In *Associazione italiana per l'intelligenza artificiale*, 4(4):22–36, 1991.

A Terminological Default Logic

Umberto Straccia
Istituto di Elaborazione dell'Informazione
Consiglio Nazionale delle Ricerche
Via S. Maria, 46 - 56126 Pisa (Italy)

Abstract

Terminological Logics are knowledge representation formalisms of enormous applicative interest, as they are specifically oriented to the vast class of application domains that are describable by means of taxonomic organizations of complex objects. Although the field of terminological logics has lately been an active area of investigation, few researchers (if any) have addressed the problem of extending these logics with the ability to perform *default reasoning*, an important form of non-monotonic reasoning. Such extensions would prove of paramount applicative value, as for many application domains a monotonic formalization may be accomplished only at the price of oversimplification. In this paper we show how we can effectively integrate terminological reasoning and default reasoning, yielding a *terminological default logic*. The kind of default reasoning we embed in our terminological logic is based on Reiter's Default Logic, but overcomes some of its drawbacks by subscribing to the implicit handling of exceptions typical of Touretzky's Multiple Inheritance Hierarchies with Exceptions.

1 Introduction

Terminological Logics (TLs, also known as *Frame Representation Languages* or *Concept Description Languages*) are knowledge representation formalisms of enormous applicative interest, as they are specifically oriented to the vast class of application domains that are describable by means of taxonomic organizations of complex objects. Such logics have a non formalized counterpart in a large class of early knowledge

representation languages based on semantic networks and inspired by the notion of *frame*, originally introduced by Minsky [12]. Unfortunately these languages, of which KL-ONE [1] may be considered the founding father, did not possess a formal semantics; hence, the fact that the notion of inference they enforced were the same as the user expected relied on an improbable like-mindedness among designer, implementor, user and critic. TLs may then be seen as the result of a work of distillation and formalization that was carried out on KL-ONE-like languages, a work that resulted in linguistic constructs of dubious semantic content being purged while the others were brought back within the canons of mathematical logic.

TLs are especially popular among knowledge engineers because of the vastity of their possible applications (in fact, many are the application domains whose formalization may be accomplished, at least partially, by means of a taxonomic structure), and because of their easy and intuitive syntax. This syntax might be dubbed *object-oriented*, as it encourages the structuring of a Knowledge Base (KB) into "clusters of knowledge" that represent structured descriptions of classes of individuals belonging to the domain of discourse.

Unlike better known logics (such as e.g. First Order Logic - FOL), whose main syntactic constituents are formulae, TLs have *terms* as their primary syntactic expressions. A term is usually an expression that denotes monadic or dyadic relations on the domain of discourse. In general, the languages of TLs allow for a number of term-forming operators by means of which one may build complex terms starting from a basic repertory of simple terms (*viz.* predicate symbols).

The field of terminological logics has lately been an active area of research, with the attention of researchers especially focusing on the investigation of their logical and computational properties: results have been established concerning either the tractability [3, 25, 4], or the intractability [8, 16, 14, 23], or the undecidability [17, 21, 22] of a number of TLs.

Nevertheless, few researchers (if any) have addressed the problem of extending these logics with the ability to perform *default reasoning*, a kind of reasoning that is to be applied whenever the rules involved allow for *exceptions*. Such rules dictate that, "by default" (i.e. whenever there is no information to the contrary), one may assume that the situation that is to be dealt with is a *typical* (i.e. not exceptional) situation. However, a possible future acquisition of new knowledge might lead one to discover that this situation is not typical as was believed, but *exceptional*; in this case, what had been assumed

"by default" is *retracted*. Hence, reasoning by default (also known as *reasoning in the presence of exceptions*) is a non-monotonic kind of reasoning[1].

An instance of this reasoning style is the inferential chain induced by the assertion "if x is a bird, assume that it flies, unless you know it does not". If we reason according to this rule and know that Opus is a bird, in the absence of information to the contrary we may conclude that Opus flies; but if we later learn that Opus is a penguin, knowing that penguins typically do not fly we have to revoke the conclusion that Opus flies.

Non-monotonic reasoning is nowadays a hot topic in Artificial Intelligence (AI); in fact, the spectrum of its applications is extremely wide, ranging from image interpretation [10], reasoning in the presence of temporal knowledge [6], troubleshooting [19] and KBs automatic generation [24]. The requirements that are imposed by these application domains have lead to the development of a variety of non-monotonic formalisms, the best known of which are perhaps Doyle and McDermott's Nonmonotonic Logic [5], Moore's Autoepistemic logic [13], Reiter's Default Logic [18] and McCarthy's Circumscription [11].

Each of these logics may be seen as extending FOL with non-monotonic reasoning capabilities of some kind. Given that TLs may be viewed as subsets of FOL, one might be led to think that a simple integration of default and terminological reasoning could be obtained simply by considering one of the non-monotonic formalisms above and restricting it to deal not with the chosen terminological logic, rather than FOL *tout court*.

Unfortunately, these non-monotonic formalisms, besides having unattractive metalogical and computational properties (i.e. undecidability of the versions with quantifiers, computational intractability of those without quantifiers), suffer from a problem that hinders their use in knowledge representation contexts requiring that KB construction be accomplished in an incremental fashion. We will call this problem the *Exceptions Explicitation Problem* (EEP).

Incrementality of KB construction is an asset of KB management systems that hardly needs to be argued for. Large KBs are the result of an evolutionary process, both because knowledge entry is a time-consuming process and because knowledge may simply become available at later stages of the process. Also, when a large KB is built

[1] A note for the uninitiated reader. Given a provability relation, $\vdash_{\mathcal{L}}$, over a formal language $\mathcal{L}$, then $\vdash_{\mathcal{L}}$ is *monotonic* iff for any set $S_{\mathcal{L}}$ and $S'_{\mathcal{L}}$ of expressions over $\mathcal{L}$, it is true that

$$S_{\mathcal{L}} \subseteq S'_{\mathcal{L}} \quad \text{implies} \quad \{A : S_{\mathcal{L}} \vdash_{\mathcal{L}} A\} \subseteq \{A : S'_{\mathcal{L}} \vdash_{\mathcal{L}} A\}$$

otherwise $\vdash_{\mathcal{L}}$ is *non-monotonic*.

by this "stepwise refinement" process, it is highly desirable that the refinement consists in the plain, piecemeal *addition* of new knowledge chunks rather than in a time-consuming *revision* (with possibly ensuing deletion) of pre-existing chunks.

1.1 The Exceptions Explicitation Problem

We will now discuss, by means of a concrete example, what the EEP is and how it manifests itself in the context of Nonmonotonic Logic (NML); to this respect, other formalisms such as Default Logic and Circumscription behave in a completely analogous way, and will not be discussed. The "penguin" example that we have seen above may be represented by means of the NML formula

$$\forall x\ Bird(x) \wedge M[Flies(x)] \Rightarrow Flies(x) \quad (1)$$

(where $M[\alpha]$ informally means "α does not contradict the KB"). Let us consider an exception to the rule, "penguins are birds that typically do not fly", and let us represent it by means of the axioms:

$$\forall x\ Penguin(x) \Rightarrow Bird(x) \quad (2)$$
$$\forall x\ Penguin(x) \wedge M[\neg Flies(x)] \Rightarrow \neg Flies(x) \quad (3)$$

There is a problem hidden in this set of formulae: in NML the addition of the assertion $Penguin(opus)$ to axioms (1)-(3) generates two different "sets of conclusions" ("fixpoints", in NML terminology): in one of them Opus flies while in the other Opus does not fly, depending on whether we "first" consider axiom (1) or axiom (3), respectively; this happens notwithstanding the fact that our knowledge of the animal world makes us strongly prefer the conclusion according to which Opus does not fly.

Hence NML seems, at first sight, inadequate to account for this kind of reasoning, as it generates a situation of ambiguity in the representation of a state of affairs that is all but ambiguous to us[2]. At first sight, it seems possible to overcome this problem without departing from NML; what one needs to do is *explicitly* consider exceptions in each default formula they involve. Accordingly, a set of axioms that induces a correct behaviour of NML for our example is the following:

$$\forall x\ Bird(x) \wedge M[Flies(x) \wedge \neg Penguin(x)] \Rightarrow Flies$$
$$\forall x\ Penguin(x) \Rightarrow Bird(x)$$
$$\forall x\ Penguin(x) \wedge M[\neg Flies(x)] \Rightarrow \neg Flies(x)$$

At this point, given $Penguin(opus)$, axiom (4) does not allow one to con-

[2] The generation of multiple fixpoints in NML may also be due to an *inherent* ambiguity of the state of affairs being represented. A very famous example of this is the so-called "Nixon Diamond", which was first described in [18].

clude $Flies(opus)$; this leaves us with one fixpoint only, a fixpoint that contains only the conclusions we actually subscribe to.

However, this solution looks more like an *ad hoc* patch rather than a real solution. In fact, it should be noted that, given the informal meaning that Doyle and McDermott have attributed to NML formulae, the conclusion according to which Opus does not fly should already follow from axioms (1)-(3). In fact, they already represent the fact that penguins are exceptional birds, as far as the ability to fly is involved; consequently, the modification of axiom (1) to yield (4) seems a redundant reassertion of this fact.

Besides one should note that, until the KB did not contain any reference to penguins, axiom (1) was more than adequate. The introduction of axioms (2) and (3) has obliged us to transform axiom (1) into axiom (4); in the more general case, one has to *retract* (in order to subsequently assert some modified version of them) *formulae from the KB*. It is then evident that the realization of default reasoning by means of NML is undermined by a problem of a semantic nature: the impossibility, unlike what happens with traditional logical formalisms, to characterize the extensional semantics of the modified KB in a *compositional* way, i.e. as a combination of the extensions of the original KB and of the formula introduced therein.

But expliciting exceptions brings about other, even more important problems that have to do with the way in which, at the time of the introduction of a new formula into the KB, it is determined which are the formulae that are to be modified and which are the ones that are not. In fact, this operation cannot be realized simply by performing a number of matchings between the formula to be introduced and the other formulae in the KB, but *requires in general an NML theorem proving operation.* This may clearly be seen by taking a look at our example: the relation between the precondition of axiom (1) and the precondition of axiom (3) has been determined by relying on the fact that $Bird(x)$ is *derivable* from $Penguin(x)$ by means of axiom (2); the transformation of axiom (1) into (4) is then realized in order to "inhibit" not axiom (1) in itself, but the inferential chain that would lead us to conclude $Flies(x)$ from $Penguin(x)$. In general, given an NML knowldge base, it is not even determined *a priori* whether there exist one, several or no inferential chains that lead to the undesired conclusion (in our case, one, several or no inferential chains between $Penguin(x)$ and $Flies(x)$); in case several such chains exist, *each of them* has to be inhibited by means of a call to the NML theorem prover and by the subsequent introduction of a formula of type

$M[\alpha]$ into the KB.

That the phase of KB construction requires repeated calls to the NML theorem prover in order to maintain the consistency of the KB that is being built, is in itself rather implausible; but the whole endeavour becomes absurd once we consider that NML is undecidable! Unless the construction of the KB is realized in a completely *static* (non incremental) way, the problem of KB construction in NML is an *unsolvable* problem.

1.2 Implicit handling of exceptions

The Exceptions Explicitation Problem that affects standard non-monotonic formalisms has lead to the definition of languages for the representation of *Multiple Inheritance Networks with Exceptions* (MINEs) [27]. Such languages have two fundamental characteristics:

1. they are specifically oriented to the representation of non-monotonic knowledge of a taxonomic nature;

2. they implement an *implicit* handling of exceptions.

Item 1 means that MINEs are less expressive of the more general non-monotonic formalisms we have mentioned in Section 1, in the sense that they only allow for monadic predicate symbols, a limited use of negation and no disjunction at all; besides, their monotonic fragment is also less expressive than TLs as, having no term constructors in their syntactic apparatus, they only allow the construction of taxonomies of *simple, unstructured objects*.

As for Item 2, MINEs *implicitly* handle exceptions by exploiting the partial ordering given by the taxonomy: as a first approximation we can say that, in case of conflicts, a "default rule" $a \rightarrow b$ is preferred to another default rule $c \not\rightarrow b$ if the precondition of the first precedes the precondition of the second in the ordering. For example, given the MINE consisting of the default rules $opus \rightarrow Penguin$, $Penguin \rightarrow Bird$, $Bird \rightarrow Flies$, $Penguin \not\rightarrow Flies$, the conclusion according to which Opus does not fly is given priority on the grounds that the premise of $Penguin \not\rightarrow Flies$ is more specific than that of $Bird \rightarrow Flies$, and that, as a consequence, the conclusion which is derivable from the former is more reliable than the one which is derivable from the latter. In other words, the implicit handling of exceptions obeys the so-called *specialization principle*, according to which "conflicts" are to be solved by preferring the properties belonging to a subclass over those belonging to a superclass.

In this paper we will show how we can extend TLs in such a way that they allow a brand of default reasoning that

obeys the specialization principle, thus creating a formalism that combines the tools for describing taxonomic organizations of complex objects which are typical of TLs, with the ability to describe default information which is typical of MINEs, but at the same time retaining incrementality in KB construction. We will call our logic $\mathcal{TDL}$ (Terminological Default Logic).

This paper is organized as follows. In Section 2 we formally introduce the syntax and the semantics of $\mathcal{TDL}$, starting from its monotonic fragment and subsequently proceeding to the non-monotonic features, defining the notion of "extension" of a set of $\mathcal{TDL}$ formulae (i.e. the set of conclusions that may be derived from these formulae). Such a definition is given in the style of Reiter's Default Logic, i.e. as a fixpoint of a consequence relation. Hence, our logic inherits some of its metalogical properties from Default Logic itself; however, an important result we discuss in this paper is that the set of extensions of a set of $\mathcal{TDL}$ formulae is recursively enumerable, while it is not such in Default Logic. In Section 3 we present an algorithm that computes an extension of a set of $\mathcal{TDL}$ formulae (when it exists). In Section 4 we discuss some issues related to the computational complexity of reasoning in $\mathcal{TDL}$. Section 5 concludes.

2 Default reasoning in terminological theories

2.1 The monotonic fragment of $\mathcal{TDL}$

The $\mathcal{TDL}$ logic, like many other TLs, allows the specification of three fundamental types of terms: *frames*, *slots* and *individual constants*. Frames are terms denoting sets of individuals, and are, so to speak, the first-class citizens of $\mathcal{TDL}$ (i.e. they have the same role that e.g. formulae have in FOL). Slots are terms denoting binary relations between individuals; their function is to allow the specification of structural constituents of frames. Individual constants denote individuals of the domain of discourse. For example, the basic repertory of simple terms (called *atoms*) that are used in order to build more complex terms might contain the frame `Polyhedron`, denoting the set of polyhedra, and the slot `side`, denoting all those pairs of individuals $\langle x, y\rangle$ such that y is one of the sides of x; this would allow us to define more complex frames, such as e.g. the term $\forall$`Side.Polygons` denoting the set of those individuals whose sides are all polygons (i.e. the set of polyhedra), and to subsequently define other frames by using those defined before.

In order to introduce the syntax of $\mathcal{TDL}$ we will need three disjoint al-

phabets of symbols: an alphabet I of individual constants (with metavariables $i, i_1, i_2, \ldots$), an alphabet MP of monadic predicate symbols (with metavariables $M, M_1, M_2, \ldots$) and an alphabet DP of dyadic predicate symbols (with metavariables $D, D_1, D_2, \ldots$).

The syntax of $\mathcal{TDL}$ is specified by the following definition.

Definition 2.1 *A* frame *in $\mathcal{TDL}$ is defined by the following syntax:*

$$\begin{array}{rcl} F_1, F_2 & \rightarrow & F_1 \sqcap F_2 \mid M \mid \neg M \mid \forall S.F_1 \mid \{i, \ldots, i\} \mid \bot \mid \top \\ S & \rightarrow & D \end{array}$$ ∎

We will use metavariables $F, F_1, F_2, \ldots$ ranging on frames and metavaraibles $S, S_1, S_2, \ldots$ ranging on slots.

Let us now switch to the semantics of the language of $\mathcal{TDL}$. The meaning of the linguistic constructs introduced above may be given in terms of the notion of extension function.

Definition 2.2 *An* interpretation $\mathcal{I}$ *over a nonempty set of individuals $\mathcal{D}$ is a function that maps elements of MP into subsets of $\mathcal{D}$, elements of DP into subsets of $\mathcal{D} \times \mathcal{D}$ and elements of I into elements of $\mathcal{D}$ such that $\mathcal{I}(i_1) \neq \mathcal{I}(i_2)$ if $i_1 \neq i_2$*[3]*. We will say that $\mathcal{I}$ is an* extension function *iff it satisfies the following equations:*

$$\begin{array}{rcl} \mathcal{I}(\bot) & = & \emptyset \\ \mathcal{I}(\top) & = & \mathcal{D} \\ \mathcal{I}(\neg M) & = & \mathcal{D} \setminus \mathcal{I}(M) \\ \mathcal{I}(\{i_1, \ldots, i_n\}) & = & \{x \in \mathcal{D} \mid x = \mathcal{I}(i_1) \vee \ldots \vee x = \mathcal{I}(i_n)\} \\ \mathcal{I}(F_1 \sqcap F_2) & = & \mathcal{I}(F_1) \cap \mathcal{I}(F_2) \\ \mathcal{I}(\forall S.F) & = & \{x \in \mathcal{D} \mid \forall y : \langle x, y\rangle \in \mathcal{I}(S) \Rightarrow y \in \mathcal{I}(F)\} \end{array}$$

Moreover, we can associate names to complex frames, with the net effect that we will subsequently be able to define new frames using names instead of the corresponding complex frames.

Definition 2.3 *A* name introduction *is an expression of the form $M \doteq F$ or of the form $M \lessdot F$, where F is a frame and M an element of MP. We say that M is the* frame name *of F.* ∎

Name introductions of type $M \doteq F$ define necessary and sufficient conditions for an individual to belong to the extension of M, whereas the conditions defined by name introductions of type $M \lessdot F$ are only necessary. This is formalized by the following definition.

Definition 2.4 *An extension function $\mathcal{I}$ over a nonempty domain $\mathcal{D}$* satisfies a name introduction δ *iff*

$$\begin{array}{rcll} \mathcal{I}(M) & = & \mathcal{I}(F) & \text{if} \quad \delta = M \doteq F \\ \mathcal{I}(M) & \subseteq & \mathcal{I}(F) & \text{if} \quad \delta = M \lessdot F \end{array}$$ ∎

[3] This restriction on the interpretation function ensures that different individual constants denote different individuals. This property is usually called *unique names assumption.*

For example, the following expressions are name introductions.

$$\begin{array}{rcl} \texttt{Regular\_triangle} & \doteq & \texttt{Triangle} \sqcap \texttt{Regular\_polygon} \\ \texttt{Regular\_tetrahedron} & \doteq & \texttt{Polyhedron} \sqcap \forall \texttt{Side.Regular\_triangle} \\ \texttt{Man} & \lessdot & \texttt{Mammal} \sqcap \texttt{Male} \end{array}$$

Our language allows also assertions, stating that individual constants are instances of frames and pairs of individual constants are instances of slots.

Definition 2.5 *An* assertion *is an expression of the form $i_1{:}F$ or $\langle i_1, i_2\rangle : S$ where i_1, i_2 are individual constants, F is a frame and S is a slot.* ∎

Definition 2.6 *An extension function $\mathcal{I}$ over a nonempty domain $\mathcal{D}$* satisfies an assertion α *iff*

$$\begin{array}{rcll} \mathcal{I}(i_1) & \in & \mathcal{I}(F) & \text{if} \quad \alpha = i_1 : F \\ \langle \mathcal{I}(i_1), \mathcal{I}(i_2)\rangle & \in & \mathcal{I}(S) & \text{if} \quad \alpha = \langle i_1, i_2\rangle : S \end{array}$$

∎

Definition 2.7 *Let Ω be a set of name introductions and assertions, and $\mathcal{I}$ an extension function over a nonempty domain $\mathcal{D}$. $\mathcal{I}$ is a* model *of Ω iff $\mathcal{I}$ satisfies all name introductions and assertions in Ω.* ∎

Definition 2.8 *Let Ω be a set of name introductions and assertions. Ω is* satisfiable *iff there exists a model of Ω. Otherwise we say that Ω is* unsatisfiable. ∎

introductions and assertions, and F a frame. empty in Ω iff for every model $\mathcal{I}$ of Ω is true that $\mathcal{I}(F) = \emptyset$. ∎

Definition 2.9 *Let Ω be a satisfiable set of name introductions and assertions, and M_1, M_2 be two elements of MP*

1. *M_1 is* subsumed *by M_2 in Ω (written $M_1 \preceq_\Omega M_2$) iff for every model $\mathcal{I}$ of Ω it is true that $\mathcal{I}(M_1) \subseteq \mathcal{I}(M_2)$;*

2. Ω implies logically *an assertion α (written $\Omega \models \alpha$) iff α is satisfied by all models of Ω.* ∎

name introductions and assertions, and F a frame. empty in Ω iff $\mathcal{I}(F)$ is empty in every model $\mathcal{I}$
The following definition formalizes what name introductions and assertions hold, given a set of name introductions and assertions Ω. ∎

Definition 2.10 *Let Ω be a satisfiable set of name introductions and assertions. The* transistive closure *of Ω (written $TC(\Omega)$), is given by the following set:*

$$TC(\Omega) \;=\; \Omega \;\cup \{\alpha \mid \Omega \models \alpha\}$$ ∎

Notice that the notion of logical implication is defined for assertions only: this means that a set Ω differs from its closure $TC(\Omega)$ only in the assertions it contains.

It is easy to show that the following propositions hold.

Proposition 2.1 *If Ω and Ω' are satisfiable sets of name introductions and assertions with $\Omega \subseteq \Omega'$, then $TC(\Omega) \subseteq TC(\Omega')$.* ∎

Proposition 2.2 *If Ω is a set of name introductions and assertions, then $TC(\Omega) = TC(TC(\Omega))$.* ∎

2.2 The non-monotonic part of $\mathcal{TDL}$

Up to now we have described the monotonic fragment of $\mathcal{TDL}$. Let us now describe the addition of non-monotonic features.

Definition 2.11 *A* default *is an expression of the form $M \mapsto \forall S.F$, where M is an element of MP.* ∎

Informally, $M \mapsto \forall S.F$ means that "if i is an M and the assumption that all the S's of i are F does not lead to a contradiction, assume it". For example, the default `Italian_University` $\mapsto$ $\forall$`Faculty_Member.Italian` means that "if i is an `Italian_University` and the assumption that all its `Faculty_Member`s are `Italian`s does not lead to a contradiction (i.e. we do not know of any particular `Faculty_Member` of i who is not an `Italian`), assume it".

The particular form we have chosen for defaults is due to two main facts:

1. an analysis of the literature concerning the interaction between frames and default knowledge, ranging from the more informal and impressionistic proposals (such as those of e.g. [12, 20]) to the more formally justified ones [2], reveals that universally quantified default rules with consequents in a "slot-filler" form have been identified, since the very inception of the field, as the most natural way in which to convey default frame-like knowledge;

2. universally quantified rules are sufficient to highlight the main problems resulting from the interaction between default knowledge and terminological knowledge; on the other hand, the extension to other forms of defaults (such as e.g. those with existential quantification, or those involving numeric restrictions) is conceptually easy, but does not teach us much with respect to the issue of the integration between default and terminological reasoning.

Next we define what we mean by a $\mathcal{TDL}$ theory.

Definition 2.12 *A $\mathcal{TDL}$* theory *is a tuple $\mathcal{T} = <\Psi, \Delta>$, where Ψ is a satisfiable finite set of name introductions and assertions, and Δ is a finite set of defaults.* ∎

We are now able to define what the *extensions* of a $\mathcal{TDL}$ theory are. Informally, by the term "extension" we mean the set of name introductions and assertions that we can reasonably believe to be true in the presence of default assumptions. For instance, if we knew that, by default, the faculty members of Italian universities are Italians, that the University of Bellosguardo is an Italian university, and that Professor Dolcevita is a faculty member thereof, we would like to conclude (formally: to include in the corresponding extension) that Dolcevita is an Italian.

The definition of "extension" is similar to the one given by Reiter for Default Logic, i.e. an extension is a fixpoint of a consequence relation. However, unlike in Default Logic, the specialization principle is "wired" in our definition: in the presence of conflicting defaults, the one with the more specific premise will be preferred. For instance, suppose that, besides our knowledge on the faculty members of Italian universities being typically Italian themselves, we also knew that the faculty members of South Tyrolean universities are typically not Italian; knowing that South Tyrolean universities are Italian universities, that the University of Pflunders is a South Tyrolean university, and that Professor Katzenjammer is a faculty member thereof, we will be able to derive, as desired, that Katzenjammer is not Italian.

Definition 2.13 *Let $\mathcal{T} = <\Psi, \Delta>$ be a $\mathcal{TDL}$ theory. For any satisfiable set Ω of name introductions and assertions, let $\Gamma(\Omega)$ be the smallest satisfiable set of name introductions and assertions satisfying the following closure conditions:*

1. $\Gamma(\Omega) = TC(\Gamma(\Omega))$;
2. $\Psi \subseteq \Gamma(\Omega)$;
3. *if $M_1 \mapsto \forall S.F_1 \in \Delta$ and*
   (a) $i_1 : M_1 \in \Gamma(\Omega)$;
   (b) *for all M_2 such that $M_2 \preceq_\Omega M_1$, $i_1{:}M_2 \in \Omega$ and $M_2 \mapsto \forall S.F_2 \in \Delta$, it is true that $\Omega \cup \{i_1{:}\forall S.F_1 \sqcap F_2, \langle i_1, i_2\rangle : S\}$ is satisfiable for some individual constant i_2*

   then $i_1 : \forall S.F_1 \in \Gamma(\Omega)$.

A satisfiable set $\mathcal{E}$ of name introductions and assertions is an extension *of the $\mathcal{TDL}$ theory $\mathcal{T}$ iff $\mathcal{E} = \Gamma(\mathcal{E})$, i.e. iff $\mathcal{E}$ is a fixpoint of the operator Γ.* ∎

In this definition Condition 1 ensures that $\Gamma(\Omega)$ is closed under "$\models$", while Condition 2 ensures that $\Gamma(\Omega)$ complies with the name introductions and assertions of $\mathcal{T}$; these two conditions closely resemble those in Reiter's definition of "extension". Finally, Condition 3 embodies the specialization principle: a default $M_1 \mapsto \forall S.F_1$ is "applicable" iff (Condition 3a) the precondition $i_1{:}M_1$

holds and (Condition 3b) there is no evidence contrary to the conclusion of the default, i.e. i_1 does not belong to any subclass of M_1 which is the premise of a default "conflicting" with $\forall S.F_1$[4].

Notice that, according to our definition, a default like $M_1 \mapsto \forall S.F_1$, where F_1 is empty in Ω never satisfies Condition 3b[5], since $\Omega \cup \{i_1 : \forall S.F_1, \langle i_1, i_2\rangle : S\}$ is unsatisfiable for each individual constant i_2 (see footnote 4). If we wanted to take into account this form of defaults too, we should reformulate Condition 3b, distinguishing between the four possible cases in which F_1/F_2 are empty/nonempty. This reformulation would be relatively easy, but would lead to unnecessary complexities; see [26] for a detailed investigation of this case.

We go on to discuss some properties of the notion of extension as formalized in definition 2.13. The following propositions are parallel those given by Reiter for Default Logic, and are proven in [26].

Proposition 2.3 *Let $\mathcal{T} = <\Psi, \Delta>$ be a $\mathcal{TDL}$ theory. $\mathcal{E}$ is an extension of $\mathcal{T}$ iff $\mathcal{E} = \bigcup_{i=0}^{\infty} \mathcal{E}_i$, where*

$$\begin{array}{lcl} \mathcal{E}_0 & = & \Psi \\ \mathcal{E}_{i+1} & = & TC(\mathcal{E}_i) \cup C_i \qquad \textit{with} \end{array}$$

$$\begin{array}{lcl} C_i & = & \{i_1 : \forall S.F_1 \mid M_1 \mapsto \forall S.F_1 \in \Delta, \mathcal{E}_i \models i_1 : M_1, \\ & & \textit{for all } M_2 \textit{ such that } M_2 \preceq_{\mathcal{E}} M_1, \mathcal{E} \models i_1 : M \\ & & \textit{it is true that } \mathcal{E} \cup \{i_1 : \forall S.F_1 \sqcap F_2, \langle i_1, i_2\rangle : \\ & & \textit{for some individual constant } i_2\} \end{array}$$

∎

Using Proposition 2.3 it is easy to prove Proposition 2.4, which shows that extensions are "maximal" sets.

Proposition 2.4 *Let $\mathcal{T} = <\Psi, \Delta>$ be a $\mathcal{TDL}$ theory, and let $\mathcal{E}_1$ and $\mathcal{E}_2$ be extensions of $\mathcal{T}$. If $\mathcal{E}_1 \subseteq \mathcal{E}_2$, then $\mathcal{E}_1 = \mathcal{E}_2$.*

∎

Let us now consider some examples to show how Definition 2.13 works. The first example is the well know penguin example.

Example 2.1 Let $\mathcal{T}_1 = <\Psi_1, \Delta_1>$ be a $\mathcal{TDL}$ theory, with

$$\Psi_1 = \{\texttt{opus:Bird}, \texttt{Penguin} \prec \texttt{Bird} \sqcap \forall \texttt{Fly.}\{\texttt{no}\}\}$$
$$\Delta_1 = \{\texttt{Bird} \mapsto \forall \texttt{Fly.}\{\texttt{yes}\}\}$$

Let

$$\mathcal{E}_1 = TC(\Psi_1 \cup \{\texttt{opus}{:}\forall \texttt{Fly.}\{\texttt{yes}\}\}) \quad \text{and}$$
$$\Gamma(\mathcal{E}_1) = \mathcal{E}_1$$

We show that $\Gamma(\mathcal{E}_1)$ satisfies the conditions of Definition 2.13; therefore $\mathcal{E}_1$ is an extension of $\mathcal{T}$.

[4] Observe that condition 3b must obtain also in the case in which $M_1 \equiv M_2$ and $F_1 \equiv F_2$, hence ensures that $\Omega \cup \{i_1 : \forall S.F_1, \langle i_1, i_2\rangle : S\}$ must be satisfiable.

[5] F_1 is *empty* in Ω iff for every model $\mathcal{I}$ of Ω it is true that $\mathcal{I}(F_1) = \emptyset$.

First we observe that $\mathcal{E}_1$ is satisfiable, so $\Gamma(\mathcal{E}_1)$ is satisfiable too. From Proposition 2.2, Condition 1 holds. From the definition of $\mathcal{E}_1$ it follows that $\Psi_1 \subseteq \mathcal{E}_1$, so Condition 2 holds too. It is easy to see that Condition 3 holds for $\texttt{Bird}\mapsto \forall\texttt{Fly.}\{\texttt{yes}\} \in \Delta_1$ and $\texttt{opus:Bird} \in \Gamma(\mathcal{E}_1)$. So $\texttt{opus:}\forall\texttt{Fly.}\{\texttt{yes}\}$ must be in $\Gamma(\mathcal{E}_1)$ and $\Gamma(\mathcal{E}_1)$ cannot be smaller. Thus $\Gamma(\mathcal{E}_1)$ satisfies the conditions of Definition 2.13.

It is interesting to observe that for $\mathcal{E}' = TC(\Psi_1 \cup \{\texttt{opus:Penguin}\})$ we can choose $\Gamma(\mathcal{E}')$ to be equal to $TC(\Psi_1)$, so that $\Gamma(\mathcal{E}')$ satisfies Conditions 1, 2, 3[6]. So $\Gamma(\mathcal{E}') \neq \mathcal{E}'$ and, thus, $\mathcal{E}'$ is not an extension of $\mathcal{T}$. ∎

We will use this example to show that $\mathcal{TDL}$ is in fact non-monotonic.

Proposition 2.5
$\mathcal{TDL}$ is non-monotonic, i.e. there exists a $\mathcal{TDL}$ theory $\mathcal{T} = <\Psi, \Delta>$ with an extension $\mathcal{E}$ such that: there exists a set of name introductions and assertions Ψ' and a set of defaults Δ', such that none of the extensions $\mathcal{E}'$ of $\mathcal{T}' = <\Psi \cup \Psi', \Delta \cup \Delta'>$ is such that $\mathcal{E} \subseteq \mathcal{E}'$.

Proof: Let $\mathcal{T}_1$ be the $\mathcal{TDL}$ theory of Example 2.1 and $\mathcal{E}_1$ his (unique) extension. Let $\Psi' = \{\texttt{opus:Penguin}\}$ and $\Delta' = \emptyset$. It is easy to see that the unique extension of $\mathcal{T}' = <\Psi_1 \cup \Psi', \Delta_1 \cup \Delta'>$ is $\mathcal{E}' = TC(\Psi_1 \cup \Psi')$. Therefore $\mathcal{E}_1 \not\subseteq \mathcal{E}'$. ∎

In the next example we show how $\mathcal{TDL}$ employs an implicit handling of exceptions, and discuss how the same example may be formalized in Nonmonotonic Logic only at the price of a cumbersome operation of exceptions explicitation.

Example 2.2 Let $\mathcal{T}_2 = <\Psi_2, \Delta_2>$ be the $\mathcal{TDL}$ theory that formalizes out the "South Tyrolean Professors" example[7], with

$$\Psi_2 = \{\texttt{p:STU},\ \langle\texttt{p,k}\rangle\texttt{:FM},\ \texttt{STU} <\cdot\ \texttt{IU}\}$$
$$\Delta_2 = \{\texttt{IU}\mapsto \forall\texttt{FM.I},\ \texttt{STU}\mapsto \forall\texttt{FM.}\neg\texttt{I}\}$$

In a way similar to the one employed in Example 2.1 we can show that

$$\mathcal{E}_2 = TC(\Psi_2 \cup \{\texttt{p:}\forall\texttt{FM.}\neg\texttt{I}\})$$

is an extension of $\mathcal{T}_2$. This means that the inference according to which the faculty members of Pflunders University are typically not Italian is licensed, together with the ensuing fact that Professor Katzenjammer is not Italian. If we had to formalize this example in Nonmonotonic Logic we would have to impose the following set of axioms,

[6] To verify Condition 3b, see Footnote 4, observing that $\mathcal{E}' \cup \{\texttt{opus:}\forall\texttt{Fly.}\{\texttt{yes}\}, \langle\texttt{opus,a}\rangle\texttt{:Fly}\}$ is unsatisfiable for each individual constant `a`.

[7] For the sake of brevity we will use `p`, `k`, `IU`, `FM`, `I` and `STU` as abbreviations of `pflunders`, `katzenjammer`, `Italian_University`, `Faculty_Member`, `Italian` and `South_Tirolean_University`, respectively.

where exceptions are explicitly dealt with.

$$\forall x \forall y\ IU(x) \wedge FM(x,y) \wedge M[I(y) \wedge \neg STU(x)] \Rightarrow I(y)$$
$$\forall x\ STU(x) \Rightarrow IU(x)$$
$$\forall x \forall y\ STU(x) \wedge FM(x,y) \wedge M[\neg I(y)] \Rightarrow \neg I(y)$$
$$STU(p) \wedge FM(p,k)$$

As Axiom 5 shows, we must *explicit* the fact that a South Tyrolean university is an exceptional Italian university relatively to the citizenship of their faculty members, whereas in $\mathcal{TDL}$ this is not necessary. As we have seen in Section 1, this would lead to serious problems, especially in those cases in which the KB changes dynamically. ∎

Similarly to what happens in many non-monotonic formalisms (e.g. Non-monotonic Logic, Default Logic, Circumscription, MINEs), some $\mathcal{TDL}$ theories have more than one extension[8].

Proposition 2.6 *There exists a $\mathcal{TDL}$ theory with more than one extension.*

Proof: Consider the well known Nixon Diamond example. Let $\mathcal{T}_3 = <\Psi_3, \Delta_3>$ be a $\mathcal{TDL}$ theory, with

$$\Psi_3 = \{\texttt{nixon:Republican}, \texttt{nixon:Quaker}\}$$
$$\Delta_3 = \{\texttt{Republican} \mapsto \forall \texttt{Pacifist.}\{\texttt{no}\}, \texttt{Quaker} \mapsto \forall \texttt{Pacifist.}\{\texttt{yes}\}\}$$

This $\mathcal{TDL}$ theory intends to formalize the fact that republicans are typically not pacifists, while quakers are typically pacifists, and that Nixon is both a republican and a quaker. It is easy to see that

$$\mathcal{E}_{3_1} = TC(\Psi_3 \cup \{\texttt{nixon:}\forall \texttt{Pacifist.}\{\texttt{no}\}\})$$
$$\mathcal{E}_{3_2} = TC(\Psi_3 \cup \{\texttt{nixon:}\forall \texttt{Pacifist.}\{\texttt{yes}\}\})$$

are the only two extensions of $\mathcal{T}$. This accounts for the inherent ambiguity of the Nixon Diamond example (see Footnote 2). ∎

With respect to the issue of how many extensions a $\mathcal{TDL}$ theory can have, this number can be exponential with respect to the size of the $\mathcal{TDL}$ theory, as the following proposition shows.

Proposition 2.7 *There exists a $\mathcal{TDL}$ theory $\mathcal{T}$, such that the number of extensions of $\mathcal{T}$ is exponential with respect to the size of $\mathcal{T}$.*

Proof: Consider the $\mathcal{TDL}$ theory $\mathcal{T} = <\Psi, \Delta>$ (which we will call *multiple Nixon Diamond*).

$$\Psi = \{\texttt{a}_i\texttt{:A}_j \mid 1 \leq i \leq n, 1 \leq j \leq 2\}$$
$$\Delta = \{\texttt{A}_1 \mapsto \forall \texttt{S}_i.\texttt{P}_i \mid 1 \leq i \leq n\} \cup \{\texttt{A}_2 \mapsto \forall \texttt{S}_i.\neg \texttt{P}_i \mid 1 \leq i \leq n\}$$

$\mathcal{T}$ contains n^2 embedded Nixon Diamonds. Since for every Nixon Diamond we have two possibilities (i.e. whether

[8] In Circumscription extensions are called "minimal models".

the corresponding Ss of $\mathtt{a}_i$ are $\mathtt{P}_j$ or not), $\mathcal{T}$ has 2^{n^2} extensions. Therefore, considering that $|\mathcal{T}| = |\Psi|+|\Delta|$ is $O(n)$, the number of extensions of $\mathcal{T}$ is exponential with respect to the size of $\mathcal{T}$. ∎

Although the number of extensions can be very large in the worst case, as the multiple Nixon Diamond example shows, in actual knowledge representation applications this number need not be computationally discouraging. Furthermore, we might observe that this exponential number of extensions is not a characteristic of $\mathcal{TDL}$ itself, but is rather a common characteristic of almost all non-monotonic formalisms[9]. The multiple Nixon Diamond can be easily formulated in these formalisms, and gives rise to the same phenomenon; for example, in Nonmonotonic Logic we would write

$$\begin{array}{ll} A_j(a_i) & \text{for } 1 \leq i \leq n \text{ and } 1 \leq j \leq 2 \\ \forall x\ A_1(x) \wedge M[P_i(x)] \Rightarrow P_i(x) & \text{for } 1 \leq i \leq n \\ \forall x\ A_2(x) \wedge M[\neg P_i(x)] \Rightarrow \neg P_i(x) & \text{for } 1 \leq i \leq n \end{array}$$

obtaining the same number of extensions.

Unfortunately, some $\mathcal{TDL}$ theories may not have extensions.

Proposition 2.8 *There exists a $\mathcal{TDL}$ theory with no extensions.*

[9]One formalism that does not suffer from this problem is the "skeptical" version of MINEs (see [9]).

Proof: Let $\mathcal{T}_4 = <\Psi_4, \Delta_4>$ be a $\mathcal{TDL}$ theory, with

$$\Psi_4 = \{\mathtt{a}{:}\mathtt{A},\ \mathtt{B} \lessdot \mathtt{A},\ \langle \mathtt{a},\mathtt{a}\rangle{:}\mathtt{S}\}$$
$$\Delta_4 = \{\mathtt{A} \mapsto \forall \mathtt{S}.\mathtt{B},\ \mathtt{B} \mapsto \forall \mathtt{S}.\bot\}$$

Its only possible extensions might be

$$\mathcal{E}_{4_1} = TC(\Psi_4)$$
$$\mathcal{E}_{4_2} = TC(\Psi_4 \cup \{\mathtt{a}{:}\forall \mathtt{S}.\mathtt{B}\})$$

But $\mathcal{E}_{4_1}$ cannot be an extension, because $\Gamma(\mathcal{E}_{4_1}) = TC(\Psi_4 \cup \{\mathtt{a}{:}\forall \mathtt{S}.\mathtt{B}\})$ and $\Gamma(\mathcal{E}_{4_1}) \neq \mathcal{E}_{4_1}$; on the other hand, $\mathcal{E}_{4_2}$ cannot be an extension because $\Gamma(\mathcal{E}_{4_2}) = TC(\Psi_4)$ and $\Gamma(\mathcal{E}_{4_2}) \neq \mathcal{E}_{4_2}$. ∎

Theory $\mathcal{T}_4$ in Proposition 2.8 is a very simple theory; this shows that it would not be easy to find sublanguages of $\mathcal{TDL}$ such that the existence of atleast one extension is always guranteed.

Let us see why this happens. Let $\beta = i : \forall S.F$ and $\alpha = i : M$ be assertions, and Ω be any set of name introductions and assertions. It might well be that $\Omega \not\models \alpha$ and $\Omega \cup \{\beta\} \models \alpha$. In this case, if we were to add β to Ω as a result of the "application" of a default δ, we might be forced to consider other defaults having α as a precondition and a conclusion which conflicts with β. This might make δ "unapplicable".

If any language that does not suffer from this problem, the existence of an extension would be guaranteed; unfortunately, it is clear from what we have said before that removing from a language the causes of this problem would

seriously curtail the expressive power of the language itself.

In the following section we deal with the problem of computing an extension (whenever it exists) of a $\mathcal{TDL}$ theory.

3 Computing an extension of a $\mathcal{TDL}$ theory

In this section we will present an algorithm that computes an extension of a $\mathcal{TDL}$ theory. However, we will precede this by a discussion in which we briefly show how both the relation $\models$, restricted to the *instance problem*[10], and the relation $\preceq_\Omega$ can be reduced to deciding unsatisfiability, a problem that in our case, has well-known solutions (see for example [3, 7, 8]). The (nondeterministic) algorithm that computes an extensions of a $\mathcal{TDL}$ theory is heavily dependent on the computation of these relations. The proofs of the propositions and theorems listed in this section are given in an extended version of this paper [26].

3.1 The satisfiability problem in $\mathcal{TDL}$

We start by arguing that deciding $\preceq_\Omega$ can be reduced to the decision of $\models$.

[10]The instance problem is the problem of deciding whether $\Omega \models a{:}A$ holds.

Proposition 3.1 *Let Ω be a satisfiable set of name introductions and assertions, and M_1, M_2 be two elements of MP. Then $M_1 \preceq_\Omega M_2$ iff $\Omega \cup \{i{:}M_1\} \models i{:}M_2$ with i an individual constant not occuring in Ω.* ∎

In turn, the relation $\models$ restricted to the instance problem can be reduced to deciding unsatisfiability in the monotonic fragment of $\mathcal{TDL}$.

Proposition 3.2 *Let Ω be a satisfiable set of name introductions and assertions and $i{:}M$ be an assertion. Then $\Omega \models i{:}M$ iff $\Omega \cup \{i{:}\neg M\}$ is unsatisfiable* ∎

There exist well-known techniques for deciding unsatisfiability in terminological logics; an example is the one based on constraint propagation (see e.g. [7])[11]. By using this technique we have shown that it is decidable whether a finite and "acyclic" set of name introductions and assertions is unsatisfiable, where the acyclicity property is defined by the following definition.

Definition 3.1 *Let Ω be a finite set of name introductions and assertions. We say that Ω contains a* cycle *iff there exists a frame name M occurring in Ω such that M occurs in one of the frames*

[11]The unsatisfiability problem is the problem of deciding, given a set of name introductions and assertions Ω, if Ω is unsatisfiable.

which are obtained from the right hand side of M by iterated substitutions of some of the frame names by their right hand sides. ∎

In other words, a cycle is a term definition where, in the end, the *definiendum* is defined in terms of the *definiendum itself.*

The following proposition is a simple extension of Hollunder's results reported in [7].

Proposition 3.3 *Let Ω be a finite and acyclic set of name introductions and assertions. Then it is decidable if Ω is unsatisfiable.* ∎

3.2 An algorithm for computing extensions

To extend the result of the previous proposition to $\mathcal{TDL}$ *tout court* we will have to consider acyclic $\mathcal{TDL}$ theories only.

Definition 3.2 *Let $\mathcal{T} = <\Psi, \Delta>$ be a $\mathcal{TDL}$ theory. Then $\mathcal{T}$ is* acyclic *iff $\Phi = \{\tau |\ \tau$ is a name introduction in $\Psi\} \cup \{M \lessdot F\ :\ M \mapsto F \in \Delta\}$ is acyclic.* ∎

Definition 3.3 *Let $\mathcal{T} = <\Psi, \Delta>$ be an acyclic $\mathcal{TDL}$ theory and i an individual constant. An* istantiated default *of $\mathcal{T}$ is a pair $\gamma = \langle i, \delta \rangle$ with $\delta = M \mapsto \forall S.F$ and $\delta \in \Delta$. Moreover we define the function* Consequent *to be such that Consequent(γ)$=i{:}\forall S.F$* ∎.

Definition 3.4 *Let $\mathcal{T} = <\Psi, \Delta>$ be an acyclic $\mathcal{TDL}$ theory, Ω a set of name introductions and assertions, and $\gamma = \langle i_1, M_1 \mapsto \forall S.F_1 \rangle$ an istantiated default of $\mathcal{T}$. Then γ is an* applicable default *of $\mathcal{T}$ in Ω iff*

1. *$\Omega \models i_1 : M_1$;*
2. *for all M_2 such that $M_2 \preceq_\Omega M_1$ and $\langle i_1, M_2 \mapsto \forall S.F_2 \rangle$ is an istantiated default of $\mathcal{T}$, with $\Omega \models i_1 : M_2$, it is true that $\Omega \cup \{i_1 : \forall S.F_1 \sqcap F_2, \langle i_1, i_2 \rangle : S\}$ is satisfiable for some individual constant i_2.* ∎

We will write the expression $A_\mathcal{T}(\Omega)$ as short for the set of applicable defaults of $\mathcal{T}$ in Ω.

Given an acyclic $\mathcal{TDL}$ theory $\mathcal{T} = <\Psi, \Delta>$ and a satisfiable, finite and acyclic set of name introductions and assertions Ω, Δ and Ψ are finite sets; hence, the following proposition is a consequence of Proposition 3.3.

Proposition 3.4 *Let $\mathcal{T} = <\Psi, \Delta>$ be an acyclic $\mathcal{TDL}$ theory and Ω a finite and acyclic set of name introductions and assertions. Then the set of istantiated defaults of $\mathcal{T}$ and $A_\mathcal{T}(\Omega)$ are recursive sets.* ∎

Let us now discuss the EXT algorithm for computing extensions. The extension is built by a series of subsequent approximations. Each approximation

Ω_n is built from the first component Ψ of an acyclic $\mathcal{TDL}$ theory $\mathcal{T}$ by using applicable defaults, one at time. At each step, the istantiated default to be applied is chosen from those which have not yet been considered and which were applicable in the previous approximation and still are in the current state of the current approximation. When no more istantiated defaults are applicable, the algorithm continues with the next approximation. If two succesive approximations are the same sets, the algorithm is said to *converge*.

The choice of which istantiated default to apply at each step of the inner loop introduces in general a degree of nondeterminism. Generality requires this nondterminism, since terminological theories with defaults$\mathcal{TDL}$ theories need not have unique extensions (see Proposition 2.6). The algorithm is detailed in Figure 1.
To see how this algorithm works, let us consider the following example.

Let $\mathcal{T} = <\Psi, \Delta>$ be an acyclic $\mathcal{TDL}$ theory;
begin
  $\Omega_0 \leftarrow \Psi; n \leftarrow 0;$
  repeat
    $n \leftarrow n+1; \omega_0 \leftarrow \Psi; D_0 \leftarrow \emptyset; i \leftarrow 0;$
    repeat
      $A^i_{\mathcal{T}} \leftarrow (A_{\mathcal{T}}(\omega_i) \cap A_{\mathcal{T}}(\Omega_{n-1})) \setminus D_i;$
      if not $empty(A^i_{\mathcal{T}})$
        then choose γ **from** $A^i_{\mathcal{T}}$;
            $D_{i+1} \leftarrow D_i \cup \{\gamma\};$
            $\omega_{i+1} \leftarrow \omega_i \cup \{Consequent(\gamma)\};$
      endif
      $i \leftarrow i+1;$
    until $empty(A^{i-1}_{\mathcal{T}})$;
    $\Omega_n \leftarrow \omega_{i-1};$
  until $\Omega_n = \Omega_{n-1};$
end

Figure 1: The EXT algorithm computes an extension of a $\mathcal{TDL}$ theory.

Example 3.1 Let $\mathcal{T}_5 = <\Psi_5, \Delta_5>$ be a $\mathcal{TDL}$ theory, with

$$\Psi_5 = \{\mathtt{a}{:}\mathtt{A},\ \mathtt{b}{:}\mathtt{B},\ \langle \mathtt{a},\mathtt{a}\rangle{:}\mathtt{S},\ \mathtt{C} \doteq \mathtt{F} \sqcap \forall \mathtt{S}.\mathtt{E},\ \mathtt{B} \doteq \mathtt{F} \sqcap \forall \mathtt{S}.\{\mathtt{a}\}\}$$
$$\Delta_5 = \{\mathtt{A} \mapsto \forall \mathtt{S}.\mathtt{E},\ \mathtt{C} \mapsto \forall \mathtt{R}.\mathtt{D}\}$$

Let $\gamma_1 = \langle \mathtt{a}, \mathtt{A} \mapsto \forall \mathtt{S}.\mathtt{B}\rangle$ and $\gamma_2 = \langle \mathtt{b}, \mathtt{C} \mapsto \forall \mathtt{R}.\mathtt{D}\rangle$ be istantiated defaults of $\mathcal{T}$. The application of the EXT algorithm to $\mathcal{T}_5$ would yield the following sequence of steps:

1. $\Omega_0 = \omega_0 = \Psi, D_0 = \emptyset, A_{\mathcal{T}}(\omega_0) = A_{\mathcal{T}}(\Omega_0) = A^0_{\mathcal{T}} = \{\gamma_1\}$;
2. $\omega_1 = \omega_0 \cup \{Consequent(\gamma_1)\}, D_1 = \{\gamma_1\}$;
3. $A_{\mathcal{T}}(\omega_1) = \{\gamma_1, \gamma_2\}, A^1_{\mathcal{T}} = \emptyset$;
4. $\Omega_1 = \omega_1$;
5. $\omega_0 = \Psi, D_0 = \emptyset, A_{\mathcal{T}}(\omega_0) = A^0_{\mathcal{T}} = \{\gamma_1\}, A_{\mathcal{T}}(\Omega_0) = \{\gamma_1, \gamma_2\}$;
6. $\omega_1 = \omega_0 \cup \{Consequent(\gamma_1)\}, D_1 = \{\gamma_1\}$;
7. $A_{\mathcal{T}}(\omega_1) = \{\gamma_1, \gamma_2\}, A^1_{\mathcal{T}} = \{\gamma_2\}$;
8. $\omega_2 = \omega_1 \cup \{Consequent(\gamma_2)\}, D_1 = \{\gamma_1, \gamma_2\}$;
9. $A_{\mathcal{T}}(\omega_2) = \{\gamma_1, \gamma_2\}, A^2_{\mathcal{T}} = \emptyset$;
10. $\Omega_2 = \omega_2$;

    $\vdots$ a sequence similar to (5)-(10)

16. $\Omega_3 = \Omega_2 = \Psi \cup \{$a:$\forall$S.B, b:$\forall$R.D$\}$. ∎

In the following theorem we show that all and only the extensions of a $\mathcal{TDL}$ theory $\mathcal{T}$ can be computed by the algorithm.

Theorem 3.1 *Let $\mathcal{T} = <\Psi, \Delta>$ be a $\mathcal{TDL}$ theory. $\mathcal{E}$ is an extension of $\mathcal{T}$ iff the application of the* EXT *algorithm to $\mathcal{T}$ has a converging computation such that $\Omega_n = \Omega_{n-1}$ and $TC(\Omega_n) = \mathcal{E}$.* ∎

Note that, according to Theorem 3.1, the computation converges if and only if there exists an extension. The following example shows that, if there does not exist an extension, the computation does not converge.

Example 3.2 Consider the $\mathcal{TDL}$ theory $\mathcal{T}_4$ of Proposition 2.8. It is easy to show that each approximation Ω_i is such that $\Omega_{2k} = \Psi_4$ and $\Omega_{2k+1} = \Psi_4 \cup \{$a:$\forall$S.B$\}$, for each $k \geq 0$. Therefore $\Omega_{2k} \neq \Omega_{2k+1}$ for each $k \geq 0$, and the computation never stops. ∎

The next corollary follows from Proposition 3.3 and Theorem 3.1.

Corollary 3.1 *The set of extensions of an acyclic $\mathcal{TDL}$ theory is recursively enumerable.* ∎

Observe that Default Logic does not enjoy the same property.

4 Remarks about the computational complexity of $\mathcal{TDL}$

Notice that, according to Theorem 3.1, it is not necessary to compute an entire extension $\mathcal{E}$: the corresponding Ω_n is sufficient. In fact, in order to check if any relation of the type $M_1 \preceq_{\mathcal{E}} M_2$ or of the type $\mathcal{E} \models i : M_1$ holds, for some or all extensions $\mathcal{E}$ of a given $\mathcal{TDL}$

theory, it is sufficient to compute the corresponding Ω_n, since $M_1 \preceq_{\mathcal{E}} M_2$ or $\mathcal{E} \models a : M_1$ iff $M_1 \preceq_{\Omega_n} M_2$ or $\Omega_n \models a : M_1$.

Finally, we discuss some issues related to the computational complexity of computing $A_{\mathcal{T}}(\Omega)$. From Definition 3.4 it follows that this complexity depends on the complexity of $\models$ restricted to the instance problem and and on the complexity of $\preceq_\Omega$. Unfortunately, computing $\preceq_\Omega$ is inherently intractable, since we use name introductions. In fact, Nebel [16] has shown that the problem of checking if $M_1 \preceq_\Omega M_2$ is NP-Hard in our case[12]. Moreover, in order to compute $A_{\mathcal{T}}(\Omega)$, at each step we must recompute the relation $\preceq_\Omega$, since it is possible in principle that $\preceq_\Omega \neq \preceq_{\Omega\cup\{\alpha\}}$ for some assertion α. This increases the computational complexity enormously.

Interestingly enough, there is case in which this recomputation is not necessary. Let us consider the following restricted version of $\mathcal{TLD}$, which we will call $\mathcal{TLD}^-$.

Definition 4.1 *A* frame *in $\mathcal{TDL}^-$ is defined by the following syntax:*

$$\begin{array}{rcl} F_1, F_2 & \rightarrow & F_1 \sqcap F_2 \mid M \mid \neg M \mid \forall S.F_1 \mid \top \mid \bot \\ S & \rightarrow & D \quad \blacksquare \end{array}$$

[12]The problem is NP-Complete if Ω is only a set of name introductions of the form $M \doteq F$ where only the operators $\sqcap$ and $\forall$ occur.

Therefore, $\mathcal{TDL}^-$ is obtained from $\mathcal{TLD}$ by not allowing sets of individual constants to be frames.

The following proposition shows that $\mathcal{TDL}^-$ is more computationally interesting than $\mathcal{TLD}$.

Proposition 4.1 *Let Ω be a set of name introductions and assertions in $\mathcal{TDL}^-$, M_1 and M_2 two elements of MP, and α an assertion. Then $M_1 \preceq_\Omega M_2$ iff $M_1 \preceq_{\Omega\cup\{\alpha\}} M_2$.* $\blacksquare$

This proposition says that if we add an assertion α to Ω, $\preceq_\Omega$ does not change, i.e. $\preceq_\Omega = \preceq_{\Omega\cup\{\alpha\}}$. Therefore, if we want to compute an extension of a $\mathcal{TDL}$ theory $\mathcal{T} = <\Psi, \Delta>$, it is sufficient to compute $\preceq_\Psi$ as the first step of the computation, thus reducing the computational complexity. Furthermore, the exponential complexity of $\preceq_\Omega$ with respect to $\mathcal{TDL}^-$ is a worst case behaviour that seldom occurs. If we consider the average case characteristics of Ω, by using the result described by Sebastiani and Straccia in [25] it can be shown (as in Nebel [16]) that checking if $M_1 \preceq_\Omega M_2$ holds is computable in $O(s \log s)$, where $s = |\Omega|$.

Unfortunately, Proposition 4.1 does not hold for name introductions and assertions in $\mathcal{TDL}$. To see this consider $\Omega = \{\langle \mathtt{a},\mathtt{a}\rangle\!:\!\mathtt{S},\ \mathtt{A} \doteq \forall \mathtt{S}.\mathtt{B},\ \mathtt{B} \doteq \forall \mathtt{S}.\{\mathtt{a}\}\}$, $\alpha = \mathtt{a}\!:\!\forall \mathtt{S}.\mathtt{B}$. It turns out that $\mathtt{B} \not\preceq_\Omega \mathtt{A}$ but $\mathtt{B} \preceq_{\Omega\cup\{\alpha\}} \mathtt{A}$.

With respect to the relation $\models$ restricted to the instance problem, it is

easy to see that $\models$ must be recomputed at each step[13] but in $\mathcal{TDL}^-$ this is a less complex problem than in $\mathcal{TDL}$, since in the former case we can profitably use $\preceq_\Psi$ (see Nebel [15]).

5 Conclusion

In this paper we have shown how we can extend terminological logics in such a way that they allow a brand of default reasoning that obeys the specialization principle, thus creating a formalism (that we have dubbed $\mathcal{TDL}$) that combines

- the tools for describing taxonomic organizations of complex objects which are typical of TLs
- the ability to describe default information which is typical of MINEs
- the incrementality in KB construction which is typical of MINEs.

This has been obtained by relying on the notion of "extension of a $\mathcal{TDL}$ theory", a notion that has been defined in the style pioneered by Reiter in his Default Logic, i.e. as a fixpoint of a consequence relation. We have also studied a number of properties of $\mathcal{TDL}$ related to issues such as the existence and the uniqueness of extensions.

The language of $\mathcal{TDL}$ has been designed with the aim of providing a minimal framework allowing to study the interaction of terminological and default information in a meaningful way. Besides, an analysis of past (informal) work on terminological default reasoning reveals that the kind of defaults we have plugged into our language (namely, universally quantified defaults with consequents in a "slot-filler" form) are universally believed as the most natural way in which to convey default frame-like knowledge. Quite obviously, extensions to this framework may be conceived that enable the expression of default information of a nature different from the one considered here.

[13] If $\Omega = \{\langle \mathtt{a},\mathtt{a}\rangle\!:\!\mathtt{S}\}$, $\beta = \mathtt{a}\!:\!\mathtt{B}$ and $\alpha = \mathtt{a}\!:\!\forall \mathtt{S}\,.\,\mathtt{B}$, then $\Omega \cup \{\alpha\} \models \beta$, but $\Omega \not\models \beta$.

References

[1] Ronald J. Brachman. A structural paradigm for representing knowledge. Technical Report 3605, Bolt Beranek and Newman, Cambridge, MA, 1978.

[2] Gerhard Brewka. The logic of inheritance in frame systems. In *Proceedings of IJCAI-87, 10th International Joint Conference on Artificial Intelligence*, pages 483–488, Milano, Italy, 1987.

[3] Francesco M. Donini, Maurizio Lenzerini, and Daniele Nardi. An efficient method for hybrid deduction. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 246–252, Stockholm, Sweden, 1990.

[4] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. Tractable concept languages. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 458–463, Sidney, Australia, 1991.

[5] Jon Doyle and Drew McDermott. Nonmonotonic logic I. *Artificial Intelligence*, 13:41–72, 1980.

[6] Steve Hanks and Drew McDermott. Default reasoning, nonmonotonic logics and the frame problem. In *Proceedings of AAAI-86, 5th Conference of the American Association for Artificial Intelligence*, pages 328–333, Philadelphia, PA, 1986.

[7] Bernhard Hollunder. Hybrid inferences in KL-ONE-based knowledge representation systems. In *Proceedings of the 14th German Workshop on Artificial Intelligence*, pages 38–47, Eringerfeld, FRG, 1990.

[8] Bernhard Hollunder, Werner Nutt, and Manfred Schmidt-Schauß. Subsumption algorithms for concept description languages. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 348–353, Stockholm, Sweden, 1990.

[9] John F. Horty, Richmond H. Thomason, and David S. Touretzky. A skeptical theory of inheritance in nonmonotonic semantic nets. Technical Report CMU-CS-87-175, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, 1987.

[10] Alan K. Mackworth and Raymond Reiter. A logical framework for depiction and image interpretation. *Artificial Intelligence*, 41:125–155, 1989.

[11] John McCarthy. Circumscription - a form of nonmonotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[12] Marvin Minsky. A framework for representing knowledge. In Patrick J. Winston, editor, *The psychology of computer vision*, pages 211–277. McGraw-Hill, New York, NY, 1975.

[13] Robert C. Moore. Semantical considerations on nonmonotonic

logic. *Artificial Intelligence*, 25:75–94, 1985.

[14] Bernhard Nebel. Computational complexity of terminological reasoning in BACK. *Artificial Intelligence*, 34:371–383, 1988.

[15] Bernhard Nebel. *Reasoning and revision in hybrid representation systems*. Springer, Heidelberg, FRG, 1990.

[16] Bernhard Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249, 1990.

[17] Peter F. Patel-Schneider. Undecidability of subsumption in NIKL. *Artificial Intelligence*, 39:263–272, 1989.

[18] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[19] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987. [a] Appears also in [**?**], pp. 352–371.

[20] R. Bruce Roberts and Ira P. Goldstein. The FRL manual. Technical Report 409, The Artificial Intelligence Laboratory, MIT, Cambridge, MA, 1977.

[21] Klaus Schild. Undecidability of $\mathcal{U}$. Technical Report KIT 67, Department of Computer Science, Technische Universität Berlin, Berlin, FRG, 1988.

[22] Manfred Schmidt-Schauß. Subsumption in KL-ONE is undecidable. In *Proceedings of KR-89, 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431, Toronto, Ontario, 1989.

[23] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with unions and complements. Technical Report SR-88-21, FB Informatik, Universität Kaiserslautern, Kaiserslautern, Germany, 1988.

[24] Fabrizio Sebastiani. On heterogeneous model-preference default theories. In *Proceedings of CSCSI/SCEIO-90, 8th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, pages 84–91, Ottawa, Ontario, 1990.

[25] Fabrizio Sebastiani and Umberto Straccia. A computationally tractable terminological logic. In *Proceedings of SCAI-91, 3rd Scandinavian Conference on Artificial Intelligence*, pages 307–315, Roskilde, Denmark, 1991.

[26] Umberto Straccia. A terminological default logic (extended report). Technical report, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy, 1991.

[27] David S. Touretzky. *The mathematics of inheritance systems*. Pitman, London, GB, 1986.