcal{ALC} with the \mathbf{T} operator. The operator is supposed to satisfy a set of postulates that are essentially a reformulation of Kraus, Lehmann, and Magidor (KLM) axioms of preferential logic, namely the assertion $\mathbf{T}(C) \sqsubseteq P$ is equivalent to the conditional assertion $C \rightsquigarrow P$ of KLM preferential logic \mathbf{P} . It turns out that the semantics of the typicality operator can be equivalently specified by considering a preference relation (a strict partial order) on individuals: the typical members of a concept C are just the most preferred individuals, or “most normal”, of C according to the preference relation. The preference relation is the only additional ingredient that we need in our semantics. We assume that “most normal” members of a concept C always exist, whenever the concept C is non-empty. This assumption corresponds to the *Smoothness Condition* of KLM logics, or the well-known *Limit Assumption* in conditional logics. Taking advantage of this semantic setting, we can give a modal interpretation to the typicality operator: the modal operator \square has intuitively the same properties as in Gödel-Löb modal logic \mathbf{G} of arithmetic provability. We then define a tableau system for this extension based on this modal interpretation, thereby obtaining a decision procedure and an upper complexity bound (NEXPTIME).

These are the main results of the paper, however the monotonic extension is not enough to perform inheritance reasoning of the kind described above: (i) we need a way of inferring defeasible properties of individuals, (ii) we need a way of handling Irrelevance.

In the paper we deal with (i) by defining a completion of an ABox: the idea is that each individual is assumed to be a typical member of the most specific concept

to which it belongs. Such a completion allows to perform inferences as the ones 1.,2.,3. above. The paper outlines how we intend to cope with typicality of all instances and with Irrelevance. In particular, dealing with Irrelevance (ii) requires a nonmonotonic mechanism. The idea is to complete a KB with a set of default rules. The default rules are not used to express defeasible properties of concepts (as in default extension of DLs), but to propagate defeasible properties of a concept to its subsumed concepts, e.g. to infer $\mathbf{T}(Student \sqcap SportLover) \sqsubseteq \neg TaxPayer$ from $\mathbf{T}(Student) \sqsubseteq \neg TaxPayer$. Thus in our approach the nonmonotonic mechanism is only needed to handle Irrelevance, whose treatment by means of default rules will be the object of future work.

2 The logic $\mathcal{ALC} + \mathbf{T}$: the typicality operator \mathbf{T}

We consider an alphabet of concept names \mathcal{C} , of role names \mathcal{R} , and of individuals \mathcal{O} . The language \mathcal{L} of the logic $\mathcal{ALC} + \mathbf{T}$ is defined by distinguishing *concepts* and *extended concepts* as follows: (Concepts) $A \in \mathcal{C}$ and \top are *concepts* of \mathcal{L} ; if $C, D \in \mathcal{L}$ and $R \in \mathcal{R}$, then $C \sqcap D, C \sqcup D, \neg C, \forall R.C, \exists R.C$ are *concepts* of \mathcal{L} . (Extended concepts) if C is a concept, then C and $\mathbf{T}(C)$ are *extended concepts*, and all the boolean combinations of extended concepts are extended concepts of \mathcal{L} . A knowledge base is a pair (TBox, ABox). TBox contains subsumptions $C \sqsubseteq D$, where $C \in \mathcal{L}$ is either a concept or an extended concept $\mathbf{T}(C')$, and $D \in \mathcal{L}$ is a concept. ABox contains expressions of the form $C(a)$ and aRb where $C \in \mathcal{L}$ is an extended concept, $R \in \mathcal{R}$, and $a, b \in \mathcal{O}$.

In order to provide a semantics to the operator \mathbf{T} , we extend the definition of a model used in “standard” terminological logic \mathcal{ALC} :

Definition 2.1 [Semantics of \mathbf{T} with selection function] A model is any structure $\langle \Delta, I, f_{\mathbf{T}} \rangle$, where: Δ is the domain; I is the extension function that maps each extended concept C to $C^I \subseteq \Delta$, and each role R to a $R^I \subseteq \Delta^I \times \Delta^I$. I is defined in the usual way (as for \mathcal{ALC}) and, in addition, $(\mathbf{T}(C))^I = f_{\mathbf{T}}(C^I)$. $f_{\mathbf{T}} : Pow(\Delta) \rightarrow Pow(\Delta)$ is a function satisfying the following properties:

$$\begin{array}{ll}
 (f_{\mathbf{T}} - 1) \ f_{\mathbf{T}}(S) \subseteq S & (f_{\mathbf{T}} - 2) \ \text{if } S \neq \emptyset, \text{ then also } f_{\mathbf{T}}(S) \neq \emptyset \\
 (f_{\mathbf{T}} - 3) \ \text{if } f_{\mathbf{T}}(S) \subseteq R, \text{ then } f_{\mathbf{T}}(S) = f_{\mathbf{T}}(S \cap R) & (f_{\mathbf{T}} - 4) \ f_{\mathbf{T}}(\bigcup S_i) \subseteq \bigcup f_{\mathbf{T}}(S_i) \\
 (f_{\mathbf{T}} - 5) \ \bigcap f_{\mathbf{T}}(S_i) \subseteq f_{\mathbf{T}}(\bigcup S_i) &
 \end{array}$$

Intuitively, given the extension of some concept C , $f_{\mathbf{T}}$ selects the *typical* instances of C . $(f_{\mathbf{T}} - 1)$ requests that typical elements of S belong to S . $(f_{\mathbf{T}} - 2)$ requests that if there are elements in S , then there are also *typical* such elements. The next properties constraint the behavior of $f_{\mathbf{T}}$ with respect to \cap and \cup in such a way that they do not entail monotonicity. According to $(f_{\mathbf{T}} - 3)$, if the typical elements of S are in R , then they coincide with the typical elements of $S \cap R$, thus expressing a weak form of monotonicity (namely *cautious monotonicity*). $(f_{\mathbf{T}} - 4)$ corresponds to one direction of the equivalence $f_{\mathbf{T}}(\bigcup S_i) = \bigcup f_{\mathbf{T}}(S_i)$, so that it does not entail monotonicity. Similar considerations apply to the equation $f_{\mathbf{T}}(\bigcap S_i) = \bigcap f_{\mathbf{T}}(S_i)$, of which only the inclusion $\bigcap f_{\mathbf{T}}(S_i) \subseteq f_{\mathbf{T}}(\bigcap S_i)$ is derivable. $(f_{\mathbf{T}} - 5)$ is a further constraint on the behavior of $f_{\mathbf{T}}$ with respect to arbitrary unions and intersections;

it would be derivable if $f_{\mathbf{T}}$ were monotonic. We can prove the following proposition:

Proposition 2.2 $f_{\mathbf{T}}(S \cup R) \cap S \subseteq f_{\mathbf{T}}(S)$

We can give an alternative semantics for \mathbf{T} based on a preference relation. The idea is that there is a global preference relation among individuals and that the typical members of a concept C , i.e. selected by $f_{\mathbf{T}}(C^I)$, are the minimal elements of C with respect to this preference relation. Observe that this notion is global, that is to say, it does not compare individuals with respect to a specific concept (something like y is more typical than x with respect to concept C). In this framework, an object $x \in \Delta$ is a *typical instance* of some concept C , if $x \in C^I$ and there is no C -element in Δ more typical than x . The typicality preference relation is partial since it is not always possible to establish which object is more typical than which other. The following definition is needed before we provide the Representation Theorem.

Definition 2.3 Given a relation $<$, which is a strict partial order (i.e. an irreflexive and transitive relation) over a domain Δ , for all $S \subseteq \Delta$, we define $Min_{<}(S) = \{x : x \in S \text{ and } \nexists y \in S \text{ s.t. } y < x\}$. We say that $<$ satisfies the *Smoothness Condition* iff for all $S \subseteq \Delta$, for all $x \in S$, either $x \in Min_{<}(S)$ or $\exists y \in Min_{<}(S)$ such that $y < x$.

We are now ready to prove the Representation Theorem below, showing that given a model with a selection function, we can define *on the same domain* a preference relation $<$ such that, for all $S \subseteq \Delta$, $f_{\mathbf{T}}(S) = Min_{<}(S)$. Notice that, as a difference with respect to related results (Theorem 3 in [12]), the relation is defined on the same domain Δ of $f_{\mathbf{T}}$. On the other hand, if $<$ is a preference relation satisfying the Smoothness Condition, then the operator defined as $f_{\mathbf{T}}(S) = Min_{<}(S)$ satisfies the postulates of Definition 2.1.

Theorem 2.4 (Representation Theorem) *Given any model $\langle \Delta, I, f_{\mathbf{T}} \rangle$, $f_{\mathbf{T}}$ satisfies postulates $(f_{\mathbf{T}} - 1)$ to $(f_{\mathbf{T}} - 5)$ above iff it is possible to define on Δ a strict partial order $<$, satisfying the Smoothness Condition, such that for all $S \subseteq \Delta$, $f_{\mathbf{T}}(S) = Min_{<}(S)$.*

Proof. (“Only if” direction) Given $f_{\mathbf{T}}$ satisfying postulates $(f_{\mathbf{T}} - 1)$ to $(f_{\mathbf{T}} - 5)$, we define $<$ as follows: for all $x, y \in \Delta$, we let $x < y$ if $\forall S \subseteq \Delta$, if $y \in f_{\mathbf{T}}(S)$ then $x \notin S$, and $\exists R \subseteq \Delta$ such that $S \subset R$ and $x \in f_{\mathbf{T}}(R)$. We prove that:

- (i) $<$ is irreflexive. Easily follows by the definition of $<$.
- (ii) $<$ is transitive. Let (a) $x < y$ and (b) $y < z$. Let $z \in f_{\mathbf{T}}(S)$ for some S , then by definition of $<$, $y \notin S$, and $\exists R$ s.t. $S \subset R$ and $y \in f_{\mathbf{T}}(R)$. Furthermore, $x \notin R$ and $\exists Q : R \subset Q$ and $x \in f_{\mathbf{T}}(Q)$. From this we can conclude that $x \notin S$ (otherwise $x \in R$), and $S \subset Q$, hence $x < z$.
- (iii) $f_{\mathbf{T}}(S) \subseteq Min_{<}(S)$. Let $x \in f_{\mathbf{T}}(S)$. Suppose $x \notin Min_{<}(S)$, i.e. for some $y \in S$, $y < x$. By definition of $<$, $y \notin S$, contradiction, hence $x \in Min_{<}(S)$.
- (iv) $Min_{<}(S) \subseteq f_{\mathbf{T}}(S)$. Let $x \in Min_{<}(S)$. Then $x \in S$, i.e. $S \neq \emptyset$. By $(f_{\mathbf{T}} - 2)$, $f_{\mathbf{T}}(S) \neq \emptyset$. Suppose $x \notin f_{\mathbf{T}}(S)$. Consider $\bigcup R_i$ for all $R_i \subseteq \Delta$ s.t. $x \in f_{\mathbf{T}}(R_i)$. By $(f_{\mathbf{T}} - 5)$, we have $x \in f_{\mathbf{T}}(\bigcup R_i)$. Consider now $f_{\mathbf{T}}(\bigcup R_i \cup S)$. We can easily show that $f_{\mathbf{T}}(\bigcup R_i \cup S) \not\subseteq \bigcup R_i$

(otherwise, by $(f_{\mathbf{T}} - 3)$ $f_{\mathbf{T}}(\bigcup R_i \cup S) = f_{\mathbf{T}}(\bigcup R_i)$, and by Proposition 2.2, $f_{\mathbf{T}}(\bigcup R_i) \cap S \subseteq f_{\mathbf{T}}(S)$, which contradicts the fact that $x \in f_{\mathbf{T}}(\bigcup R_i)$ but $x \notin f_{\mathbf{T}}(S)$). Consider hence $y \in f_{\mathbf{T}}(\bigcup R_i \cup S)$ s.t. $y \notin \bigcup R_i$. By definition of $<$, $y < x$. Furthermore, by $(f_{\mathbf{T}} - 1)$ $y \in S$ (since $y \in \bigcup R_i \cup S$ and $y \notin \bigcup R_i$). It follows that $x \notin \text{Min}_{<}(S)$, contradiction, hence $\text{Min}_{<}(S) \subseteq f_{\mathbf{T}}(S)$.

- (v) $<$ satisfies the Smoothness Condition. Let $S \neq \emptyset$ and $x \in S$. If $x \in f_{\mathbf{T}}(S)$ then by point 3 we have $x \in \text{Min}_{<}(S)$. If $x \notin f_{\mathbf{T}}(S)$ we can reason as for point 4 to conclude that there is $y \in f_{\mathbf{T}}(\bigcup R_i \cup S)$ s.t. $y \notin \bigcup R_i$ (hence $y \in S$), and $y < x$. By Proposition 2.2, we have $y \in f_{\mathbf{T}}(S)$, hence by point 3 we conclude $y \in \text{Min}_{<}(S)$.

The points above allow us to conclude.

(“If” direction) Given a strict partial order $<$ satisfying the Smoothness Condition, we can define $f_{\mathbf{T}} : \text{Pow}(\Delta) \rightarrow \text{Pow}(\Delta)$ by letting $f_{\mathbf{T}}(S) = \text{Min}_{<}(S)$. It can be easily shown that $f_{\mathbf{T}}$ satisfies postulates $(f_{\mathbf{T}} - 1)$ to $(f_{\mathbf{T}} - 5)$. The proof is left to the reader. \square

Having the above Representation System, from now on, we will refer to the following semantics for $\mathcal{ALC} + \mathbf{T}$:

Definition 2.5 [Semantics of $\mathcal{ALC} + \mathbf{T}$] A model \mathcal{M} is any structure $\langle \Delta, <, I \rangle$, where Δ and I are defined as in Definition 2.1, and $<$ is a strict partial order over Δ satisfying the Smoothness Condition (see Definition 2.3 above). As a difference with respect to Definition 2.1, the semantics of the \mathbf{T} operator is: $(\mathbf{T}(C))^I = \text{Min}_{<}(C^I)$. For concepts (built from operators of \mathcal{ALC}), C^I is defined in the usual way.

Definition 2.6 [Model satisfying a Knowledge Base] Consider a model \mathcal{M} , as defined in Definition 2.5. We extend I so that it assigns to each individual a of \mathcal{O} an element a^I of the domain Δ . Given a KB (TBox, ABox), we say that:

- \mathcal{M} satisfies TBox if for all inclusions $C \sqsubseteq D$ in TBox, and all elements $x \in \Delta$, if $x \in C^I$ then $x \in D^I$.
- \mathcal{M} satisfies ABox if: (i) for all $C(a)$ in ABox, we have that $a^I \in C^I$, (ii) for all aRb in ABox, we have that $(a^I, b^I) \in R^I$.

\mathcal{M} satisfies a knowledge base if it satisfies both its TBox and its ABox.

Notice that the meaning of \mathbf{T} can be split into two parts: for any object x of the domain Δ , $x \in (\mathbf{T}(C))^I$ just in case (i) $x \in C^I$, and (ii) there is no $y \in C^I$ such that $y < x$. In order to isolate the second part of the meaning of \mathbf{T} (for the purpose of the calculus that we will present in section 3) we introduce a new modality \square whose interpretation in \mathcal{M} is defined as follows.

Definition 2.7 $(\square C)^I = \{x \in \Delta \mid \text{for every } y \in \Delta, \text{ if } y < x \text{ then } y \in C^I\}$

The basic idea is simply to interpret the preference relation $<$ as an accessibility relation. By the Smoothness Condition, it turns out that the modality \square has the properties of Gödel-Löb modal logic of provability G. The Smoothness Condition ensures that typical elements of C^I exist whenever $C^I \neq \emptyset$, by preventing infinitely

descending chains of elements. This condition therefore corresponds to the finite-chain condition on the accessibility relation (as in G). A similar correspondence has been presented in [9,8] to interpret the preference relation in KLM logics. The following relation between \mathbf{T} and \Box holds:

Proposition 2.8 *For all $x \in \Delta$, we have $x \in (\mathbf{T}(C))^I$ iff $x \in C^I$ and $x \in (\Box\neg C)^I$*

Since we only use \Box to capture the meaning of \mathbf{T} , in the following we will always use \Box followed by a negated concept, as in $\Box\neg C$.

We can establish the following equation between our typicality operator and the nonmonotonic (conditional) inference operator \vdash (describing what can be *typically* derived from a given premise) by letting $C \vdash D$ iff $\mathbf{T}(C) \sqsubseteq D$. It can be easily shown that there is a correspondence between the properties of \mathbf{T} and the properties of \vdash in the system \mathbf{P} described in [12].

3 Reasoning

In this section we present a tableau calculus for deciding the satisfiability of a knowledge base. Given a KB (TBox, ABox), any concrete reasoning system should provide the usual reasoning services, namely *satisfiability of the KB*, *concept satisfiability*, *subsumption*, and *instance checking*. It is well known that the latter three services are reducible to the satisfiability of a KB.

We introduce a labelled tableau calculus for our logic $\mathcal{ALC} + \mathbf{T}$, which enriches the labelled tableau calculus for \mathcal{ALC} presented in [4]. The calculus is called $T^{\mathcal{ALC}+\mathbf{T}}$ and it is based on the notion of *constraint system*. We consider a set of *variables* drawn from a denumerable set \mathcal{V} . $T^{\mathcal{ALC}+\mathbf{T}}$ makes use of labels, which are denoted with x, y, z, \dots . Labels represent *objects*. An object is either a variable or an individual of the ABox, that is to say an element of $\mathcal{O} \cup \mathcal{V}$.

A *constraint* is a syntactic entity of the form $x \xrightarrow{R} y$ or $x : C$, where R is a role and C is either an extended concept or has the form $\Box\neg D$ or $\neg\Box\neg D$, where D is a concept. As we will define in Definition 3.1, the ABox of an $\mathcal{ALC} + \mathbf{T}$ -knowledge base can be translated into a set of constraints by replacing every membership assertion $C(a)$ with the constraint $a : C$ and every role aRb with the constraint $a \xrightarrow{R} b$. A tableau is a tree whose nodes are pairs $\langle S \mid U \rangle$, where:

- S contains constraints (or *labelled* formulas) of the form $x : C$ or $x \xrightarrow{R} y$;
- U contains formulas of the form $C \sqsubseteq D^L$, representing subsumption relations $C \sqsubseteq D$ of the TBox. L is a list of labels. As we will discuss later, this list is used in order to ensure the termination of the tableau calculus.

A node $\langle S \mid U \rangle$ is also called a *constraint system*. A branch is a sequence of nodes $\langle S_1 \mid U_1 \rangle, \langle S_2 \mid U_2 \rangle, \dots, \langle S_n \mid U_n \rangle, \dots$, where each node $\langle S_i \mid U_i \rangle$ is obtained by its immediate predecessor $\langle S_{i-1} \mid U_{i-1} \rangle$ by applying a rule of $T^{\mathcal{ALC}+\mathbf{T}}$, 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 (Clash), otherwise it is open. We say that a tableau is closed if all its branches are closed.

Given a KB, we define its *corresponding constraint system* as follows:

Definition 3.1 [Corresponding constraint system] Given an $\mathcal{ALC} + \mathbf{T}$ -knowledge base $(\text{TBox}, \text{ABox})$, we define its *corresponding constraint system* $\langle S \mid U \rangle$ as follows: $S = \{a : C \mid C(a) \in \text{ABox}\} \cup \{a \xrightarrow{R} b \mid aRb \in \text{ABox}\}$ and $U = \{C \sqsubseteq D^0 \mid C \sqsubseteq D \in \text{TBox}\}$.

Definition 3.2 [Model satisfying a constraint system] Let \mathcal{M} be a model as defined in Definition 2.6. We define a function α which assigns to each variable of \mathcal{V} an element of Δ , and assigns every individual $a \in \mathcal{O}$ to $a^I \in \Delta$. \mathcal{M} satisfies $x : C$ under α if $\alpha(x) \in C^I$ and $x \xrightarrow{R} y$ under α if $(\alpha(x), \alpha(y)) \in R^I$. A constraint system $\langle S \mid U \rangle$ is satisfiable if there is a model \mathcal{M} and a function α such that \mathcal{M} satisfies under α every constraint in S and that, for all $C \sqsubseteq D \in U$ and for all x occurring in S , we have that if $\alpha(x) \in C^I$ then $\alpha(x) \in D^I$.

It can be easily shown that:

Proposition 3.3 *Given an $\mathcal{ALC} + \mathbf{T}$ -knowledge base, it is satisfiable if and only if its corresponding constraint system is satisfiable.*

Therefore, in order to check the satisfiability of $(\text{TBox}, \text{ABox})$, we build its corresponding constraint system $\langle S \mid U \rangle$, and then we use $T^{\mathcal{ALC} + \mathbf{T}}$ to check the satisfiability of $\langle S \mid U \rangle$. In order to check a constraint system $\langle S \mid U \rangle$ for satisfiability, our calculus $T^{\mathcal{ALC} + \mathbf{T}}$ adopts the usual technique of applying the rules until either a contradiction is generated (Clash) or a model satisfying $\langle S \mid U \rangle$ can be obtained from the resulting constraint system.

In order to take into account the TBox, we use a technique of *unfolding*, similar to the one described in [4]. Given a node $\langle S \mid U \rangle$, for each subsumption $C \sqsubseteq D^L \in U$ and for each label x that appears in the tableau, we add to S the constraint $x : \neg C \sqcup D$. As mentioned above, each formula $C \sqsubseteq D$ is equipped by the list L of labels in which it has been unfolded in the current branch. This is needed in order to avoid multiple unfolding of the same subsumption by using the same label, generating non-termination in a proof search.

Before introducing the rules of $T^{\mathcal{ALC} + \mathbf{T}}$ we need some more definitions. First, as in [4], we assume that labels are introduced in a tableau according to an ordering \prec , that is to say if y is introduced in the tableau, then $x \prec y$ for all labels x that are already in the tableau.

Given a tableau node $\langle S \mid U \rangle$ and an object x , we define $\sigma(\langle S \mid U \rangle, x) = \{C \mid x : C \in S\}$. Furthermore, we say that two labels x and y are *S-equivalent*, written $x \equiv_S y$, if they label the same set of concepts, i.e. $\sigma(\langle S \mid U \rangle, x) = \sigma(\langle S \mid U \rangle, y)$. Intuitively, *S-equivalent* labels can represent the same element in the model built by the rules of $T^{\mathcal{ALC} + \mathbf{T}}$. Last, we define $S_{x \rightarrow y}^M = \{y : C, y : \Box \neg C \mid x : \Box \neg C \in S\}$.

The rules of $T^{\mathcal{ALC} + \mathbf{T}}$ are presented in Figure 1. Rules (\exists^+) , (\forall^-) , and (\Box^-) are called *dynamic* since they 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 $S_{x \rightarrow y}^M$. The side conditions on the rules (\exists^+) and (\forall^-) are introduced in order to ensure a terminating proof search, by implementing the standard *blocking* technique described below. The rules of $T^{\mathcal{ALC} + \mathbf{T}}$ are applied with the following *standard strategy*: 1. apply a rule to a variable $x \in \mathcal{V}$ only if no rule is applicable to a variable $y \in \mathcal{V}$ such

$\langle S, x : C, x : \neg C \mid U \rangle \quad (\text{Clash})$	$\frac{\langle S, x : \neg\neg C \mid U \rangle}{\langle S, x : C \mid U \rangle} (\neg)$
$\frac{\langle S, x : \mathbf{T}(C) \mid U \rangle}{\langle S, x : C, x : \Box\neg C \mid U \rangle} (\mathbf{T}^+)$	$\frac{\langle S, x : \neg\mathbf{T}(C) \mid U \rangle}{\langle S, x : \neg C \mid U \rangle \quad \langle S, x : \neg\Box\neg C \mid U \rangle} (\mathbf{T}^-)$
$\frac{\langle S, x : \forall R.C, x \xrightarrow{R} y \mid U \rangle}{\langle S, x : \forall R.C, x \xrightarrow{R} y, y : C \mid U \rangle} (\forall^+)$ <p style="text-align: center; margin: 0;"><small>if $y : C \notin S$</small></p>	$\frac{\langle S, x : \exists R.C \mid U \rangle}{\langle S, x : \exists R.C, x \xrightarrow{R} y, y : C \mid U \rangle} (\exists^+)$ <p style="text-align: center; margin: 0;"><small>y new if $\exists z \prec x$ s.t. $z \equiv_{S, x : \exists R.C} x$ and $\exists u$ s.t. $x \xrightarrow{R} u \in S$ and $u : C \in S$</small></p>
$\frac{\langle S, x : \neg\forall R.C \mid U \rangle}{\langle S, x : \neg\forall R.C, x \xrightarrow{R} y, y : \neg C \mid U \rangle} (\forall^-)$ <p style="text-align: center; margin: 0;"><small>y new if $\exists z \prec x$ s.t. $z \equiv_{S, x : \neg\forall R.C} x$ and $\exists u$ s.t. $x \xrightarrow{R} u \in S$ and $u : \neg C \in S$</small></p>	$\frac{\langle S, x : \neg\exists R.C, x \xrightarrow{R} y \mid U \rangle}{\langle S, x : \neg\exists R.C, x \xrightarrow{R} y, y : \neg C \mid U \rangle} (\exists^-)$ <p style="text-align: center; margin: 0;"><small>if $y : \neg C \notin S$</small></p>
$\frac{\langle S, x : \neg\Box\neg C \mid U \rangle}{\langle S, y : C, y : \Box\neg C, S_{x \rightarrow y}^M \mid U \rangle} (\Box^-)$ <p style="text-align: center; margin: 0;"><small>y new</small></p>	$\frac{\langle S \mid U, C \sqsubseteq D^L \rangle}{\langle S, x : \neg C \sqcup D \mid U, C \sqsubseteq D^{L, x} \rangle} (\text{Unfold})$ <p style="text-align: center; margin: 0;"><small>if x occurs in S and $x \notin L$</small></p>

Fig. 1. The calculus $T^{\text{ACC}+\mathbf{T}}$. To save space, we omit the standard rules for \sqcup and \sqcap .

that $y \prec x$; 2. apply dynamic rules ((\Box^-) first) only if no static rule is applicable. This strategy ensures that the variables are considered one at a time according to the ordering \prec . Consider an application of a dynamic rule to a variable x of a constraint system $\langle S \mid U \rangle$. For all $\langle S' \mid U' \rangle$ obtained from $\langle S \mid U \rangle$ by a sequence of rule applications, it can be easily shown that (i) no rule can be applied in $\langle S' \mid U' \rangle$ to a variable y s.t. $y \prec x$ and (ii) $\sigma(\langle S \mid U \rangle, x) = \sigma(\langle S' \mid U' \rangle, x)$. The calculus so obtained is sound and complete with respect to the semantics described in Definition 3.2. In order to prove this, we first introduce the notion of *witness*:

Definition 3.4 [Witness] Given a constraint system $\langle S \mid U \rangle$ and two variables x and y occurring in S , we say that x is a witness of y if the following conditions hold: 1. $x \equiv_S y$; 2. $x \prec y$; 3. there is no variable z s.t. $z \prec x$ and z satisfies conditions 1. and 2., i.e. x is the least variable satisfying conditions 1. and 2. w.r.t. \prec . We say that y is *blocked* by x in $\langle S \mid U \rangle$ if y has witness x .

By the strategy on the application of the rules described in section 3 and by Definition 3.4, we can prove the following Lemma:

Lemma 3.5 *In any constraint system $\langle S \mid U \rangle$, if x is blocked then it has exactly one witness.*

Since all the rules are invertible, we can assume that there is only one single tableau (there can be several that only differ as far as the labels' names are concerned). In order to prove the completeness of the calculus, we need to introduce the notion of saturated branch. Informally speaking, this is a branch in which all the rules of the calculus have been applied as much as possible. More formally:

Definition 3.6 [Saturated Branch] A branch $\mathbf{B} = \langle S_0 \mid U_0 \rangle, \langle S_1 \mid U_1 \rangle, \dots, \langle S_i \mid U_i \rangle, \dots$ is *saturated* if the following conditions hold:

- (i) for all $C \sqsubseteq D^L$ and for all labels x occurring in \mathbf{B} , either $x : \neg C$ or $x : D$ belong to \mathbf{B} ;
- (ii) if $x : \mathbf{T}(C)$ occurs in \mathbf{B} , then $x : C$ and $x : \Box\neg C$ occur in \mathbf{B} ;

- (iii) if $x : \neg\mathbf{T}(C)$ occurs in \mathbf{B} , then either $x : \neg C$ or $x : \neg\Box\neg C$ occurs in \mathbf{B} ;
- (iv) if $x : \neg\Box\neg C$ occurs in \mathbf{B} , then for some y , $y : C$, $y : \Box\neg C$, and $S_{x \rightarrow y}^M$ occur in \mathbf{B} ;
- (v) if $x : \exists R.C$ occurs in \mathbf{B} , then either there is y such that $x \xrightarrow{R} y$ and $y : C$ occur in \mathbf{B} or x is blocked by a witness w , and $w \xrightarrow{R} y$ and $y : C$ occur in \mathbf{B} ;
- (vi) if $x : \forall R.C$ and $x \xrightarrow{R} y$ occur in \mathbf{B} , also $y : C$ occurs in \mathbf{B} ;
- (vii) for $x : \neg\forall R.C$ and for $x : \neg\exists R.C$ the condition of saturation is defined symmetrically;
- (viii) for the boolean rules the condition of saturation is defined in the usual way. For instance, if $x : C \Box D$ occurs in \mathbf{B} , also $x : C$ and $x : D$.

By following the strategy on the order of application of the rules outlined in section 3 and by Lemma 3.5, the following Proposition holds:

Proposition 3.7 *Any open branch can be expanded into an open saturated branch.*

In order to show the completeness of $T^{\mathcal{ALC}+\mathbf{T}}$, given an open, saturated branch \mathbf{B} , we explicitly add the relation $y < x$ among all the objects y and x such that y has been obtained by applying (\Box^-) to a $x : \neg\Box\neg C$ occurring in \mathbf{B} . Before proving the completeness, we prove the following Lemma:

Lemma 3.8 *In the tableau, there is no open branch \mathbf{B} containing an infinite descending chain $\dots < x_0$.*

Proof. The only way to obtain an infinite descending chain $\dots x_i < \dots x_0$ would be to have either a loop or an infinite set of distinct labels. However, a loop cannot be obtained, since $<$ is only introduced by rule (\Box^-) between an object already present in the branch and a *new* object. Suppose there were an infinite descending chain $\dots < x_i \dots < x_0$. Each relation must be generated by a $\neg\Box\neg C$ that has not yet been used in the chain (indeed, if $\neg\Box\neg C$ had been previously used in the chain, say in introducing $x_i < x_{i-1}$, for each x_j such that $x_j < \dots < x_i$, $x_j : \Box\neg C$ is in \mathbf{B} , hence $x_j : \neg\Box\neg C$ cannot be in \mathbf{B} , otherwise \mathbf{B} would be closed, against the hypothesis). Notice however that the only formulas $\neg\Box\neg C$ that appear in the branch are derived from $\mathbf{T}(C)$ appearing in the initial constraint system. Since the number of such $\mathbf{T}(C)$ is finite, it follows that also the number of possible different $\neg\Box\neg C$ is finite, and the infinite descending chain cannot be generated. □

With the above propositions at hand, we can show that:

Theorem 3.9 (Soundness and Completeness of $T^{\mathcal{ALC}+\mathbf{T}}$) *Given a constraint system $\langle S \mid U \rangle$, it is unsatisfiable iff it has a closed tableau.*

Proof. As usual, the soundness can be proven by induction on the height of the closed tableau for $\langle S \mid U \rangle$. By considering each rule of $T^{\mathcal{ALC}+\mathbf{T}}$, it can be easily shown that if the premise is satisfiable, so is (at least) one of its conclusion. The theorem follows by contraposition.

Concerning the completeness, we show the contrapositive, i.e. that if the tableau is open, then the starting constraint system is satisfiable. An open tableau contains

an open branch that by Proposition 3.7 can be expanded into an open *saturated* branch. From such a branch, call it \mathbf{B} , we define a canonical model $\mathcal{M} = \langle \Delta_B, <', I \rangle$ where: $\Delta_B = \{x : x \text{ is a label appearing in } \mathbf{B}\}$; $<'$ is the transitive closure of relation $<$ in \mathbf{B} ; I is an interpretation function s. t. for all atomic concept $A \in ATM$, $A^I = \{x \text{ such that } x : A \text{ occurs in } \mathbf{B}\}$. I is then extended to all concepts C in the standard way, according to the semantics of the operators. For role names R , $R^I = \{(x, y) : \text{either } x \xrightarrow{R} y \text{ occurs in } \mathbf{B} \text{ or } x \text{ is blocked and } w \text{ is the witness of } x \text{ (by Lemma 3.5 such } w \text{ exists) and } w \xrightarrow{R} y \text{ occurs in } \mathbf{B}\}$. We can show that:

- $<'$ is *irreflexive, transitive, and satisfies the Smoothness Condition*. Irreflexivity follows from the fact that the relation $<$ is only introduced by rule (\Box^-) between an object already present in \mathbf{B} and a new object; transitivity follows from definition of $<'$; the Smoothness Condition follows from transitivity of $<'$ together with the finiteness of chains of $<$ deriving from Lemma 3.8.
- *for all concepts C we have: (a) if $x : C$ occurs in \mathbf{B} , then $x \in C^I$; (b) if $x : \neg C$ occurs in \mathbf{B} , then $x \in (\neg C)^I$* . We reason by induction on the complexity of C . If C is a boolean combination of concepts, the proof is simple and left to the reader.

If C is $\exists R.D$, then by saturation, either $x \xrightarrow{R} y$, $y : C$ occur in \mathbf{B} or $w \xrightarrow{R} y$, $y : C$ occur in \mathbf{B} , for w witness of x . In both cases, $(x, y) \in R^I$ by construction, and by inductive hypothesis $y \in C^I$, hence (a) follows. (b) can be proven similarly to the following case (a).

If C is $\forall R.D$, then by saturation, for all y s.t. $x \xrightarrow{R} y$ occurs in \mathbf{B} , also $y : D$ occurs in \mathbf{B} , and the (a) holds by the inductive hypothesis. (b) can be proven similarly to the previous case (a).

If C is $\Box \neg D$, and $x : \Box \neg D$ occurs in \mathbf{B} , let $x_i <' x$, then there is $x_i < \dots < x$, where each relation has been introduced by rule (\Box^-) . It can be easily shown that $x_i : \neg D$ occurs in \mathbf{B} , hence by inductive hypothesis, $x_i \in (\neg D)^I$, hence $x \in (\Box \neg D)^I$. If $x : \neg \Box \neg D$, then by saturation there is y s.t. $y : D$ occurs in \mathbf{B} and $y : \Box \neg D$ occurs in \mathbf{B} . Since $y <' x$, and by inductive hypothesis $y \in D^I$ and $y \in (\Box \neg D)^I$, it follows that $x \in (\neg \Box \neg D)^I$.

If C is $\mathbf{T}(D)$, and $x : \mathbf{T}(D)$ occurs in \mathbf{B} , by saturation, both $x : D$ and $x : \Box \neg D$ occur in \mathbf{B} , hence by inductive hypothesis $x \in D^I$ and $x \in (\Box \neg D)^I$, and by Proposition 2.8, $x \in (\mathbf{T}(D))^I$. If $x : \neg \mathbf{T}(D)$ occurs in \mathbf{B} , then by saturation also $x : \neg D$ occurs in \mathbf{B} or $x : \neg \Box \neg D$ occurs in \mathbf{B} . By inductive hypothesis either $x \notin D^I$ or $x \notin (\Box \neg D)^I$. In both cases, by Proposition 2.8, we conclude that $x \in (\neg \mathbf{T}(D))^I$.

- for all $C \sqsubseteq D \in U$, and all labels x , either $x \in (\neg C)^I$ or $x \in D^I$, i.e. $C^I \subseteq D^I$. By saturation either $x : \neg C$ occurs in \mathbf{B} or $x : D$ occurs in \mathbf{B} . The property immediately follows by inductive hypothesis.
- \mathcal{M} satisfies the starting constraint system: this is an immediate consequence of the previous points.

and the proof of completeness is over. \square

An example of derivation in $T^{\mathcal{ALC}+\mathbf{T}}$ is presented in Figure 2.

Let us conclude this section by analyzing termination and complexity of $T^{\mathcal{ALC}+\mathbf{T}}$.

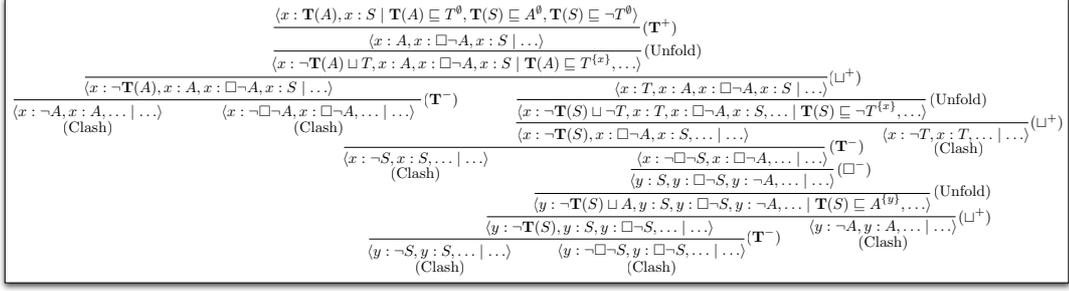


Fig. 2. A closed tableau showing that $\mathbf{T}(Adult) \sqsubseteq \neg Student$ can be inferred from $\{\mathbf{T}(Adult) \sqsubseteq TaxPayer, \mathbf{T}(Student) \sqsubseteq Adult, \mathbf{T}(Student) \sqsubseteq \neg TaxPayer\}$. It is shown that the KB whose TBox contains the above subsumptions, and whose ABox contains an x which is both a typical *Adult* and a *Student*, is unsatisfiable. To save space, we use A for *Adult*, T for *TaxPayer*, and S for *Student*.

In general, non-termination in labelled tableau calculi can be caused by two different reasons: 1. some rules copy their principal formula in the conclusion(s), and can thus be reapplied over the same formula without any control; 2. dynamic rules may generate infinitely-many labels, creating infinite branches.

Concerning the first source of non-termination (point 1), the only rules copying their principal formulas in their conclusions are (\forall^+) , (\exists^-) , (Unfold) , (\forall^-) , and (\exists^+) . However, the side conditions on these rules avoid multiple applications on the same formula. Indeed, (Unfold) can be applied to a constraint system $\langle S \mid U, C \sqsubseteq D^L \rangle$ 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). Concerning (\forall^+) , the rule can be applied to $\langle S, x : \forall R.C, x \xrightarrow{R} y \mid U \rangle$ only if $y : C$ does not belong to S . When $y : C$ is introduced in the branch, the rule will not further apply to $x : \forall R.C$. The same for (\exists^-) , (\exists^+) , and (\forall^-) .

Concerning the second source of non-termination (point 2), we can prove that we only need to adopt the standard loop-checking machinery known as *blocking*, which ensures that the rules (\exists^+) and (\forall^-) do not introduce infinitely-many labels on a branch. Thanks to the properties of \square , no other additional machinery is required to ensure termination. Indeed, we can show that the interplay between rules (\mathbf{T}^-) and (\square^-) does not generate branches containing infinitely-many labels. Let us discuss the termination in more detail.

Without the side conditions on the rules (\exists^+) and (\forall^-) , the calculus $T^{\mathcal{ALC}+\mathbf{T}}$ does not ensure a terminating proof search. Indeed, given a constraint system $\langle S \mid U \rangle$, it could be the case that (\exists^+) is applied to a constraint $x : \exists R.C \in S$, introducing a new label y and the constraints $x \xrightarrow{R} y$ and $y : C$. If an inclusion $\mathbf{T}(\exists R.C) \sqsubseteq D$ belongs to U , then (Unfold) can be applied by using y , thus generating a branch containing $y : \neg \mathbf{T}(\exists R.C)$, to which (\mathbf{T}^-) can be applied introducing $y : \neg \square \neg (\exists R.C)$. An application of (\square^-) introduces a new variable z and the constraint $z : \exists R.C$, to which (\exists^+) can be applied generating a new label u . (Unfold) can then be re-applied on $\mathbf{T}(\exists R.C) \sqsubseteq D$ by using u , incurring a loop. In order to prevent this source of non termination, we adopt the standard technique of *blocking*: the side condition of the (\exists^+) rule says that this rule can be applied to a node $\langle S, x : \exists R.C \mid U \rangle$ only if there is no z occurring in S such that $z \prec x$ and $z \equiv_{S, x : \exists R.C} x$. In other words, if there is an “older” label z which is equivalent to x with respect to $S, x : \exists R.C$, then (\exists^+) is not applicable, since the condition and

the strategy imply that the (\exists^+) rule has already been applied to z . In this case, we say that x is *blocked* by z . The same for (\forall^-) .

As mentioned, another possible source of infinite branches could be determined by the interplay between rules (\mathbf{T}^-) and (\square^-) . This cannot occur, i.e. the interplay between these two rules does not generate branches containing infinitely-many labels. Intuitively, the application of (\square^-) to $x : \neg\square\neg C$ adds $y : \square\neg C$ to the conclusion, so that (\mathbf{T}^-) can no longer consistently introduce $y : \neg\square\neg C$. This is due to the properties of \square (no infinite descending chains of $<$ are allowed). More in detail, if (Unfold) is applied to $\mathbf{T}(C) \sqsubseteq D$ by using x , an application of (\mathbf{T}^-) introduces a branch containing $x : \neg\square\neg C$; when a new label y is generated by an application of (\square^-) on $x : \neg\square\neg C$, we have that $y : \square\neg C$ is added to the current constraint system. If (Unfold) and (\mathbf{T}^-) are also applied to $\mathbf{T}(C) \sqsubseteq D$ on the new label y , then the conclusion where $y : \neg\square\neg C$ is introduced is closed, by the presence of $y : \square\neg C$. By this fact, we do not need to introduce any loop-checking machinery on the application of (\square^-) .

Let us now show that $T^{\mathcal{ALC}+\mathbf{T}}$ ensures a terminating proof search. In order to prove this result in a rigorous manner, we proceed as follows: first, we introduce the complexity measure on a constraint system $\langle S \mid U \rangle$ of Definition 3.13, denoted by $m(\langle S \mid U \rangle)$, which consists of seven measures $c_1, c_2, c_3, c_4, c_5, c_6$ and c_7 in a lexicographic order, and the auxiliary Definitions 3.10 and 3.12; then, we prove that each application of $T^{\mathcal{ALC}+\mathbf{T}}$'s rules reduces this measure, until the rules are no longer applicable, or their application leads to a closed tableau.

Definition 3.10 [Complexity of a constraint] We define the complexity of a constraint as follows:

- $cp(x : A) = 1$, if A is an atom;
- $cp(x \xrightarrow{R} y) = 1$;
- $cp(x : \neg C) = 1 + cp(x : C)$;
- $cp(x : C \otimes D) = 1 + cp(x : C) + cp(x : D)$, with $\otimes \in \{\sqcup, \sqcap\}$;
- $cp(x : \square\neg C) = 2 + cp(x : C)$;
- $cp(x : \odot R.C) = 1 + cp(x : C)$, with $\odot \in \{\forall, \exists\}$;
- $cp(x : \mathbf{T}(C)) = 3 + 2cp(x : C)$.

Given a constraint system $\langle S \mid U \rangle$, we call $n_{\langle S \mid U \rangle}$ the number of extended concepts appearing in $\langle S \mid U \rangle$, including also all the concepts appearing as a substring of another concept. We can prove that:

Proposition 3.11 *In any set of variables in S including more than $2^{n_{\langle S \mid U \rangle}}$ variables there are at least two variables x and y s.t. $x \equiv_S y$.*

Proof. Since there are $n_{\langle S \mid U \rangle}$ extended concepts, given a variable x there cannot be more than $2^{n_{\langle S \mid U \rangle}}$ different sets of constraints $x : C$ in S . As a consequence, in S there are at most $2^{n_{\langle S \mid U \rangle}}$ non-blocked variables. \square

We use square brackets $[\dots]$ to denote *multisets*.

Definition 3.12 Given an initial constraint system $\langle S_0 \mid U_0 \rangle$, we define:

- the set $\mathcal{L}_{\square+}^{S_0}$ of boxed formulas $\square\neg C$ that can be generated in a tableau for $\langle S_0 \mid U_0 \rangle$, i.e. $\mathcal{L}_{\square+}^{S_0} = \{\square\neg C \mid \mathbf{T}(C) \sqsubseteq D \in U_0\}$. We let $n_0 = |\mathcal{L}_{\square+}^{S_0}|$;
- the *multiset* $\mathcal{L}_{\square-}^{S_0}$ of negated boxed formulas that can be generated in a tableau for $\langle S_0 \mid U_0 \rangle$, i.e. $\mathcal{L}_{\square-}^{S_0} = [\neg\square\neg C \mid \mathbf{T}(C) \sqsubseteq D \in U_0]$. We let $k_0 = |\mathcal{L}_{\square-}^{S_0}|$;
- the *multiset* $\mathcal{L}_{\exists}^{S_0}$ of existential formulas $\exists R.C$ that can be generated in a tableau for $\langle S_0 \mid U_0 \rangle$. We let $e_0 = |\mathcal{L}_{\exists}^{S_0}|$;
- the *multiset* $\mathcal{L}_{\neg\forall}^{S_0}$ of negated \forall formulas $\neg\forall R.C$ that can be generated in a tableau for $\langle S_0 \mid U_0 \rangle$. We let $a_0 = |\mathcal{L}_{\neg\forall}^{S_0}|$.

Given a label x and a constraint system $\langle S \mid U \rangle$ in the tableau for the initial constraint system $\langle S_0 \mid U_0 \rangle$, we define:

- the number n_x of positive boxed formulas $\square\neg C$ not labelled by x , i.e. $n_x = n_0 - |\{\square\neg C \in \mathcal{L}_{\square+}^{S_0} \mid x : \square\neg C \in S\}|$;
- the number k_x of negated boxed formulas $\neg\square\neg C$ not yet expanded in a label x , i.e. $k_x = k_0 - |[\neg\square\neg C \in \mathcal{L}_{\square-}^{S_0} \mid x : \neg\square\neg C \text{ has been expanded}]|$ ⁷
- the maximum number e_{max} of existential formulas $x : \exists R.C$ that can be labelled by non-blocked variables, i.e. $e_{max} = 2^{n_{\langle S_0 \mid U_0 \rangle}} \times e_0$
- the maximum number a_{max} of negated \forall formulas $x : \neg\forall R.C$ that can be labelled by non-blocked variables, i.e. $a_{max} = 2^{n_{\langle S_0 \mid U_0 \rangle}} \times a_0$

Definition 3.13 [Lexicographic order] Given a constraint system $\langle S \mid U \rangle$, we define $m(\langle S \mid U \rangle) = \langle c_1, c_2, c_3, c_4, c_5, c_6, c_7 \rangle$ where:

- $c_1 = e_{max} - | [x : \exists R.C \mid \exists R.C \in \mathcal{L}_{\exists}^{S_0} \text{ and } x \text{ is a variable in } S \text{ and } x : \exists R.C \text{ has been expanded}] |$, i.e. c_1 is the number of formulas $x : \exists R.C$ that can still be expanded
- $c_2 = a_{max} - | [x : \neg\forall R.C \mid \neg\forall R.C \in \mathcal{L}_{\neg\forall}^{S_0} \text{ and } x \text{ is a variable in } S \text{ and } x : \neg\forall R.C \text{ has been expanded}] |$, i.e. c_2 is the number of formulas $x : \neg\forall R.C$ that can still be expanded
- c_3 is the multiset given by $[c_3^{x_1}, c_3^{x_2}, \dots, c_3^{x_l}]$, where x_1, x_2, \dots, x_l are the labels occurring in S and, given a label x , c_3^x is a pair (n_x, k_x) in a lexicographic order (n_x and k_x are defined as in Definition 3.12). We consider the integer multiset ordering given by c_3
- $c_4 = | \{(x, C \sqsubseteq D) \mid x \text{ occurs in } S \text{ and } C \sqsubseteq D^L \in U \text{ and } x \notin L\} |$
- $c_5 = | \{x \xrightarrow{R} y \in S \mid x : \forall R.C \in S \text{ and } y : C \notin S\} |$
- $c_6 = | \{x \xrightarrow{R} y \in S \mid x : \neg\exists R.C \in S \text{ and } y : \neg C \notin S\} |$
- $c_7 = \sum_{F \in S} cp(F)$, where F is a constraint $x : C$ or $x \xrightarrow{R} y$.

We consider the lexicographic order given by $m(\langle S \mid U \rangle)$.

⁷ Notice that, in case there are two relations $\mathbf{T}(C) \sqsubseteq D$ and $\mathbf{T}(C) \sqsubseteq E$, then the multiset $\mathcal{L}_{\square-}^{S_0}$ contains two instances of $\neg\square\neg C$. Therefore, if the rule (\square^-) is applied to $x : \neg\square\neg C$ (for instance, generated by an application of (\mathbf{T}^-) following an application of (Unfold) in x on $\mathbf{T}(C) \sqsubseteq D$), then k_x decreases only by 1 unit, whereas the second instance of $\neg\square\neg C$, i.e. the one “associated” with $\mathbf{T}(C) \sqsubseteq E$, is still considered to be *not expanded* in x , thus it still “contributes” to k_x .

First of all, we prove that the application of any rule of $T^{\mathcal{ALC}+\mathbf{T}}$ reduces $m(\langle S \mid U \rangle)$ or leads to a node which is an instance of (Clash), as stated by the following Lemma:

Lemma 3.14 *Let $\langle S' \mid U' \rangle$ be a constraint system obtained as a conclusion of an application of a rule of $T^{\mathcal{ALC}+\mathbf{T}}$ to a constraint system $\langle S \mid U \rangle$. We have that either the tableau for $\langle S' \mid U' \rangle$ is closed, i.e. $\langle S' \mid U' \rangle$ is an instance of (Clash), or $m(\langle S' \mid U' \rangle) < m(\langle S \mid U \rangle)$.*

Proof. We only present the most interesting cases of (\exists^+) and (\Box^-) . The other ones are easy and then left to the reader.

(\exists^+) : suppose the rule is applied to $\langle S, x : \exists R.C \mid U \rangle$, leading to a conclusion $\langle S, x : \exists R.C, x \xrightarrow{R} y, y : C \mid U \rangle$ where y is new. x is a non-blocked variable, otherwise the rule would not be applicable (its side condition would not be fulfilled). We can conclude that c_1 decreases, since $x : \exists R.C$ has been expanded in x , thus increasing the cardinality of the multiset $[x : \exists R.C \mid \exists R.C \in \mathcal{L}_{\exists}^{S_0}]$ and x is a variable in S and $x : \exists R.C$ has been expanded];

(\Box^-) : first of all, observe that c_1 and c_2 are not augmented in the conclusion. The application of (\Box^-) reduces c_3 or leads to a closed tableau. An application of (\Box^-) to $\langle S, x : \Box \neg C \mid U \rangle$ leads to $\langle S, y : C, y : \Box \neg C, S_{x \rightarrow y}^M \mid U \rangle$, where y is a new label. c_3 in the premise, say c_{3_p} , is a multiset $[\dots, c_{3_p}^x, \dots]$, whereas in the conclusion we have to consider a measure, called c_{3_c} , of the form $[\dots, c_{3_c}^x, c_{3_c}^y, \dots]$. By the standard definition of integer multiset ordering, we prove that either $c_{3_c} < c_{3_p}$ (by showing that $c_{3_c}^x < c_{3_p}^x$ and $c_{3_c}^y < c_{3_p}^x$, i.e. we replace an integer $c_{3_p}^x$ with two smaller integers), or that the procedure leads to a closed tableau. We conclude the proof as follows. First case: let us consider $c_{3_c}^y$ and $c_{3_p}^x$. By definition, $c_{3_c}^y$ is a pair (n_{y_c}, k_{y_c}) and $c_{3_p}^x$ is a pair (n_{x_p}, k_{x_p}) . We distinguish two cases: if $x : \Box \neg C \notin \Gamma$, then we easily prove that $n_{y_c} < n_{x_p}$, since $y : \Box \neg C$ belongs to the conclusion: therefore, the number of boxed formulas $\Box \neg C$ not occurring with label y is smaller than the number of boxed formulas not occurring with label x , and we are done (remember that all the positive boxed formulas labelled by x are also labelled by y in the conclusion, by the presence of $S_{x \rightarrow y}^M$); if $x : \Box \neg C \in \Gamma$, then the application of (\Box^-) leads to a node containing both $y : C$ and $y : \neg C$ and, which is an instance of (Clash). Second case: let us consider $c_{3_c}^x$ and $c_{3_p}^x$. It is easy to observe that $n_{x_p} = n_{x_c}$, since no formula $x : \Box \neg C$ is added nor removed in the conclusion (the positive boxed formulas labelled by x are the same in both the premise and the conclusion). We conclude since $k_{x_c} < k_{x_p}$, since $x : \neg \Box \neg C$ has been expanded in x , so it “contributes” to k_{x_p} whereas it does not to k_{x_c} . □

Now we can prove that $T^{\mathcal{ALC}+\mathbf{T}}$ ensures a terminating proof search:

Theorem 3.15 (Termination of $T^{\mathcal{ALC}+\mathbf{T}}$) *Let $\langle S \mid U \rangle$ be a constraint system, then any tableau generated by $T^{\mathcal{ALC}+\mathbf{T}}$ is finite.*

Proof. Let $\langle S' \mid U' \rangle$ be obtained by an application of a rule of $T^{\mathcal{ALC}+\mathbf{T}}$ to a premise $\langle S \mid U \rangle$. By Lemma 3.14 we have that either $\langle S' \mid U' \rangle$ is an instance of (Clash) (and in this case we are done) or we have that $m(\langle S' \mid U' \rangle) < m(\langle S \mid U \rangle)$. This means that a finite number of applications of the rules leads either to a node containing a clash

or to a node whose measure is as follows: $\langle 0, 0, [(0, 0), (0, 0), \dots, (0, 0)], 0, 0, 0, c_{7_{min}} \rangle$, where $c_{7_{min}}$ is the minimal value that c_7 can assume for $\langle S \mid U \rangle$. This means that no rule of $T^{\mathcal{ALC}+\mathbf{T}}$ is further applicable. The rules (\exists^+) and (\forall^-) are no longer applicable. Indeed, suppose on the contrary that the current node has the form $\langle S'', x : \exists R.C \mid U \rangle$, to which apply the (\exists^+) rule. If x is a non-blocked variable, then $x : \exists R.C$ has been already expanded in the current branch, otherwise $c_1 > 0$. Therefore, it can only be the case that x is a blocked variable, then the (\exists^+) rule is no longer applicable by its side condition. The same for (\forall^-) since $c_2 = 0$. The (\square^-) rule is no longer applicable too: indeed, we can observe that there is no $x : \neg \square \neg C \in S$, by the fact that $c_3 = [(0, 0), \dots, (0, 0)]$, then $c_3^x = (0, 0)$, thus $k_x = 0$ (remember that k_x is the number of $\neg \square \neg C$ not expanded in x). The rule (Unfold) is no longer applicable: since $c_4 = 0$, given any $C \sqsubseteq D^L \in U$ and any label x in the tableau, we have that $x \in L$, therefore the side condition on the applicability of the rule does no longer hold. The rule (\forall^+) is no longer applicable: indeed, since $c_5 = 0$, we have that, for all $x \xrightarrow{R} y \in S$ and $x : \forall R.C \in S$, we have that $y : C \in S$, and the side condition of the rule can no longer be satisfied. The same for (\exists^-) , since $c_6 = 0$. Finally, since c_7 assumes its *minimal* value $c_{7_{min}}$, (\mathbf{T}^+) , (\mathbf{T}^-) , and the boolean rules are no longer applicable. If one of them was applicable, then its application would reduce the value of c_7 in its conclusion(s) by Lemma 3.14, against the minimality of $c_{7_{min}}$ in the premise. \square

Since the calculus $T^{\mathcal{ALC}+\mathbf{T}}$ is sound and complete (Theorem 3.9), and since an $\mathcal{ALC} + \mathbf{T}$ -knowledge base is satisfiable iff its corresponding constraint system is satisfiable (Proposition 3.3), from Theorem 3.15 above it immediately follows that:

Theorem 3.16 (Decidability) *Checking whether a given $\mathcal{ALC} + \mathbf{T}$ -knowledge base $(TBox, ABox)$ is satisfiable is a decidable problem.*

Let us conclude this section with a complexity analysis of the calculus $T^{\mathcal{ALC}+\mathbf{T}}$:

Theorem 3.17 (Complexity) *Given an $\mathcal{ALC} + \mathbf{T}$ -knowledge base $(TBox, ABox)$, the problem of checking whether it is satisfiable can be solved in nondeterministic exponential time.*

Proof. We first show that the number of labels generated on a branch is at most exponential in the size of KB. Let n be the size of a KB. Given a constraint system $\langle S \mid U \rangle$, the number of extended concepts appearing in $\langle S \mid U \rangle$, including also all the ones appearing as a subformula of other concepts, is $O(n)$. As there are at most $O(n)$ concepts, there are at most $O(2^n)$ variables labelling distinct sets of concepts. Hence, there are $O(2^n)$ non-blocked variables in S .

Let m be the maximum number of direct successors of each variable $x \in S$, obtained by applying dynamic rules. m is bound by the number of $\exists R.C$ concepts ($O(n)$) plus the number of $\neg \forall R.C$ concepts ($O(n)$) plus the number of $\neg \square \neg C$ concepts ($O(n)$). Therefore, there are at most $O(2^n \times m)$ variables in S , where $m \leq 3n$. The number of *individuals* in the ABox is bound by n too, and each individual has at most m direct successors. The number of *labels* in S is then bound by $O((2^n + n) \times m)$, and hence by $O(2^{2n})$.

For a given label x , the concepts labelled by x introduced in the branch (namely, all the possible subconcepts of the initial constraint system, as well as all boxed subconcepts) are $O(n)$. According to the standard strategy, after all static rules have been applied to a label x in phase 1, no other concepts labelled by x can be introduced later on a branch. Hence, the labelled concepts introduced on the branch is $O(n)$ for each label, and the number of all labelled concepts on the branch is $O(n \times 2^{2n})$. Therefore, a branch can contain at most an exponential number of applications of tableau rules.

The satisfiability of a KB can thus be solved by defining a procedure which nondeterministically generates an open branch of exponential size (in the size of KB). The problem is in NEXPTIME. \square

Although we have shown that the satisfiability is in NEXPTIME, we can show that this bound is not optimal; in detail, we can show that the problem of deciding whether an $\mathcal{ALC} + \mathbf{T}$ knowledge base $\text{KB}=(\text{TBox}, \text{Abox})$ is satisfiable is in EXPTIME.

3.1 Complexity of $\mathcal{ALC} + \mathbf{T}$

As a first step, we define the language \mathcal{L}_{KB} of all subformulas of KB plus all boxed formulas $\Box\neg C$ such that $\mathbf{T}(C)$ occurs in KB. Formally:

Definition 3.18 Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $\text{KB}=(\text{TBox}, \text{Abox})$, we define the language \mathcal{L}_{KB} as follows:

- if C is a subformula of KB, then $C \in \mathcal{L}_{KB}$ and $\neg C \in \mathcal{L}_{KB}$;
- if $\mathbf{T}(C)$ occurs in KB, then $\Box\neg C \in \mathcal{L}_{KB}$ and $\neg\Box\neg C \in \mathcal{L}_{KB}$.

Let us now define the *saturation* of a set A of formulas of \mathcal{L}_{KB} . Intuitively, given a set of formulas A of \mathcal{L}_{KB} , we say that it is saturated if it is consistent and it contains a maximal number of formulas, i.e. for all $C \in \mathcal{L}_{KB}$, either $C \in A$ or $\neg C \in A$. Formally:

Definition 3.19 Let $\text{KB}=(\text{TBox}, \text{Abox})$ be an $\mathcal{ALC} + \mathbf{T}$ knowledge base, and let A be a set of formulas of \mathcal{L}_{KB} , i.e. $A \subseteq \mathcal{L}_{KB}$. We say that A is saturated if the following conditions hold:

- for each $C \in \mathcal{L}_{KB}$, then either $C \in A$ or $\neg C \in A$;
- if $\neg C \in \mathcal{L}_{KB}$, then $C \in A$ if and only if $\neg C \notin A$;
- if $C \sqcup D \in \mathcal{L}_{KB}$, then $C \sqcup D \in A$ if and only if $C \in A$ or $D \in A$;
- if $C \sqcap D \in \mathcal{L}_{KB}$, then $C \sqcap D \in A$ if and only if $C \in A$ and $D \in A$;
- if $\mathbf{T}(C) \in \mathcal{L}_{KB}$, then $\mathbf{T}(C) \in A$ if and only if $C \in A$ and $\Box\neg C \in A$;
- if $\neg\mathbf{T}(C) \in \mathcal{L}_{KB}$, then $\neg\mathbf{T}(C) \in A$ if and only if $\neg C \in A$ or $\neg\Box\neg C \in A$;
- if $C \sqsubseteq D \in \text{TBox}$, then if $C \in A$ then $D \in A$.

We want to prove that, given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $\text{KB}=(\text{TBox}, \text{Abox})$ such that the TBox is consistent and the ABox is consistent w.r.t. the TBox, we

can build a canonical model for KB by means of a construction whose temporal complexity is exponential in the size of the KB. The basic idea is that the domain of the canonical model contains all sets $A \subseteq \mathcal{L}_{KB}$ which are saturated in the sense of Definition 3.19. To this aim, we define the set of all saturated sets of formulas of \mathcal{L}_{KB} :

Definition 3.20 Let $KB=(TBox, Abox)$ be an $\mathcal{ALC} + \mathbf{T}$ knowledge base, we define

$$At(\mathcal{L}_{KB}) = \{ A \subseteq \mathcal{L}_{KB} \mid A \text{ is saturated (Definition 3.19)} \}$$

Now we can build a canonical model for a given $\mathcal{ALC} + \mathbf{T}$ knowledge base:

Definition 3.21 Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $KB=(TBox, Abox)$, we define a *canonical* model $\mathcal{M} = \langle \Delta, I, < \rangle$ as follows:

- $\Delta = At(\mathcal{L}_{KB})$
- extension function I :
 - *concepts*: given $A \in \Delta$, we have that $A \in C^I$ if and only if $C \in A$;
 - *roles*: given $A \in \Delta$ and $B \in \Delta$, we have that $(A, B) \in R^I$ if and only if there exists $\exists R.C \in A$ (resp. $\neg \forall R.C \in A$) such that:
 - (i) $C \in B$ (resp. $\neg C \in B$);
 - (ii) for all $\forall R.D \in A$ (resp. for all $\neg \exists R.D \in A$), then $D \in B$ (resp. $\neg D \in B$);
- preference relation $<$:
 - given $A \in \Delta$ and $B \in \Delta$, we define $A <_0 B$ if and only if there exists $\neg \Box \neg C \in B$ such that:
 - (i) $C \in A$ and $\Box \neg C \in A$;
 - (ii) for each $\Box \neg D \in B$, then $D \in A$ and $\Box \neg D \in A$;
 - $<$ is the transitive closure of $<_0$.

In order to show that the canonical model of Definition 3.21 can be extended to a model of KB, we need the following lemmas.

Lemma 3.22 *Let $KB=(TBox, Abox)$ be an $\mathcal{ALC} + \mathbf{T}$ knowledge base and let $\mathcal{M} = \langle \Delta, I, < \rangle$ be the canonical model of Definition 3.21. Given $A \in \Delta$ and given an \mathcal{ALC} concept C , then the extension function I is an extension function in a model as in Definition 2.1, i.e. given a concept $C \in \mathcal{L}_{KB}$, if $A \in C^I$, then we have that:*

- if $C = D \sqcup E$, then either $A \in D^I$ or $A \in E^I$;
- if $C = D \sqcap E$, then $A \in D^I$ and $A \in E^I$;
- if $C = \neg D$, then $A \notin D^I$.

Proof. Consider the case $C = D \sqcap E$: given $A \in C^I$, by Definition 3.21 we have that $C \sqcap E \in A$ and, since A is saturated, we can observe that $D \in A$ and $E \in A$. Again by the construction of the canonical model in Definition 3.21, we have that $A \in D^I$ and $A \in E^I$, and we are done. The case in which $C = D \sqcup E$ is similar and left to the reader. Consider the case $C = \neg D$: given $A \in C^I$, by Definition 3.21 we have that $\neg D \in A$ and, since A is saturated, we conclude that $D \notin A$. We conclude that $A \notin D^I$ by the construction of the canonical model in Definition 3.21. \square

Lemma 3.23 *Let $KB=(TBox, Abox)$ be an $\mathcal{ALC} + \mathbf{T}$ knowledge base and let $\mathcal{M} = \langle \Delta, I, < \rangle$ be the canonical model of Definition 3.21. Given $A \in \Delta$, if $\Box \neg C \in A$,*

then for all $B < A$ we have that $\neg C \in B$ and $\Box\neg C \in B$.

Proof. We distinguish two cases:

- $B <_0 A$: by construction, for all $\Box\neg C \in A$ we have that $\neg C \in B$ and $\Box\neg C \in B$ and we are done;
- $B < A$ has been obtained by the transitive closure of $<_0$ with $B <_0 A_1 <_0 A_2 <_0 \dots <_0 A_{n-1} <_0 A_n <_0 A$. By construction, for all $\Box\neg C \in A$, we have that $\neg C \in A_n$ and $\Box\neg C \in A_n$. For the same reason, we have that $\neg C \in A_{n-1}$ and $\Box\neg C \in A_{n-1}$, and so on, then $\neg C \in B$ and $\Box\neg C \in B$, and we are done. \square

Lemma 3.24 *Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $KB=(TBox, Abox)$ and given the canonical model $\mathcal{M} = \langle \Delta, I, < \rangle$ of Definition 3.21, we have that $(\mathbf{T}(C))^I = Min_{<}(C^I)$.*

Proof.

- $(\mathbf{T}(C))^I \subseteq Min_{<}(C^I)$: given $A \in (\mathbf{T}(C))^I$, we have to show that $A \in Min_{<}(C^I)$. By the definition of the extension function I in Definition 3.21, we have that $\mathbf{T}(C) \in A$ since $A \in (\mathbf{T}(C))^I$. Since A is saturated (Definition 3.19), we have that $C \in A$ and $\Box\neg C \in A$, then (ii) $A \in C^I$ by construction of I in the model. Let us consider each $B \in \Delta$ such that $B < A$: since $\Box\neg C \in A$, by Lemma 3.23 we have that $\neg C \in B$ then, by the definition of the extension function I in the model, we have that (ii) $B \notin C^I$. We can conclude that (i) $A \in C^I$ and (ii) for each $B < A$ we have $B \notin C^I$, i.e. $A \in Min_{<}(C^I)$;
- $Min_{<}(C^I) \subseteq (\mathbf{T}(C))^I$: consider $A \in Min_{<}(C^I)$. By definition of $Min_{<}$, we have that $A \in C^I$ and there is not $B < A$ such that $B \in C^I$. Since $A \in C^I$, we have that (i) $C \in A$ by the definition of I in Definition 3.21. By absurd, suppose that $\Box\neg C \notin A$: since A is saturated (Definition 3.19), $\neg\Box\neg C \in A$. By construction of the canonical model, there exists $B' < A$ such that $C \in B'$ and $\Box\neg C \in B'$ then, by the definition of I , $B' \in C^I$. Therefore, there exists $B' < A$ such that $B' \in C^I$, against the hypothesis that $A \in Min_{<}(C^I)$. We can conclude that (ii) $\Box\neg C \in A$. Since A is saturated (Definition 3.19), $\mathbf{T}(C) \in A$. We conclude that $A \in (\mathbf{T}(C))^I$ by the definition of I in Definition 3.21. \square

Lemma 3.25 *Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $KB=(TBox, Abox)$ and given the canonical model $\mathcal{M} = \langle \Delta, I, < \rangle$ of Definition 3.21, we have that $<$ satisfies the smoothness condition.*

Proof. We have to show that, given a set S , if $S \neq \emptyset$, then also $Min_{<}(S) \neq \emptyset$. We have to prove that $<$ does not contain infinite descending chains of elements of Δ ; since $\Delta = At(\mathcal{L}_{KB})$ is finite, then we have only to show that $<$ is acyclic. By absurd, suppose that $B < A_1 < A_2 < \dots < A_n < A < B$, for some $A, B, A_1, A_2, \dots, A_n \in \Delta$. Since $A < B$, there is $\neg\Box\neg C \in B$ such that $\Box\neg C \in A$. By Lemma 3.23, for all $B' < A$ we have that $C \in B'$ and $\Box\neg C \in B'$, then also $\Box\neg C \in B$ since $B < A$. Since $B \in \Delta$, B is saturated, in contrast with the fact that both $\neg\Box\neg C \in B$ and

$\Box\neg C \in B$. □

Lemma 3.26 *Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $KB=(TBox, ABox)$ and given the canonical model $\mathcal{M} = \langle \Delta, I, < \rangle$ of Definition 3.21, we have that $<$ is an irreflexive and transitive relation.*

Proof. Transitivity follows from the definition of $<$ from $<_0$. For the irreflexivity, we proceed by absurd. By construction, given $A \in \Delta$ and $B \in \Delta$, we have that if $B < A$ then there exists $\neg\Box\neg C \in A$ such that $C \in B$ and $\Box\neg C \in B$. Suppose $B < B$, then there is $\neg\Box\neg C \in B$ and $\Box\neg C \in B$, against the saturation of B . □

Now we show that the canonical model of Definition 3.21 can be extended in order to obtain a model for a given $KB=(TBox, ABox)$. Intuitively, we extend I in a way such that each individual a explicitly mentioned in the $ABox$ is mapped to an element $A \in \Delta$ such that $C \in A$ for each $C(a) \in ABox$. Moreover, for each $R(a, b) \in ABox$, we extend I in a way such that the element of Δ to which a is mapped is related to the element of Δ to which b is mapped by R^I .

We need a further definition:

Definition 3.27 *Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $KB=(TBox, ABox)$ and given an individual a explicitly mentioned in the $ABox$, we define $\sigma(a)$ the set of concepts of which a is an instance, i.e.*

$$\sigma(a) = \{C \in \mathcal{L}_{KB} \mid C(a) \in ABox\}$$

Lemma 3.28 *Let $KB=(TBox, ABox)$ be an $\mathcal{ALC} + \mathbf{T}$ knowledge base, where the $TBox$ is consistent and the $ABox$ is consistent w.r.t. the $TBox$, and let $\mathcal{M} = \langle \Delta, I, < \rangle$ be a canonical model as in Definition 3.21. The following properties hold:*

- (i) *given an individual a explicitly mentioned in the $ABox$, then there exists $A \in \Delta$ such that $\sigma(a) \subseteq A$;*
- (ii) *if $R(a, b) \in ABox$, there exist $A \in \Delta$ such that $\sigma(a) \subseteq A$, $B \in \Delta$ such that $\sigma(b) \subseteq B$ such that, for all $\forall R.C \in A$ (resp. for all $\neg\exists R.C \in A$), we have that $C \in B$ (resp. $\neg C \in B$).*

Proof.

- (i) For absurd, suppose that there is an individual a in the $ABox$ such that there is not any $A \in \Delta$ with $\sigma(a) \subseteq A$. Since $\sigma(a) \subseteq \mathcal{L}_{KB}$, if there is no $A \in \Delta$ with $\sigma(a) \subseteq A$, then we can observe that, for all $B \subseteq \mathcal{L}_{KB}$ such that $\sigma(a) \subseteq B$, $B \notin At(\mathcal{L}_{KB})$, that is to say B is not saturated. Suppose B is not saturated only because $C \sqcup D \in B$, $C \in B$ but $\neg D \in B$. In this case, we can consider $B' \subseteq \mathcal{L}_{KB}$, different from B only because $D \in B'$, whereas $\neg D \notin B'$. We can observe that $B' \in At(\mathcal{L}_{KB})$. If $\sigma(a) \not\subseteq B'$, this means that $D \notin \sigma(a)$, therefore $(C \sqcup D)(a)$ and $C(a)$ can be inferred by the $ABox$, whereas $D(a)$ cannot, against the consistency of the $ABox$. The other cases are similar and left to the reader.
- (ii) For absurd, suppose that there is $R(a, b) \in ABox$ not satisfying the property. This means that, for each $A \in \Delta$ such that $\sigma(a) \subseteq A$:
 - there is a concept $\forall R.C \in A$ (resp. $\neg\exists R.C \in A$);

- for all $B \in \Delta$ such that $\sigma(b) \subseteq B$, we have that $\neg C \in B$ (resp. $C \in B$).
One can choose $A' \in \Delta$, differing from A only because $\neg \forall R.C \in A'$ (then, $\forall R.C \notin A'$), rather than A . If this was not possible, this means that $\sigma(a) \not\subseteq A'$, i.e. $(\forall R.C)(a)$ is in the ABox. One can then think of choosing $B' \in \Delta$, differing from B only because $C \in B'$ (then, $\neg C \notin B'$), rather than B . If this was not possible, it must be $\sigma(b) \not\subseteq B'$, i.e. $(\neg C)(b) \in \text{ABox}$. Since we have that $R(a,b) \in \text{ABox}$, $(\forall R.C)(a) \in \text{ABox}$, and $(\neg C)(b) \in \text{ABox}$, we can conclude that the ABox is inconsistent, against the hypothesis.

□

Now we can define a model \mathcal{M}' of a KB:

Definition 3.29 Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $\text{KB}=(\text{TBox}, \text{ABox})$, where the TBox is consistent and the ABox is consistent w.r.t. the TBox, and given the canonical model \mathcal{M} of Definition 3.21, we define the model $\mathcal{M}' = \langle \Delta, I', \langle \rangle \rangle$ as follows:

- Δ and $\langle \rangle$ are defined as in Definition 3.21;
- I' is defined as in Definition 3.21 and is extended:
 - by assigning to each individual a of the ABox an element $A \in \Delta$, say it $a^{I'}$, such that:
 - $\sigma(a) \subseteq A$
 - for all $R(a,b) \in \text{ABox}$ and for all $\forall R.C \in A$ (resp. $\neg \exists R.C \in A$) we have $C \in B$ (resp. $\neg C \in B$), where $B = b^{I'}$, and
 - by adding $(a^{I'}, b^{I'}) \in R^{I'}$ for each $R(a,b) \in \text{ABox}$.

The addition of $(a^{I'}, b^{I'}) \in R^{I'}$ for a given $R(a,b) \in \text{ABox}$ does not affect the construction of the model of a given KB, that is to say:

Lemma 3.30 Let $\text{KB}=(\text{TBox}, \text{ABox})$ be an $\mathcal{ALC} + \mathbf{T}$ knowledge base, where the TBox is consistent and the ABox is consistent w.r.t. the TBox, and let $\mathcal{M}' = \langle \Delta, I', \langle \rangle \rangle$ be the model of Definition 3.29. Given $A \in \Delta$ and $B \in \Delta$, then if $(A, B) \in R^{I'}$ and $A \in (\forall R.C)^{I'}$ (resp. $A \in (\neg \exists R.C)^{I'}$), then also $B \in C^{I'}$ (resp. $B \in (\neg C)^{I'}$).

Proof. We distinguish two cases:

- $(A, B) \in R^I$: by construction, this means that, for all $\forall R.C \in A$ (resp. for all $\neg \exists R.C \in A$) we have that $C \in B$ (resp. $\neg C \in B$). By construction of \mathcal{M}' , we can conclude that $B \in C^{I'}$ (resp. $B \in (\neg C)^{I'}$);
- $(A, B) \in R^{I'}$ since $R(a,b) \in \text{ABox}$, and $A = a^{I'}$ and $B = b^{I'}$: this immediately follows from part 2. of Lemma 3.28.

□

We are able to prove that:

Theorem 3.31 Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $\text{KB}=(\text{TBox}, \text{ABox})$, where the TBox is consistent and the ABox is consistent w.r.t. the TBox, the model $\mathcal{M}' = \langle \Delta, I', \langle \rangle \rangle$ of Definition 3.29 satisfies KB.

Proof. First of all, Lemmas 3.22 and 3.24 show that I' is an extension function respecting Definition 2.1. Moreover, $<$ is an irreflexive and transitive relation (Lemma 3.26) and satisfies the smoothness condition (Lemma 3.25). We have to show that both the TBox and the ABox are satisfiable in \mathcal{M}' :

- TBox: for each $C \sqsubseteq D \in \text{TBox}$, we have to show that, for each $A \in \Delta$, if $A \in C^{I'}$ then also $A \in D^{I'}$. Suppose $A \in C^{I'}$. By construction, we have that $C \in A$. Since A is saturated, we have also that $D \in A$ and, by construction of the model, $A \in D^{I'}$, and we are done;
- ABox: we have to show that, for each $C(a) \in \text{ABox}$ and for each $R(a, b) \in \text{ABox}$, we have that $a^I \in C^{I'}$ and $(a^I, b^I) \in R^{I'}$, respectively. This immediately follows from Lemmas 3.28 and 3.30 and from the definition of I' in Definition 3.29. \square

We conclude this section with a complexity analysis of $\mathcal{ALC} + \mathbf{T}$:

Theorem 3.32 *Given an $\mathcal{ALC} + \mathbf{T}$ knowledge base $KB = (\text{TBox}, \text{ABox})$, the problem of deciding satisfiability of KB is in EXPTIME.*

Proof. If KB is satisfiable, we can build a model as defined in Definition 3.29. Let n be the length of the string representing KB : the cardinality of the language \mathcal{L}_{KB} (Definition 3.18) is $O(n)$, then the number of different sets of formulas of \mathcal{L}_{KB} is $O(2^n)$. Also the cardinality of $\text{At}(\mathcal{L}_{KB})$ (Definition 3.20) is $O(2^n)$. The cardinality of each set of formulas of $\text{At}(\mathcal{L}_{KB})$ is $O(n)$. The construction of the canonical model in Definition 3.29 requires the addition of relations $<$ and roles R among elements of $\text{At}(\mathcal{L}_{KB})$: this operation requires, at most, a number of steps which is exponential in n . Indeed, the construction introduces at most $R(A, B)$ for each R (at most n) and $A < B$, for each pair of elements $A \in \text{At}(\mathcal{L}_{KB})$ and $B \in \text{At}(\mathcal{L}_{KB})$, therefore $O(n) \times O(2^n) \times O(2^n)$ operations are performed, that is to say the construction requires an exponential number of steps. \square

4 Reasoning about Typicality

Logic $\mathcal{ALC} + \mathbf{T}$ allows one to reason monotonically about typicality. In $\mathcal{ALC} + \mathbf{T}$ we can consistently express, for instance, the fact that three different concepts, as *student*, *working student* and *working student with children*, have a different status as taxpayers.

What about the typical properties of an individual *john* that we know being a working student, and having children? Of course, if we know that *john* is a typical instance of the concept $\text{Student} \sqcap \text{Worker} \sqcap \exists \text{HasChild}.\top$, i.e. if the ABox contains the assertion $(*) \mathbf{T}(\text{Student} \sqcap \text{Working} \sqcap \exists \text{HasChild}.\top)(\text{john})$, then, in $\mathcal{ALC} + \mathbf{T}$, we can conclude that $\neg \text{TaxPayer}(\text{john})$. However, in absence of $(*)$, we cannot derive $\neg \text{TaxPayer}(\text{john})$.

In general, we would like to infer that individuals have the properties which are typical of the most specific concept to which they belong. To this purpose, we define a completion of the knowledge base which adds to the ABox, for each individual a occurring in the ABox, the assertion that a is a typical instance of the most specific

concept C to which it belongs. Although in general ABoxes can contain typicality assertions about individuals, in practice we assume that typicality assertions are automatically generated by the system by means of the completion, and are not inserted by the user. From now on, we therefore assume that the initial ABox of a KB does not contain any typicality assertion.

Definition 4.1 [Completion of a Knowledge Base] The KB (TBox, ABox') is the completion of the KB (TBox, ABox), if ABox' is obtained from ABox by adding to it, for all individual names a in the ABox, the assertion $\mathbf{T}(C_1 \sqcap \dots \sqcap C_j)(a)$, where C_1, \dots, C_j are all the concepts C_i such that: (1) C_i is a subconcept of any concept occurring in (TBox, ABox); (2) C_i does not contain \mathbf{T} ; (3) a is an instance of C_i , i.e. $C_i(a)$ is derivable in \mathcal{ALC} from (TBox, ABox).

For instance, assuming that $Student(john)$, $Worker(john)$ and $\exists HasChild.\top(john)$ are the only assertions concerning $john$ derivable from the KB, the completion above would add $\mathbf{T}(Student \sqcap Worker \sqcap \exists HasChild.\top)(john)$ to the ABox, as $Student \sqcap Worker \sqcap \exists HasChild.\top$ is the most specific concept of which $john$ is an instance. From this, we can conclude in $\mathcal{ALC} + \mathbf{T}$ that $john$ does not pay taxes.

The completion adds $\mathbf{T}(C_1 \sqcap \dots \sqcap C_j)(a)$ by considering each $C_i(a)$ derivable in \mathcal{ALC} from the KB, rather than considering only $C_i(a)$ in the ABox. This is needed, for instance, to infer that $john$ does not pay taxes from the KB containing $Professor \sqsubseteq \forall HasChild.Student$, $Professor(paul)$, and $HasChild(paul, john)$.

As a matter of fact, if we had in the ABox the information that $john$ is a *TaxPayer*, this would not cause an inconsistent completion of the KB. Indeed, in such a case, $Student \sqcap Worker \sqcap \exists HasChild.\top \sqcap TaxPayer$ would be the most specific concept of which $john$ is an instance, so that the assertion $\mathbf{T}(Student \sqcap Worker \sqcap \exists HasChild.\top \sqcap TaxPayer)(john)$ would be added in the completion of the KB. This does not allow to infer that $\neg TaxPayer(john)$. Hence, no inconsistency arises. However, it could be the case that the KB obtained by the completion is inconsistent, even if the initial KB is consistent. For instance, the KB containing $\mathbf{T}(C) \sqsubseteq \forall R.E$, $\mathbf{T}(D) \sqsubseteq \forall R.\neg E$, $C(a)$, $D(b)$, $R(a, c)$, and $R(b, c)$ is consistent, whereas its completion, including also $\mathbf{T}(C)(a)$ and $\mathbf{T}(D)(b)$, is not. In this case, we keep the initial KB unaltered, instead of the one obtained by the completion.

Notice that the completion of the ABox only introduces $O(n)$ new formulas $a : \mathbf{T}(C_1 \sqcap \dots \sqcap C_j)$, one for each named individual a in the ABox. Furthermore, the size of each formula $\mathbf{T}(C_1 \sqcap \dots \sqcap C_j)$ is $O(n^2)$ as C_1, \dots, C_j are all distinct subformulas of the initial formula ($O(n)$), and each C_i has size $O(n)$. Hence, after the completion construction, the size of the KB is polynomial in n . Moreover, for each individual a ($O(n)$) and for each concept C ($O(n)$), we have to check whether $C(a)$ is derivable in \mathcal{ALC} from the KB, which is a problem in EXPTIME. Hence, the completion construction requires exponential time and produces a KB of size polynomial in the size of the original one:

Theorem 4.2 *The problem of deciding satisfiability of the knowledge base after completion is in NEXPTIME in the size of the original KB.*

As mentioned, given a consistent KB, its completion could be inconsistent. In this case, we choose to keep the original KB. As an alternative, we could consider

all maximal consistent KBs (extensions) that can be generated by adding, for all individuals, the relative most-specific concept assumptions. We could then perform either a skeptical or a credulous reasoning with respect to such extensions.

Preferential logic allows to deal with some forms of inheritance among concepts, by the property of cautious monotonicity (which comes from the semantic property ($f_{\mathbf{T}} - 3$): if $\mathbf{T}(C) \sqsubseteq D$ and $\mathbf{T}(C) \sqsubseteq E$, then $\mathbf{T}(C \sqcap D) \sqsubseteq E$. Coming back to the example above, if we knew that all students typically have a teacher, i.e. $\mathbf{T}(Student) \sqsubseteq \exists HasTeacher.\top$, and that *john* is a student and has a teacher ($Student(john)$ and $\exists HasTeacher.\top(john)$ are in the ABox) then, by the completion construction above, we would get $\mathbf{T}(Student \sqcap \exists HasTeacher.\top)(john)$, and, by cautious monotonicity, we would conclude that *john* does not pay taxes.

5 Extensions

We consider this work as a first step. We plan to extend our approach in the following directions.

5.1 Inheritance with exceptions

Once the completion of a KB has been defined as above, the problem of inferring the typical properties of an individual is reduced to the problem of inferring the properties of the most specific concept to which it belongs. This can be done by reasoning on the typical properties of concepts in the TBox. Although Preferential Description Logic allows to capture - through cautious monotonicity - some form of inheritance of typical properties among concepts, there are cases in which cautious monotonicity is not strong enough to derive the intended conclusions. For instance, if we know that *jack* is a student who is a sport lover, we cannot conclude that *jack* is not a tax payer, as we do not have the property that typical students (or all students) are sport lovers, and hence cautious monotonicity is not applicable. Here we are faced with the problem of *Irrelevance*. Since the property of being a sport lover is irrelevant with respect to the property of paying taxes, we would like to infer that also $\mathbf{T}(Student \sqcap SportLover) \sqsubseteq \neg TaxPayer$, and therefore that *jack* is not a tax payer. In order to allow this form of inheritance among concepts, we can introduce a default rule of the following type:

$$\frac{\mathbf{T}(Student) \sqsubseteq \neg TaxPayer \quad : \mathbf{T}(Student \sqcap SportLover) \not\sqsubseteq TaxPayer}{\mathbf{T}(Student \sqcap SportLover) \sqsubseteq \neg TaxPayer} (IRR)$$

By the default rule above, if typical students are not tax payers, and *it is consistent* to assume that typical students who are sport lovers are not tax payers, then we could conclude that typical sport lover students are not tax payers. With this rule, the typical properties of a more general concept C are considered one by one, and are inherited by a more specific concept ($C \sqcap D$) if it is consistent to do so. In order to deal with default rules like this one, we need to integrate our calculus with a standard mechanism to reason about defaults.

5.2 Reasoning on the typicality of all instances

The completion of a knowledge base, as defined above, only applies to individuals explicitly named in the ABox. However, we would like to reason on the typical properties of all individuals. Assume, for instance, that the ABox contains the assertions: $\exists HasChild.Worker(bill)$ and $\forall HasChild.Student(bill)$. Thus, *bill* has a child who is a student and is working. We want to be able to infer that *bill* has a child who is a tax payer. To this purpose, we need to assume that the *bill*'s child is a typical working student.

To reason about the typicality of all individuals, we would need to assume that all individuals generated during the tableau construction are *typical* instances of the most specific concept to which they belong. To this purpose, we could think of applying the completion construction of Definition 4.1 to all generated individuals. The completion construction would be applied only when all relevant formulas $y : C_1, \dots, y : C_j$ with label y have already been introduced in the branch. According to the strategy described in section 3, the completion should be performed only after the application of the rules to all x s.t. $x \prec y$.

5.3 Extension to other DLs

We want to study the extension of our approach to more expressive description logics. For instance, we plan to extend $\mathcal{ALCN}\mathcal{R}$ considered in [4] with our typicality operator \mathbf{T} , and consider which is the complexity corresponding to this extension. Finally, we want to study the extension of the language of concepts by allowing arbitrary occurrences of the operator \mathbf{T} .

6 Conclusions

We have proposed an extension of \mathcal{ALC} for reasoning about typicality in Description Logic framework. The resulting logic is called $\mathcal{ALC}+\mathbf{T}$. We have proposed a calculus for deciding the satisfiability of a general knowledge base in $\mathcal{ALC}+\mathbf{T}$. The calculus, called $T^{\mathcal{ALC}+\mathbf{T}}$, is analytic, terminating, and allows us to decide the satisfiability of a knowledge base in $\mathcal{ALC}+\mathbf{T}$ in nondeterministic exponential time. The calculus is reminiscent of the tableaux calculi for KLM logics presented in [9,8]. We have then shown how to complete the ABox by means of typicality assumptions, in order to infer prototypical properties of the individuals explicitly mentioned in the ABox. We have argued how to apply a similar completion also to individuals implicitly mentioned in the ABox, in order to infer their properties. Finally, we have sketched how to reason about the inheritance of typical properties from more general to more specific concepts handling with irrelevant information, by using appropriate default rules.

KLM logics are related to probabilistic reasoning. A probabilistic extension of DLs has been proposed in [11]. In particular, the notion of conditional constraint in [11] allows typicality assertions to be expressed (with a specified probability). We plan to compare in details this probabilistic approach to ours elsewhere.

References

- [1] Baader, F. and B. Hollunder, *Embedding defaults into terminological knowledge representation formalisms*, Journal of Automated Reasoning **14** (1995), pp. 149–180.
- [2] Baader, F. and B. Hollunder, *Priorities on defaults with prerequisites, and their application in treating specificity in terminological default logic*, Journal of Automated Reasoning **15** (1995), pp. 41–68.
- [3] Bonatti, P. A., C. Lutz and F. Wolter, *Description logics with circumscription.*, in: *Proceedings of KR*, 2006, pp. 400–410.
- [4] Buchheit, M., F. M. Donini and A. Schaerf, *Decidable reasoning in terminological knowledge representation systems*, Journal of Artificial Intelligence Research (JAIR) **1** (1993), pp. 109–138.
- [5] Donini, F. M., M. Lenzerini, D. Nardi, W. Nutt and A. Schaerf, *An epistemic operator for description logics*, Artificial Intelligence **100** (1998), pp. 225–274.
- [6] Donini, F. M., D. Nardi and R. Rosati, *Description logics of minimal knowledge and negation as failure*, ACM Transactions on Computational Logic (ToCL) **3** (2002), pp. 177–225.
- [7] Eiter, T., T. Lukasiewicz, R. Schindlauer and H. Tompits, *Combining answer set programming with description logics for the semantic web*, in: *Proceedings of KR*, 141–151, 2004.
- [8] Giordano, L., V. Gliozzi, N. Olivetti and G. L. Pozzato, *Analytic Tableaux Calculi for KLM Rational Logic R*, in: M. Fisher, W. van der Hoek, B. Konev and A. Lisitsa, editors, *Proceedings of JELIA 2006*, LNAI **4160**, pp. 190–202.
- [9] Giordano, L., V. Gliozzi, N. Olivetti and G. L. Pozzato, *Analytic Tableaux for KLM Preferential and Cumulative Logics*, in: G. Sutcliffe and A. Voronkov, editors, *Proceedings of LPAR 2005*, LNAI **3835**, pp. 666–681.
- [10] Giordano, L., V. Gliozzi, N. Olivetti and G. L. Pozzato, *Preferential Description Logics*, in: N. Dershowitz and A. Voronkov, editors, *Proceedings of LPAR 2007*, LNAI **4790**, pp. 257–272.
- [11] Giugno, R. and T. Lukasiewicz, *P-SHOQ(D): A Probabilistic Extension of SHOQ(D) for Probabilistic Ontologies in the Semantic Web*, in: S. Flesca, S. Greco, N. Leone and G. Ianni, editors, *Proceedings of JELIA 2002*, LNAI **2424**, pp. 86–97.
- [12] Kraus, S., D. Lehmann and M. Magidor, *Nonmonotonic reasoning, preferential models and cumulative logics*, Artificial Intelligence **44** (1990), pp. 167–207.
- [13] Straccia, U., *Default inheritance reasoning in hybrid kl-one-style logics*, in: *Proceedings of IJCAI*, 1993, pp. 676–681.