Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems 160 (2009) 3382 – 3402

FUZZY sets and systems

www.elsevier.com/locate/fss

# Fuzzy description logics with general t-norms and datatypes☆

Fernando Bobillo[a,*], Umberto Straccia[b]

[a] *Department of Computer Science and Systems Engineering, University of Zaragoza, Spain*

[b] *Istituto di Scienza e Tecnologie dell'Informazione (ISTI), Consiglio Nazionale delle Ricerche (CNR), Pisa, Italy*

Received 21 May 2007; received in revised form 19 November 2008; accepted 19 March 2009

Available online 5 April 2009

**Abstract**

Fuzzy description logics (DLs) are a family of logics which allow the representation of (and the reasoning within) structured knowledge affected by vagueness. Although a relatively important amount of work has been carried out in the last years, current fuzzy DLs still present several limitations. In this work we face two problems: the common restriction to Zadeh and Łukasiewicz fuzzy logics and the inability to deal with datatypes different from fuzzy sets. In particular, we propose a semantics based on the use of a general left-continuous t-norm and an involutive negation (specially focused on Product logic) and, furthermore, we show how to handle functional concrete roles relating individuals of the domain and strings, real or integer numbers.



*Keywords:* Fuzzy description logics; Fuzzy ontologies; Fuzzy logic; Logic for the semantic web

## 1. Introduction

*Description logics* (DLs) [1] are a logical reconstruction of the so-called frame-based knowledge representation languages, with the aim of providing a simple well-established Tarski-style declarative semantics to capture the meaning of the most popular features of structured representation of knowledge. Nowadays, DLs have gained even more popularity due to their application in the context of the *semantic web* [3,4]. For example, the current standard language for specifying ontologies is the Web Ontology Language (OWL [32]), which comprises three sublanguages of increasing expressive power: OWL Lite, OWL DL and OWL Full [24]. OWL 2 is a recently proposed extension of OWL [13]. OWL Lite, OWL DL and OWL 2 are nearly equivalents to $\mathcal{SHIF}(\mathbf{D})$, $\mathcal{SHOIN}(\mathbf{D})$ and $\mathcal{SROIQ}(\mathbf{D})$ DLs, respectively, [13,23].

The problem to deal with imprecise concepts has been addressed several decades ago by Zadeh [50], which gave birth in the meanwhile to the so-called *fuzzy set and fuzzy logic theory* and a huge number of real life applications exist. In fuzzy logic, there are a lot of families of fuzzy operators (or fuzzy logics). The most important ones are Łukasiewicz, Gödel and Product [19]. We call here Zadeh family to the operators originally proposed by Zadeh in his seminal work [50]: Gödel conjunction and disjunction, Łukasiewicz negation and Kleene–Dienes implication. Table 1 shows the definition of these four families.

☆ Part of this work has been previously published in "A fuzzy description logic with product t-norm", in the Proceedings of the 16th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2007).

* Corresponding author. Ed. Ada Byron, C. María de Luna 1, 50018, Zaragoza, Spain. Tel.: +34 976 76 23 37.

*E-mail addresses:* fbobillo@unizar.es (F. Bobillo), straccia@isti.cnr.it (U. Straccia).

0165-0114/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2009.03.006

Table 1
Popular families of fuzzy operators.

| Family | t-norm $\alpha \otimes \beta$ | t-conorm $\alpha \oplus \beta$ | Negation $\ominus\alpha$ | Implication $\alpha \Rightarrow \beta$ |
|---|---|---|---|---|
| Zadeh | $\min\{\alpha, \beta\}$ | $\max\{\alpha, \beta\}$ | $1-\alpha$ | $\max\{1-\alpha, \beta\}$ |
| Łukasiewicz | $\max\{\alpha+\beta-1, 0\}$ | $\min\{\alpha+\beta, 1\}$ | $1-\alpha$ | $\min\{1-\alpha+\beta, 1\}$ |
| Product | $\alpha \cdot \beta$ | $\alpha+\beta-\alpha\cdot\beta$ | $\begin{cases} 1, & \alpha=0 \\ 0, & \alpha>0 \end{cases}$ | $\begin{cases} 1, & \alpha\leqslant\beta \\ \beta/\alpha, & \alpha>\beta \end{cases}$ |
| Gödel | $\min\{\alpha, \beta\}$ | $\max\{\alpha, \beta\}$ | $\begin{cases} 1, & \alpha=0 \\ 0, & \alpha>0 \end{cases}$ | $\begin{cases} 1 & \alpha\leqslant\beta \\ \beta, & \alpha>\beta \end{cases}$ |

Fuzzy set theory has been used to extend classical DLs by allowing to deal with *fuzzy/vague/imprecise concepts* for which a clear and precise definition is not possible. Since the first work of Yen in 1991 [49], an important number of works can be found in the literature. Straccia introduced assertional reasoning and provided reasoning algorithms for several reasoning tasks [41,45]. He also proposed a fuzzy extension of the expressive fuzzy DL $\mathcal{SHOIN}(\mathbf{D})$, although without considering reasoning [42]. Stoilos et al. have provided reasoning algorithms for some expressive DLs, such as $\mathcal{SHIN}$ [39], and have proposed fuzzy extensions of the languages OWL [38] and OWL 2 [36]. Reductions to crisp DLs have also been considered [5–8,11,36,46].

It is well known that different families of fuzzy operators lead to fuzzy DLs with different properties [37]. From a semantics point of view, the previous works rely on the semantics of fuzzy set operators proposed by Zadeh [50]. However, some applications require the use of new fuzzy operators.

**Example 1.1.** Product t-norm has been used in the context of information retrieval with fuzzy ontologies [28]. Consider the following concept hierarchy: $A_1 \sqsubseteq A_2 \geqslant \alpha_1$, $A_2 \sqsubseteq A_3 \geqslant \alpha_2$, $A_3 \sqsubseteq A_4 \geqslant \alpha_3, \ldots, A_n \sqsubseteq A_{n+1} \geqslant \alpha_n$, where $A_i \sqsubseteq A_j \geqslant \alpha_i$ means informally that $A_j$ can be considered more general than $A_i$ to degree $\alpha_i$.

If the retrieval algorithm decides to retrieve $A_1$, it seems natural to retrieve also more general concepts with different degrees computed using a t-norm $\otimes$. For example, suppose that $A_1$ is found to be retrieved with degree $\alpha_0$. Then, $A_2$ can be retrieved with a degree $\alpha_0 \otimes \alpha_1$ and, in general, $A_i$ can be retrieved with a degree $\alpha_0 \otimes \alpha_1 \otimes \cdots \otimes \alpha_n$, for $i = \{0, 1, \ldots, n\}$.

A desirable property for the t-norm is that the final degree should reflect the "distance" in the ontology, that is, the greatest the distance to concept $A_1$, the less the degree of retrieval. Hence, we need to consider a sub-idempotent t-norm such as the product.

There are some few works that consider alternative fuzzy operators. Straccia [43,44] and Straccia et al. [47] propose a reasoning solution for Łukasiewicz family, which is based on a mixture of tableau rules and Mixed Integer Linear Programming (MILP) and is implemented in the *fuzzyDL* reasoner [10]. Some recent works consider reductions of fuzzy DLs to their crisp versions under Łukasiewicz [11] and Gödel [8] semantics. But neither the Product family nor different t-norm have received enough attention. An exception is due to Hájek, who considered fuzzy $\mathcal{ALC}$ under arbitrary continuous t-norms and reports some reasoning algorithms based on a reduction to fuzzy propositional logic [18]. For a more detailed survey on fuzzy DLs the reader is referred to [30].

On the other hand, some fuzzy DLs offer the possibility to use datatypes by means of fuzzy concrete domains (i.e., the possibility to represent in fuzzy DLs concepts with explicit membership functions such as triangular, trapezoidal, left-shoulder and right-shoulder functions) [43]. Nevertheless, nowadays it is not possible to use alternative datatypes such as strings and integers. For instance, it is not possible to define the concept of people who are not old enough to vote as $Person \sqcap \neg(\geqslant hasAge\ 18)$.

The contributions of this paper are twofold. On the one hand, we consider fuzzy DLs under general t-norms and an involutive negation, and provide a reasoning algorithm different to that in [18]. We also show how to particularize it to the case of the Product t-norm, and how to reduce the number of generated variables with respect to similar algorithms.

On the other hand, we allow the use of functional concrete roles relating an individual with a string or a (real or integer) number. To the best of our knowledge, this is the first attempt in this direction.

The remainder of the paper is organized as follows. In the next section we refresh some necessary background on some optimization problems. Section 3 presents a general fuzzy extension of the DL $\mathcal{ALC}$. Section 4 propose a reasoning algorithm under general t-norms. Next, Section 5 particularizes it with a semantics based on the Product t-norm, discusses some logical properties and addresses the inference algorithm. Section 6 gives a step further and considers fuzzy $\mathcal{ALCF}(\mathbf{D})$, presenting the syntax, semantics and reasoning rules for functional concrete roles. Finally, Section 7 sets out some conclusions and ideas for future work.

## 2. Background: MILP, MIQCP and MINLP problems

In this section we recall Mixed Integer Linear Programming, Mixed Integer Quadratically Constrained Programming (MIQCP) and Mixed Integer NonLinear Programming (MINLP) optimization problems.

*MILP*: A general Mixed Integer Linear Programming [34] problem consists in minimizing a linear function with respect to a set of constraints that are linear inequations in which rational and integer variables can occur. More precisely, let $x = \langle x_1, \ldots, x_k \rangle$ and $y = \langle y_1, \ldots, y_m \rangle$ be variables over $\mathbb{Q}$ and $\mathbb{Z}$, respectively, over the integers and let $A$, $B$ be integer matrices and $h$ an integer vector. The variables in $y$ are called *control variables*. Let $f(x, y)$ be a $k+m$-ary linear function. Then the general MILP problem is to find $\bar{x} \in \mathbb{Q}^k$, $\bar{y} \in \mathbb{Z}^m$ such that $f(\bar{x}, \bar{y}) = \min\{f(x, y) \mid Ax + By \geqslant h\}$.

The general case can be restricted to what concerns the paper as we can deal with *bounded* MILP (bMILP). That is, the rational variables usually range over [0, 1], while the integer variables range over {0, 1}. It is well known that the bMILP problem is NP-COMPLETE (for the belonging to NP, guess the $y$ and solve in polynomial time the linear system, NP-hardness follows from NPHardness of 0–1 Integer Programming). Furthermore, we say that $M \subseteq [0, 1]^k$ is bMILP-representable iff there is a bMILP $(A, B, h)$ with $k$ real and $m$ 0–1 variables such that $M = \{x : \exists y \in \{0, 1\}^m$ such that $Ax + By \geqslant h\}$.

In general, we require that every constructor is bMILP representable. For instance, classical logic, Zadeh's fuzzy logic, and Łukasiewicz connectives, are bMILP-representable, while Gödel negation is not. In general, connectives whose graph can be represented as the union of a finite number of convex polyhedra are bMILP-representable [27], however, discontinuous functions may not be bMILP representable.

There are a lot of available tools for solving these problems, such as `Cbc`[1] or `lpsolve`.[2]

*MIQCP*: Let $x = \langle x_1, \ldots, x_k \rangle$ and $y = \langle y_1, \ldots, y_m \rangle$ be variables over $\mathbb{Q}$ and $\mathbb{Z}$, respectively. Now, for all $i \in 0, 1, \ldots, n$, let $a_i$ be an integer vector of length $k$, $b_i$ be an integer vector of length $m$, $h_i$ be an integer number and $Q_i(x, y) = 1/2 \cdot (x + y)^T \cdot C_i \cdot (x + y)$, where $C_i$ is a symmetric integer matrix of dimension $(k + m) \times (k + m)$. Let $f(x, y)$ be a $k + m$-ary linear function. The MICQP problem is to find $\bar{x} \in \mathbb{Q}^k$, $\bar{y} \in \mathbb{Z}^m$ such that $f(\bar{x}, \bar{y}) = \min\{f(x, y) : a_0 \cdot x + b_0 \cdot y \geqslant h_0\}$ subject to a set of $n$ constraints of the form: $a_i \cdot x + b_i \cdot y + Q_i(x, y) \geqslant h_i$ or $a_i \cdot x + b_i \cdot y + Q_i(x, y) \leqslant h_i$, for all $i = 1, \ldots, n$. Notice that the objective function is linear, while the restrictions can contain quadratic sections.

The general case can be restricted to what concerns the paper as we can deal with bounded MIQCP (bMIQCP), with rational variables ranging over [0, 1] and integer variables range over {0, 1}. $M \subseteq [0, 1]^k$ is bMICQP-representable iff there is a bMICQP $(a_i, b_i, C_i, h_i)$ with $k$ real and $m$ 0–1 variables such that $M = \{x : \exists y \in \{0, 1\}^m$ such that $a_i \cdot x + b_i \cdot y + Q_i(x, y) \geqslant h_i$ or $a_i \cdot x + b_i \cdot y + Q_i(x, y) \leqslant h_i$.

This problem is known to be NP-HARD. Some examples of solvers are `CPLEX`[3] or `mosek`.[4]

*MINLP*: Let $x = \langle x_1, \ldots, x_k \rangle$ and $y = \langle y_1, \ldots, y_m \rangle$ be variables over $\mathbb{Q}$ and $\mathbb{Z}$, respectively, and, for all $i \in 0, 1, \ldots, n$, let $h_i$ be an integer number, and $f_i(x, y)$ be a $k + m$-ary (possibly nonlinear) function. The Mixed Integer NonLinear Programming problem [16] is to find $\bar{x} \in \mathbb{Q}^k$, $\bar{y} \in \mathbb{Z}^m$ such that $f_0(\bar{x}, \bar{y}) = \min_{x \in \bar{x}, y \in \bar{y}} \{f_0(x, y)\}$ subject to a set of $n$ constraints of the form: $f_i(x, y) \geqslant h_i$, for all $i = 1, \ldots, n$.

As in the previous cases, in the bounded MINLP (bMINLP), rational variables range over [0, 1] and integer variables range over {0, 1}. The problem is NP-HARD, and there some available solvers, such as `Bonmin`.[5]

[1] http://www.coin-or.org/projects/Cbc.xml
[2] http://sourceforge.net/projects/lpsolve
[3] http://www.ilog.com/products/cplex/
[4] http://www.mosek.com/
[5] http://www.coin-or.org/projects/Bonmin.xml

## 3. Fuzzy description logics

In this section we define a fuzzy extension of $\mathcal{ALC}$ [35]. We recall here the semantics given in [18,42,43,45].

### 3.1. Syntax

Now, let $\mathbf{A}$, $\mathbf{R}_A$ and $\mathbf{I}$ be non-empty enumerable and pair-wise disjoint sets of *concept names* (denoted $A$), *abstract role names* (denoted $R$) and *abstract individual names* (denoted $a, b$). Concepts may be seen as unary predicates, while roles may be seen as binary predicates.

$\mathcal{ALC}$ complex concepts can be built according to the following syntax rule:

$$C := \top \mid \bot \mid A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid C_1 \rightarrow C_2 \mid \neg C \mid \forall R.C \mid \exists R.C.$$

An $\mathcal{ALC}$ *fuzzy knowledge base* (fuzzy KB) $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a fuzzy TBox $\mathcal{T}$, and a fuzzy ABox $\mathcal{A}$.

In general, a *fuzzy TBox* $\mathcal{T}$ is a finite set of *fuzzy concept inclusion axioms* $\langle C \sqsubseteq D, n \rangle$, where $C, D$ are concepts and $n \in (0, 1]$. Informally, $\langle C \sqsubseteq D, n \rangle$ states that all instances of concept $C$ are instances of concept $D$ to degree $n$, i.e. the subsumption degree between $C$ and $D$ is at least $n$. We write $C = D$ as a shorthand of the two axioms $\langle C \sqsubseteq D, 1 \rangle$ and $\langle D \sqsubseteq C, 1 \rangle$.

However, for computational reasons, we will restrict TBoxes to be *acyclic*. That is, $\mathcal{T}$ is a finite set of *fuzzy concept inclusion axioms* $\langle A \sqsubseteq C, n \rangle$, and *concept definitions* $A = C$, where $A$ is an atomic concept. Furthermore, we assume that $\mathcal{T}$ verifies two additional constraints:

- There is no concept $A$ such that it appears more than once on the left-hand side of some axiom in $\mathcal{T}$.
- No cyclic definitions are present in $\mathcal{T}$.[6]

It is well known that such TBoxes can be eliminated through a finite (although it can create an exponential growth of the KB), *expansion process*, both in the crisp [33] and in the fuzzy case [45]. Instead, we will use a extension to the fuzzy case of the *lazy expansion* technique [2], which has proved to be more useful in practice.

A *fuzzy ABox* $\mathcal{A}$ consists of a finite set of *fuzzy concept* and *fuzzy role assertion axioms* of the form $\langle a{:}\, C, n \rangle$ and $\langle (a, b){:}\, R, n \rangle$, where $a, b$ are abstract individual names, $C$ is a concept and $R$ is an abstract role. Informally, from a semantical point of view, a fuzzy axiom $\langle \tau, n \rangle$ constrains the membership degree of $\tau$ to be at least $n$.

**Example 3.1.** A fuzzy KB is $K = \{\mathcal{T}, \mathcal{A}\}$ with $\mathcal{T} = \{\langle \mathit{Inn} \sqsubseteq \mathit{Hotel}, 0.5 \rangle\}$ and $\mathcal{A} = \{\langle \mathit{jim}{:}\, \mathit{YoungPerson}, 0.2 \rangle, \langle (\mathit{jim}, \mathit{mary}){:}\, \mathit{likes}, 0.8 \rangle\}$. The terminological axiom $\langle \mathit{Inn} \sqsubseteq \mathit{Hotel}, 0.5 \rangle$ means that every inn can be considered a hotel with degree at least 0.5. $\langle \mathit{jim}{:}\, \mathit{YoungPerson}, 0.2 \rangle$ says that *jim* is a *YoungPerson* with degree at least 0.2, while $\langle (\mathit{jim}, \mathit{mary}){:}\, \mathit{likes}, 0.8 \rangle$, says that *jim* likes *mary* with degree at least 0.8.

### 3.2. Semantics

The semantics extends [45]. The main idea is that concepts and roles are interpreted as fuzzy subsets and fuzzy relations over an interpretation domain. Therefore, $\mathcal{ALC}$ axioms, rather than being satisfied (true) or unsatisfied (false) in an interpretation, become a degree of truth in [0, 1].

In the following, we use $\otimes, \oplus, \ominus$ and $\Rightarrow$ in infix notation, in place of a t-norm, t-conorm, negation function and implication function. A *fuzzy interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set $\Delta^{\mathcal{I}}$ (the *domain*) and of a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$ that assigns:

1. to each abstract concept $C$ a function $C^{\mathcal{I}}{:}\, \Delta^{\mathcal{I}} \rightarrow [0, 1]$;
2. to each abstract role $R$ a function $R^{\mathcal{I}}{:}\, \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \rightarrow [0, 1]$;
3. to each abstract individual $a$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

We also assume the *unique names assumption* over the individuals, i.e., if $a \neq b$ then $a^{\mathcal{I}} \neq b^{\mathcal{I}}$, where $a, b$ are individuals (different individuals denote different objects of the domains).

[6] We will say that $A$ *directly uses* primitive concept $B$ in $\mathcal{T}$, if there is some axiom $\tau \in \mathcal{T}$ such that $A$ is on the left-hand side of $\tau$ and $B$ occurs in the right-hand side of $\tau$. Let *uses* be the transitive closure of the relation *directly uses* in $\mathcal{T}$. $\mathcal{T}$ is cyclic iff there is $A$ such that $A$ uses $A$ in $\mathcal{T}$.

Table 2
Fuzzy DL semantics for $\mathcal{ALC}$.

$$\begin{aligned}
\bot^{\mathcal{I}}(x) &= 0\\
\top^{\mathcal{I}}(x) &= 1\\
(C_1 \sqcap C_2)^{\mathcal{I}}(x) &= C_1{}^{\mathcal{I}}(x) \otimes C_2{}^{\mathcal{I}}(x)\\
(C_1 \sqcup C_2)^{\mathcal{I}}(x) &= C_1{}^{\mathcal{I}}(x) \oplus C_2{}^{\mathcal{I}}(x)\\
(C_1 \rightarrow C_2)^{\mathcal{I}}(x) &= C_1{}^{\mathcal{I}}(x) \Rightarrow C_2{}^{\mathcal{I}}(x)\\
(\neg C)^{\mathcal{I}}(x) &= \ominus C^{\mathcal{I}}(x)\\
(\forall R.C)^{\mathcal{I}}(x) &= \inf_{y \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)\\
(\exists R.C)^{\mathcal{I}}(x) &= \sup_{y \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)
\end{aligned}$$

The fuzzy interpretation function is extended to roles and complex concepts as specified in Table 2 (where $x, y \in \Delta^{\mathcal{I}}$ are elements of the domain).

Finally, $\cdot^{\mathcal{I}}$ is extended to non-fuzzy axioms as specified below:

$$\begin{aligned}
(C \sqsubseteq D)^{\mathcal{I}} &= \inf_{x \in \Delta^{\mathcal{I}}} C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x),\\
(a{:}\, C)^{\mathcal{I}} &= C^{\mathcal{I}}(a^{\mathcal{I}}),\\
((a,b){:}\, R)^{\mathcal{I}} &= R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}).
\end{aligned}$$

Note here that, e.g., the semantics of a concept inclusion axiom $C \sqsubseteq D$ is derived directly from its FOL translation, which is of the form $\forall x. F_C(x) \Rightarrow F_D(x)$. This definition is clearly different from the approaches in which $C \sqsubseteq D$ is viewed as $\forall x. C(x) \leqslant D(x)$ [40,45]. This latter approach has the effect that the subsumption relationship is a Boolean relationship, while in our approach subsumption is determined up to a degree in [0, 1].

A fuzzy interpretation $\mathcal{I}$ is *witnessed* [18,20,21] iff it verifies:

- for all $x \in \Delta^{\mathcal{I}}$, there is $y \in \Delta^{\mathcal{I}}$ such that $(\exists R.C)^{\mathcal{I}} = R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)$,
- for all $x \in \Delta^{\mathcal{I}}$, there is $y \in \Delta^{\mathcal{I}}$ such that $(\forall R.C)^{\mathcal{I}} = R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)$,
- there is $x \in \Delta^{\mathcal{I}}$ such that $(C \sqsubseteq D)^{\mathcal{I}} = C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x)$.

In the rest of the paper we will assume that $\mathcal{I}$ is a witnessed fuzzy interpretation. Since fuzzy DLs are used in knowledge representation, we argue that non-witnessed models are not interesting; what is interesting for us are those role fillers which can be represented by specifying some particular individual of the domain. Quoting Hájek, "for the aims of description logic non-witnessed models appear to be pathological" [21].

The notion of *satisfaction* of a fuzzy axiom $\tau$ by a witnessed fuzzy interpretation $\mathcal{I}$, denoted $\mathcal{I} \models E$, is defined as follows: $\mathcal{I} \models \langle \tau \geqslant n \rangle$, where $\tau$ is a concept inclusion, a concept or a role assertion axiom, iff $\tau^{\mathcal{I}} \geqslant n$.

We say that a concept $C$ is *satisfiable* iff there is a witnessed interpretation $\mathcal{I}$ and an individual $x \in \Delta^{\mathcal{I}}$ such that $C^{\mathcal{I}}(x) > 0$. For example, the concept $C \sqcap (\neg C)$ is unsatisfiable under Łukasiewicz, Gödel or Product families, but satisfiable under Zadeh family.

For a set of fuzzy axioms $\mathcal{E}$, we say that a witnessed model $\mathcal{I}$ *satisfies* $\mathcal{E}$ iff $I$ satisfies each element in $\mathcal{E}$. We say that $\mathcal{I}$ is a *model* of $\tau$ (resp. $\mathcal{E}$) iff $\mathcal{I} \models \tau$ (resp. $\mathcal{I} \models \mathcal{E}$). $\mathcal{I}$ *satisfies* (is a *model* of) a fuzzy knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, denoted $\mathcal{I} \models \mathcal{K}$, iff $\mathcal{I}$ is a model of each component $\mathcal{T}$ and $\mathcal{A}$, respectively.

A fuzzy axiom $\tau$ is a *logical consequence* of a knowledge base $\mathcal{K}$, denoted $\mathcal{K} \models \tau$ iff every witnessed model of $\mathcal{K}$ satisfies $\tau$.

Given $\mathcal{K}$ and an axiom $\tau$ of the form $C \sqsubseteq D$, $a{:}\, C$ or $(a,b){:}\, R$, it is of interest to compute $\tau$'s best entailment degree (BED).

Lower degree value bound. The *greatest lower bound* of $\tau$ w.r.t. $\mathcal{K}$ (denoted $glb(\mathcal{K}, \tau)$) is $glb(\mathcal{K}, \tau) = \sup\{n \mid \mathcal{K} \models \langle \tau \geqslant n \rangle\}$, where $\sup \emptyset = 0$. Determining the *glb* is called the *best entailment degree* problem. For example, as we will see in Example 5.3, given the $\mathcal{K} = \{\langle (a,b) : R, 0.7 \rangle, \langle b{:}\, C, 0.8 \rangle\}$, $glb(\mathcal{K}, a{:}\, \exists R.C) = 0.56$ under the Product t-norm.

Finally, the *best satisfiability degree* of a concept $C$ and amounts to determine $bsd(\mathcal{K}, C) = \sup_{\mathcal{I} \models \mathcal{K}} \sup_{x \in \Delta^{\mathcal{I}}} \{C^{\mathcal{I}}(x)\}$. Essentially, among all models $\mathcal{I}$ of the knowledge base, we are determining the maximal degree of truth that the concept $C$ may have over all individuals $x \in \Delta^{\mathcal{I}}$.

## 4. Fuzzy DLs with general t-norms

### 4.1. Families of fuzzy operators

Given a left-continuous t-norm $\otimes$, in this section we consider arbitrary families of fuzzy operators of the following form:

$$\begin{aligned} \ominus \alpha &= 1-\alpha, \\ \alpha \oplus \beta &= \ominus(\ominus\alpha \otimes \ominus\beta), \\ \alpha \Rightarrow \beta &= \sup_{\gamma\in[0,1]} \{\alpha \otimes \gamma \leqslant \beta\}. \end{aligned}$$

That is, we have a left-continuous t-norm $\otimes$, its residuum $\Rightarrow$, Łukasiewicz negation $\ominus$ and the t-conorm $\oplus$ which is dual to $\otimes$ with respect to this negation. We are doubtful about the practical interest of non-involutive negations such as Gödel negation and, thus, we prefer to use a continuous and involutive negation i.e., a negation verifying $\ominus(\ominus\alpha) = \alpha$.

Note that a second negation can still be defined since $\ominus_{\otimes} C = C \Rightarrow \bot$, so we actually allow two negations: Łukasiewicz and the negation of the logic of $\otimes$.

Due to the standard properties of the fuzzy operators, the following concept equivalences hold [45]: $\neg\top = \bot$, $\neg\bot = \top$, $C \sqcap \top = C$, $C \sqcup \bot = C$, $C \sqcap \bot = \bot$, $C \sqcup \top = \top$, $\exists R.\bot = \bot$, $\forall R.\top = \top$.

In this logic, however, we have in general that $\neg\forall R.C \neq \exists R.(\neg C)$ and $\neg\exists R.C \neq \forall R.(\neg C)$. This is interesting since assuming the inter-definability of quantifiers is generally an unnecessary restriction [21]. Note that the equality holds for Łukasiewicz logic.

However, De Morgan laws are still verified, i.e., $\neg(C \sqcap D) = (\neg C) \sqcup (\neg D)$ and $\neg(C \sqcup D) = (\neg C) \sqcap (\neg D)$. Note also that, in general, $C \neq C \sqcap C$ and $C \neq C \sqcup C$.

### 4.2. Idea of the reasoning algorithm

For crisp DLs, the overall proof method is as follows (see, e.g., [26]): they show that a KB is satisfiable iff there is a tableau for it, where a tableau is a particular mathematical structure from which a model may easily be built. Then they provide a terminating algorithm, which builds a so-called completion forest, from which a tableau may be derived. Finally, they show that there is a tableau for a KB iff there is a clash-free completion tree for the KB and, thus, a KB is satisfiable iff there is a clash-free completion forest for it. The proof method for fuzzy DLs is essentially similar to the crisp variant. So we will only highlight the differences, while the rest can be worked out similarly.

The basic idea behind our reasoning algorithm is as follows. Consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is acyclic. In order to solve the BDB problem, we combine appropriate DL tableaux rules with methods developed in the context of *many-valued logics* (MVLs) [17]. In order to determine e.g., $glb(\mathcal{K}, a\colon C)$, we consider an expression of the form $\langle a\colon \neg C, 1-x\rangle$ (informally, $\langle a\colon C \leqslant x\rangle$), where $x$ is a [0, 1]-valued variable. Then we construct a tableaux for $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \cup \{\langle a\colon \neg C, 1-x\rangle\}\rangle$ in which the application of satisfiability preserving rules generates new fuzzy assertion axioms together with *inequations* over [0, 1]-valued variables. These inequations have to hold in order to respect the semantics of the DL constructors. Finally, in order to determine the greatest lower bound, we *minimize* the original variable $x$ such that all constraints are satisfied.[7]

Similarly, for $C \sqsubseteq D$, we can compute $glb(\mathcal{K}, C \sqsubseteq D)$ as the minimal value of $x$ such that $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \cup \{\langle a : C, x_1\rangle\} \cup \{\langle a : D, x_2\rangle\}\rangle$ is satisfiable under the constraints expressing that $x_1 \Rightarrow x_2 \leqslant x,\ x_1 \in [0,1]$ and $x_2 \in [0,1]$, where $a$ is new abstract individual. For a concrete example, see Section 5.1.

Finally, $glb(\mathcal{K}, (a, b)\colon R)$ is equivalent to $glb(\mathcal{K} \cup \{\langle b : B, 1\rangle\}, a\colon \exists R.B)$, where $B$ is a new concept (which does not appear in $\mathcal{K}$).

Therefore, the BDB problem can be reduced to minimal satisfiability problem of a KB. Finally, concerning the best satisfiability bound problem, $glb(\mathcal{K}, C)$ is determined by the maximal value of $x$ such that $\langle \mathcal{T}, \mathcal{A} \cup \{\langle a\colon C, x\rangle\}\rangle$ is satisfiable.

In general, we end up with a bMINLP problem, but in some particular cases we can have an easier problem. For example, for Łukasiewicz t-norm we end up with a bMILP problem [43,47], while for the Product t-norm we end up

[7] Informally, suppose the minimal value is $\bar{n}$. We will know then that for any interpretation $\mathcal{I}$ satisfying the knowledge base such that $(a\colon C)^{\mathcal{I}} < \bar{n}$, the starting set is unsatisfiable and, thus, $(a\colon C)^{\mathcal{I}} \geqslant \bar{n}$ has to hold. Which means that $glb(\mathcal{K}, (a\colon C)) = \bar{n}$.

with a bMICQP problem (see Section 5). Interestingly, the tableaux contains only one branch only and, thus, just one bMINLP problem has to be solved.

### 4.3. A fuzzy tableau

Now, let $\mathbf{V}$ be a new alphabet of variables $x$ ranging over [0, 1], $\mathbf{W}$ be a new alphabet of 0–1 variables $y$. We extend fuzzy assertions to the form $\langle \tau, l \rangle$, where $l$ is an arithmetic expression over variables in $\mathbf{V}$, $\mathbf{W}$ and real values.

Similar to crisp DLs, our tableaux algorithm checks the satisfiability of a fuzzy KB by trying to build a fuzzy tableau, from which it is immediate either to build a model in case KB is satisfiable or to detect that the KB is unsatisfiable. The fuzzy tableau we present here can be seen as an extension of the tableau presented in [26], and is inspired by the one presented in [39,40].

Given $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, let $\mathbf{R}_{\mathcal{K}}$ be the set of roles occurring in $\mathcal{K}$ and let $sub(\mathcal{K})$ be the set of named concepts appearing in $\mathcal{K}$. A *fuzzy tableau* $T$ for $\mathcal{K}$ is a quadruple $(\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{V})$ such that: $\mathbf{S}$ is a set of elements, $\mathcal{L} : \mathbf{S} \times sub(\mathcal{K}) \to [0, 1]$ maps each element and concept, to a membership degree (the degree of the element being an instance of the concept), and $\mathcal{E} : \mathbf{R}_{\mathcal{K}} \times (\mathbf{S} \times \mathbf{S}) \to [0, 1]$ maps each role of $\mathbf{R}_{\mathcal{K}}$ and pair of elements to the membership degree of the pair being an instance of the role, and $\mathcal{V} : \mathbf{I}_{\mathcal{A}} \to \mathbf{S}$ maps individuals occurring in $\mathcal{A}$ to elements in $\mathbf{S}$. For all $s, t \in \mathbf{S}$, $C, D \in sub(\mathcal{K})$, and $R \in \mathbf{R}_{\mathcal{K}}$, $T$ has to satisfy:

1. $\mathcal{L}(s, \bot) = 0$ and $\mathcal{L}(s, \top) = 1$ for all $s \in \mathbf{S}$,
2. If $\mathcal{L}(s, \neg A) \geqslant n$, then $\mathcal{L}(s, A) \leqslant \ominus n$.
3. If $\mathcal{L}(s, \neg\neg C) \geqslant n$, then $\mathcal{L}(s, C) \geqslant n$.
4. If $\mathcal{L}(s, C \sqcap D) \geqslant n$, then $\mathcal{L}(s, C) \geqslant m_1$, $\mathcal{L}(s, D) \geqslant m_2$ and $n = m_1 \otimes m_2$, for some $m_1$ and $m_2$.
5. If $\mathcal{L}(s, \neg(C \sqcap D)) \geqslant n$, then $\mathcal{L}(s, \neg C \sqcup \neg D) \geqslant n$.
6. If $\mathcal{L}(s, C \sqcup D) \geqslant n$, then $\mathcal{L}(s, C) \Rightarrow \mathcal{L}(s, D) \geqslant n$.
7. If $\mathcal{L}(s, \neg(C \sqcup D)) \geqslant n$, then $\mathcal{L}(s, \neg C \sqcap \neg D) \geqslant n$.
8. If $\mathcal{L}(s, C \to D) \geqslant n$, then $\mathcal{L}(s, C) \geqslant m_1$, $\mathcal{L}(s, D) \geqslant m_2$ and $n = m_1 \Rightarrow m_2$, for some $m_1$ and $m_2$.
9. If $\mathcal{L}(s, \neg(C \to D)) \geqslant n$, then $\mathcal{L}(s, C) \Rightarrow \mathcal{L}(s, D) \leqslant 1 - n$.
10. If $\mathcal{L}(s, \forall R.C) \geqslant n$, then $\mathcal{L}(t, C) \geqslant \mathcal{E}(R, \langle s, t \rangle) \otimes n$, for all $t \in \mathbf{S}$.
11. If $\mathcal{L}(s, \neg\forall R.C) \geqslant n$, then there exists $t \in \mathbf{S}$ such that $\mathcal{E}(R, \langle s, t \rangle) \Rightarrow \mathcal{L}(t, C) \leqslant 1 - n$.
12. If $\mathcal{L}(s, \exists R.C) \geqslant n$, then there exists $t \in \mathbf{S}$ such that $\mathcal{E}(R, \langle s, t \rangle) \geqslant m_1$, $\mathcal{L}(t, C) \geqslant m_2$ and $n = m_1 \otimes m_2$, for some $m_1$ and $m_2$.
13. If $\mathcal{L}(s, \neg\exists R.C) \geqslant n$, then $\mathcal{E}(R, \langle s, t \rangle) \otimes \mathcal{L}(t, C) \leqslant 1 - n$, for all $t \in \mathbf{S}$.
14. If $\langle C \sqsubseteq D, n \rangle \in \mathcal{T}$, then $\mathcal{L}(s, D) \geqslant \mathcal{L}(s, C) \otimes n$, for all $s \in \mathbf{S}$.
15. If $\langle a : C, n \rangle \in \mathcal{A}$, then $\mathcal{L}(\mathcal{V}(a), C) \geqslant n$.
16. If $\langle (a, b) : R, n \rangle \in \mathcal{A}$, then $\mathcal{E}(R, \langle \mathcal{V}(a), \mathcal{V}(b) \rangle) \geqslant n$.

**Proposition 4.1.** *$\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable iff there exists a fuzzy tableau for $\mathcal{K}$.*

**Proof.** For the *if* direction if $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{V})$ is a fuzzy tableau for $\mathcal{K}$, we can construct a fuzzy interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ that is a model of $\mathcal{A}$ and $\mathcal{T}$ as follows:

$$\begin{aligned}
\Delta^{\mathcal{I}} &= \mathbf{S}, \\
a^{\mathcal{I}} &= \mathcal{V}(a), \ \ a \text{ occurs in } \mathcal{A}, \\
\top^{\mathcal{I}}(s) &= \mathcal{L}(s, \top), \bot^{\mathcal{I}}(s) = \mathcal{L}(s, \bot) \quad \text{for all } s \in \mathbf{S}, \\
A^{\mathcal{I}}(s) &= \mathcal{L}(s, A) \quad \text{for all } s \in \mathbf{S}, \\
R^{\mathcal{I}}(s, t) &= \mathcal{E}(R, \langle s, t \rangle) \quad \text{for all } \langle s, t \rangle \in \mathbf{S} \times \mathbf{S}.
\end{aligned}$$

To prove that $\mathcal{I}$ is a model of $\mathcal{A}$ and $\mathcal{T}$, we can show by induction on the structure of concepts that $\mathcal{L}(s, C) \geqslant n$ implies $C^{\mathcal{I}}(s) \geqslant n$ for all $s \in \mathbf{S}$. As illustrative purpose, assume $C$ is $\exists R.D$. So, assume that $\mathcal{L}(s, \exists R.D) \geqslant n$. By definition, there is $t \in \mathbf{S}$ such that $\mathcal{E}(R, \langle s, t \rangle) \geqslant m_1$, $\mathcal{L}(t, D) \geqslant m_2$ and $n = m_1 \otimes m_2$, for some $m_1$ and $m_2$. By induction, $R^{\mathcal{I}}(s, t) \geqslant m_1$ and $D^{\mathcal{I}}(t) \geqslant m_2$ and, thus, $(\exists R.D)^{\mathcal{I}}(s) \geqslant n$. Together with properties 15–16, this implies that $\mathcal{I}$ is a model of $\mathcal{T}$, and that it satisfies each fuzzy assertion in $\mathcal{A}$.

For the converse, let $\mathcal{I}$ be a witnessed model of $\mathcal{K}$. Then a fuzzy tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{V})$ for $\mathcal{K}$ can be defined as follows:

$$\begin{aligned} \mathbf{S} &= \Delta^{\mathcal{I}}, \\ \mathcal{E}(R, \langle s, t\rangle) &= R^{\mathcal{I}}(s, t), \\ \mathcal{L}(s, C) &= C^{\mathcal{I}}(s), \\ \mathcal{V}(a) &= a^{\mathcal{I}}. \end{aligned}$$

It can be verified that $T$ is a fuzzy tableau for $\mathcal{K}$. As illustrative purpose, let us show that condition 12 of the fuzzy tableau is satisfied. So, assume that $\mathcal{L}(s, \exists R.C) \geqslant n$. The definition of $T$ implies that $(\exists R.C)^{\mathcal{I}}(s) \geqslant n$. Since $\mathcal{I}$ is a witnessed model, there exists $t \in \Delta^{\mathcal{I}}$ such that $(\exists R.C)^{\mathcal{I}}(s) = R^{\mathcal{I}}(s, t) \otimes C^{\mathcal{I}}(t)$, i.e., there are $m_1$ and $m_2$ such that $\mathcal{E}(R, \langle s, t\rangle) \geqslant m_1$, $\mathcal{L}(t, C) \geqslant m_2$ and $n = m_1 \otimes m_2$ and, thus, condition 12 of the fuzzy tableau is satisfied, which concludes. □

### 4.4. An algorithm for building a fuzzy tableau

Now, in order to decide the satisfiability of $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ (with $\mathcal{T}$ acyclic) a procedure that constructs a fuzzy tableau $T$ for $\mathcal{K}$ has to be determined. Like most of the tableaux algorithms (for instance [40]), our algorithm works on *completion-forests* since an ABox might contain several individuals with arbitrary roles connecting them. It is worth to note that, while reasoning algorithms within DLs usually transform concept expressions into a semantically equivalent *negation normal form* or NNF (which is obtained by pushing in the usual manner negation on front of concept names only), we cannot make this assumption now since in general $\neg\forall R.C \neq \exists R.(\neg C)$ and $\neg\exists R.C \neq \forall R.(\neg C)$.

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ be a fuzzy KB. A completion-forest $\mathcal{F}$ for $\mathcal{K}$ is a collection of trees whose distinguished roots are arbitrarily connected by edges. Each node $v$ is labelled with a set $\mathcal{L}(v)$ of expressions of the form $\langle C, l\rangle$, where $C \in sub(\mathcal{K})$, and $l$ is either a rational, a variable $x$, or a negated variable, i.e., of the form $1 - x$, where $x$ is a variable. The intuition here is that $v$ is an instance of $C$ to degree equal to or greater than the evaluation of $l$.

Each edge $\langle v, w\rangle$ is labelled with a set $\mathcal{L}(\langle v, w\rangle)$ of expressions of the form $\langle R, l\rangle$, where $R \in \mathbf{R}_{\mathcal{K}}$ are roles occurring in $\mathcal{K}$ (the intuition here is that $\langle v, w\rangle$ is an instance of $R$ to degree equal to or greater than the evaluation of $l$). If nodes $v$ and $w$ are connected by an edge $\langle v, w\rangle$ with $\langle R, l\rangle$ occurring in $\mathcal{L}(\langle v, w\rangle)$, then $w$ is called an *$R_l$-successor* of $v$.

The forest has associated a set $\mathcal{C}_{\mathcal{F}}$ of constraints of the form $l \leqslant l'$, $l = l'$, $x_i \in [0, 1]$, $y_i \in \{0, 1\}$, where $l, l'$ are arithmetic expressions, on the variables occurring the node labels and edge labels.

The algorithm initializes a forest $\mathcal{F}$ to contain (i) a root node $v_0^i$, for each individual $a_i$ occurring in $\mathcal{A}$, labelled with $\mathcal{L}(v_0^i)$ such that $\mathcal{L}(v_0^i)$ contains $\langle C_i, n\rangle$ for each fuzzy assertion $\langle a_i{:}\, C_i, n\rangle \in \mathcal{A}$, and (ii) an edge $\langle v_0^i, v_0^j\rangle$, for each fuzzy assertion $\langle (a_i, a_j){:}\, R_i, n\rangle \in \mathcal{A}$, labelled with $\mathcal{L}(\langle v_0^i, v_0^j\rangle)$ such that $\mathcal{L}(\langle v_0^i, v_0^j\rangle)$ contains $\langle R_i, n\rangle$. $\mathcal{F}$ is then expanded by repeatedly applying the completion rules described below. The completion-forest is complete when none of the completion rules are applicable. Then, the bMINLP problem on the set of constraints $\mathcal{C}_{\mathcal{F}}$ is solved.

As anticipated, we will use an extension to the fuzzy case of the *lazy expansion* technique in order to remove the axioms in $\mathcal{T}$. The basic idea is as follows (recall that there are only two types of fuzzy concept inclusions):

- Given $\langle A \sqsubseteq C, n\rangle$, add $C$ only to nodes with a label containing $A$.
- Given $\langle C \sqsubseteq A, 1\rangle$, add $\neg C$ only to nodes with a label containing $\neg A$.

We assume a fixed rule application strategy as e.g., the order of rules below, such that the rules for $(\exists)$ and $(\neg\forall)$ are applied as last. Also, all expressions in node labels are processed according to the order they are introduced into $\mathcal{F}$. Note that we do not need a notion of *blocking* as $\mathcal{T}$ is acyclic.

With $x_\tau$ we denote the variable associated to the *assertion* $\tau$ of the form $a{:}\, C$ or $(a, b){:}\, R$. $x_\tau$ will take the truth value associated to $\tau$.

Now we are ready to present the inference rules:

$(A)$ If (i) $\langle A, l\rangle \in \mathcal{L}(v)$, and (ii) $l \neq x_{v:A}$ then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{x_{v:A} \geqslant l\} \cup \{x_{v:A} \in [0, 1]\}$.
$(\bar{A})$ If $\langle \neg A, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{x_{v:A} \leqslant 1 - l\} \cup \{x_{v:A} \in [0, 1]\}$.
$(C)$ If (i) $\langle C, l\rangle \in \mathcal{L}(v)$, and (ii) $l \neq x_{v:C}$ then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{x_{v:C} \geqslant l\} \cup \{x_{v:C} \in [0, 1]\}$.

($R$) If $\langle R, l\rangle \in \mathcal{L}(\langle v, w\rangle)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{(v,w):R} \geqslant l\} \cup \{x_{(v,w):R} \in [0, 1]\}$.
($\bar{\top}$) If $\langle\neg\top, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{l = 0\}$.
($\bot$) If $\langle\bot, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{l = 0\}$.
($\neg\neg$) If $\langle\neg\neg C, l\rangle \in \mathcal{L}(v)$ then $\mathcal{L}(v) = \mathcal{L}(v) \cup \langle C, l\rangle$.
($\sqcap$) If $\langle C\sqcap D, l\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F}\cup\{x_{v:C}\otimes x_{v:D} \geqslant l\}\cup\{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
($\bar{\sqcap}$) If $\langle\neg(C \sqcap D), l\rangle \in \mathcal{L}(v)$ then append $\langle\neg C \sqcup \neg D, l\rangle$ to $\mathcal{L}(v)$.
($\sqcup$) If $\langle C\sqcup D, l\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F}\cup\{x_{v:C}\oplus x_{v:D} \geqslant l\}\cup\{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
($\bar{\sqcup}$) If $\langle\neg(C \sqcup D), l\rangle \in \mathcal{L}(v)$ then append $\langle\neg C \sqcap \neg D, l\rangle$ to $\mathcal{L}(v)$.
($\rightarrow$) If $\langle C \rightarrow D, l\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} \Rightarrow x_{v:D} \geqslant l\} \cup \{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
($\bar{\rightarrow}$) If $\langle\neg(C \rightarrow D), l\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} \Rightarrow x_{v:D} \leqslant 1 - l\} \cup \{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
($\forall$) If (i) $\langle\forall R.C, l_1\rangle \in \mathcal{L}(v)$, $\langle R, l_2\rangle \in \mathcal{L}(\langle v, w\rangle$, and (ii) the rule has not been already applied to this pair then (i) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{w:C} \geqslant l_1 \otimes l_2\} \cup \{x_{w:C} \in [0, 1]\}$.
($\bar{\exists}$) If (i) $\langle\neg\exists R.C, l_1\rangle \in \mathcal{L}(v)$, $\langle R, l_2\rangle \in \mathcal{L}(\langle v, w\rangle$, and (ii) the rule has not been already applied to this pair then (i) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{(v,w):R} \otimes x_{w:C} \leqslant 1 - l_1\} \cup \{x_{w:C} \in [0, 1]\}$.
($\sqsubseteq$) If (i) $\langle A \sqsubseteq C, n\rangle \in \mathcal{T}$, (ii) $\langle A, x_{v:A}\rangle \in \mathcal{L}(v)$, and (iii) $v$ is a node to which this rule has not yet been applied then (i) append $\langle C, x_{v:C}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} \geqslant x_{v:A} \otimes n\} \cup \{x_{v:C} \in [0, 1]\}$.
($\bar{\sqsubseteq}$) If (i) $\langle C \sqsubseteq A, 1\rangle \in \mathcal{T}$, (ii) $\langle\neg A, x_{v:\neg A}\rangle \in \mathcal{L}(v)$, and (iii) $v$ is a node to which this rule has not yet been applied then (i) append $\langle\neg C, x_{v:\neg C}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:\neg A} \geqslant x_{v:\neg C}\} \cup \{x_{v:\neg C} \in [0, 1]\}$.
($\exists$) If $\langle\exists R.C, l\rangle \in \mathcal{L}(v)$ then (i) create a new node $w$, and (ii) append $\langle R, x_{(v,w):R}\rangle$ to $\mathcal{L}(\langle v, w\rangle)$, and (iii) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (iv) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{w:C} \otimes x_{(v,w):R} \geqslant l\} \cup \{x_{w:C} \in [0, 1]\}$.
($\bar{\forall}$) If $\langle\neg\forall R.C, l\rangle \in \mathcal{L}(v)$ then (i) create a new node $w$, and (ii) append $\langle R, x_{(v,w):R}\rangle$ to $\mathcal{L}(\langle v, w\rangle)$, and (iii) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (iv) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{w:C} \Rightarrow x_{(v,w):R} \leqslant 1 - l\} \cup \{x_{w:C} \in [0, 1]\}$.

In order to write the fuzzy operators, we may need to create some new control variables. For example, under Łukasiewicz t-norm, $x_1 \otimes x_2 \geqslant l$ can be written as $\{l \leqslant y, x_1 + x_2 - 1 \leqslant l, x_1 + x_2 - y \geqslant l, y \in \{0, 1\}\}$. If $y = 0$, then $l = 0$ (it simulates the case where $x_1 + x_2 \leqslant 1$, and hence $x_1 \otimes x_2 = 0$), and if $y = 1$, then $l = x_1 + x_2 - 1$.

Now we will illustrate the full procedure with an example.

**Example 4.1.** Consider $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$, where $\mathcal{T} = \emptyset$ and $\mathcal{A} = \{\langle(a, b) : R, 0.7\rangle, \langle b: C, 0.8\rangle\}$. Let us show that $glb(\mathcal{K}, a: \exists R.C) = 0.8 \otimes 0.7$. To this end, we have to determine the minimal value for $x$ such that $\langle\mathcal{T}, \mathcal{A} \cup \{\langle a: \neg\exists R.C, 1 - x\rangle\}\rangle$ is satisfiable.

To start with, we construct a forest $\mathcal{F}$ with two root nodes $a$ and $b$ (one for each individual in $\mathcal{A}$). We process first $\langle(a, b): R, 0.7\rangle$, then $\langle b: C, 0.8\rangle$ and finally $\langle a: \neg\exists R.C, 1 - x\rangle$. Therefore, we set $\mathcal{L}(a) = [\langle\neg\exists R.C, 1 - x\rangle]$, $\mathcal{L}(\langle a, b\rangle) = [\langle R, 0.7\rangle]$, $\mathcal{L}(b) = [\langle C, 0.8\rangle]$ and $\mathcal{C_F} = \{x \in [0, 1]\}$.

We first process $\langle R, 0.7\rangle \in \mathcal{L}(\langle a, b\rangle)$, apply rule ($R$) and, thus, add $x_{(a,b):R} \geqslant 0.7$ and $x_{(a,b):R} \in [0, 1]$ to $\mathcal{C_F}$. Then we process $\langle C, 0.8\rangle \in \mathcal{L}(b)$, apply rule ($A$) and, thus, add $x_{b:C} \geqslant 0.8$ and $x_{b:C} \in [0, 1]$ to $\mathcal{C_F}$. We next process $\langle\neg\exists R.C, 1 - x\rangle \in \mathcal{L}(a)$, apply rule ($\bar{\exists}$) and, thus, we set $\mathcal{L}(b) = \langle C, x_{b:C}\rangle$ and add $\{x_{(a,b):R} \otimes x_{b:C} \leqslant 1 - (1 - x)\} = \{x_{(a,b):R} \otimes x_{b:C} \leqslant x\}$ and $x_{b:C} \in [0, 1]$ to $\mathcal{C_F}$. Next we process $\langle C, x_{b:C}\rangle \in \mathcal{L}(b)$, but since $l = x_{b:C}$, rule ($\bar{A}$) is not applied.

Now the forest $\mathcal{F}$ is complete as no more rules are applicable and we consider the set of inequations $\mathcal{C_F}$. It remains to solve the bMINLP problem on $\mathcal{C_F}$. Indeed, it holds that $glb(\mathcal{K}, a: D) = \min x.\mathcal{C_F}$. It can be verified that this value is $0.8 \otimes 0.7$.

Note that there is a significative difference with other similar algorithms for fuzzy DLs combining tableau algorithms with optimization problems [9,43,47]. In those algorithms, every time a concept $C$ appears in the list of expressions of a node $v$, a new variable $x$ is created. In this paper, we introduce a variable $x_{v:C}$ once, and reuse it the following times. This reduction in the number of generated variables is important because it makes the bMINLP problem easier to solve.

**Example 4.2.** Consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$, where $\mathcal{T} = \emptyset$ and $\mathcal{A} = \{\langle a : C \sqcap D, l_1\rangle, \langle a : C \sqcap E, l_2\rangle\}$. Each of the algorithms in [9,43,47] introduce the fuzzy assertions $\langle a : C, x_1\rangle$, $\langle a : D, x_2\rangle$, $\langle a : C, x_3\rangle$, $\langle a : E, x_4\rangle$, as well as some constraints. Hence, four new variables have been generated.

The new algorithm only introduces three variables, since it creates the fuzzy assertions $\langle a : C, x_{a:C}\rangle$, $\langle a : D, x_{a:D}\rangle$ and $\langle a : E, x_{a:E}\rangle$.

**Proposition 4.2** (*Termination*). *For each KB $\mathcal{K}$, the tableau algorithm terminates.*

**Proof.** Termination is a result of the properties of the expansion rules, as in the classical case [26]. More precisely we have the following observations: (i) The expansion rules never remove nodes from the tree or concepts from node labels or change the edge labels. (ii) Successors are only generated by the rules ($\exists$) and ($\neg\forall$). For any node and for each concept these rules are applied at-most once. (iii) Since nodes are labelled with non-empty sets of $sub(\mathcal{K})$, obviously there is a finite number of possible labelling for a pair of nodes. Consequently, any path of the tree will have a finite length. $\square$

**Proposition 4.3** (*Soundness*). *If the expansion rules can be applied to a KB $\mathcal{K}$ such that they yield a complete completion-forest $\mathcal{F}$ such that $\mathcal{C}_\mathcal{F}$ has a solution, then $\mathcal{K}$ has a fuzzy tableau for $\mathcal{K}$.*

**Proof.** Let $\mathcal{F}$ be a complete completion-forest constructed by the tableaux algorithm for $\mathcal{K}$. By hypothesis, $\mathcal{C}_\mathcal{F}$ has a solution. If $l$ is a linear expression in $\mathcal{C}_\mathcal{F}$, with $\bar{l}$ we denote the value of $l$ in this solution. If the variable $x$ does not occur in $\mathcal{C}_\mathcal{F}$ then $\bar{x} = 0$ is assumed. A fuzzy tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{V})$ can be defined as follows:

$$\begin{aligned}
\mathbf{S} &= \{v \mid v \text{ is a node in } \mathcal{F}\},\\
\mathcal{L}(v, \bot) &= 0 \quad \text{if } v \in \mathbf{S},\\
\mathcal{L}(v, \top) &= 1 \quad \text{if } v \in \mathbf{S},\\
\mathcal{L}(v, A) &= \bar{x}_{v:A} \quad \text{if } v \in \mathbf{S},\\
\mathcal{L}(v, C) &= \max(\bar{l} \mid \langle C, l\rangle \in \mathcal{L}(v)) \quad \text{if } v \in \mathbf{S},\\
\mathcal{E}(R, \langle v, w\rangle) &= \bar{x}_{(v,w):R} \quad \text{if } v, w \in \mathbf{S},\\
\mathcal{E}(R, \langle v, w\rangle) &= \bar{x}_{(v,w'):R} \quad \text{if } v, w \in \mathbf{S},\\
\mathcal{V}(a_i) &= v_0^i \quad \text{where } v_0^i \text{ is a root node.}
\end{aligned}$$

It can easily be shown that $T$ is a fuzzy tableau for $\mathcal{K}$. For illustrative purpose, let us show that condition 12 is satisfied. So assume that $\mathcal{L}(v, \exists R.C) \geqslant n$. The definition of $T$ implies that $\langle \exists R.C, l\rangle \in \mathcal{L}(v)$. Then the ($\exists$) rule ensures that there is a successor $w'$ of $v$ such that $\langle R, x_1\rangle \in \mathcal{L}(\langle v, w'\rangle)$ and $\langle C, x_2\rangle \in \mathcal{L}(w')$ with $\{x_1 \otimes x_2 \geqslant l, x_i \in [0, 1]\}$ in $\mathcal{C}_\mathcal{F}$. That is, we have that $\bar{x_1} \otimes \bar{x_2} \geqslant \bar{l}$. Now, $w' \in \mathbf{S}$, $\mathcal{E}(R, \langle v, w'\rangle) = \bar{x}_{(v,w'):R} \geqslant \bar{x_1}$ and $\mathcal{L}(w', C) \geqslant \bar{x_2}$ and, thus, condition 12 of the fuzzy tableau is satisfied. $\square$

**Proposition 4.4** (*Completeness*). *Consider a KB $\mathcal{K}$. If $\mathcal{K}$ has a fuzzy tableau, then the expansion rules can be applied in such a way that the tableaux algorithm yields a complete completion-forest for $\mathcal{K}$ such that $\mathcal{C}_\mathcal{F}$ has a solution.*

**Proof.** Let $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{V})$ be a fuzzy tableau for $\mathcal{K}$. Using $T$, we can trigger the application of the expansion rules such that they yield a completion-forest $\mathcal{F}$ that is complete. Using $\mathcal{L}$ and $\mathcal{E}$ we can find a solution to $\mathcal{C}_\mathcal{F}$. As illustrative purpose, assume that for some node $s$, $\langle \exists R.C, l\rangle \in \mathcal{L}(s)$ and the ($\exists$) rule can be applied to it. Then, as condition 12 of the fuzzy tableau holds, we apply the ($\exists$) rule by creating a new node $t$, and append $\langle R, x_1\rangle$ to $\mathcal{L}(\langle s, t\rangle)$ and $\langle C, x_2\rangle$ to $\mathcal{L}(t)$, and set $\mathcal{C}_\mathcal{F} = \mathcal{C}_\mathcal{F} \cup \{x_1 \otimes x_2 \geqslant l, x_i \in [0, 1]\}$, where $x_i$ are new variables. Now, from condition 12 we get immediately that for any $n \in [0, 1]$, by letting $l = n$ there are $m_1, m_2 \in [0, 1]$ such that for $x_1 = m_1$, $x_2 = m_2$ the new constraints are satisfied. $\square$

## 5. The case of the Product t-norm

### *5.1. Properties of the logic*

In this section we will focus on the particular case of the Product t-norm. Hence, the semantics is given by the fuzzy operators of the Product logic family extended with Łukasiewicz negation. Note that Product negation can still be defined since $\ominus_{\otimes} C = C \Rightarrow \bot$, so we actually allow two negations: Łukasiewicz and the negation of Product logic (Gödel negation).

We follow the inspiration of probabilistic theory, which combines this negation (in the probability of the negated event) and the product (in the probability of the conjunction of independent events) [29]. Note that the t-norm is subidempotent and the t-conorm is superidempotent, so in general $C \neq C \sqcap C$ and $C \neq C \sqcup C$. Furthermore, universal and existential quantifiers are not inter-definable.

Note also that extending Product logic with Łukasiewicz involutive negation leads to the logic $Ł\Pi$[8] [15] since:

- As shown in [12], Łukasiewicz t-conorm can be defined from Łukasiewicz negation $\ominus$, Product t-norm $\otimes$ and Product implication $\Rightarrow$ as

  $$\alpha \oplus_{Ł} \beta = \ominus\,(\ominus\alpha \;\otimes\; \ominus(\ominus\alpha \;\Rightarrow \beta))$$

- As it is well known, in Łukasiewicz logic the negation and the t-conorm can be used to define the remaining operators using duality (for the t-norm) and the definition of S-implication (for the implication).

Hájek showed for Łukasiewicz logic that if $\mathcal{K}$ has a model then it also has a witnessed model. Unfortunately, our logic does not share this witnessed model property, as the following example shows[9]:

**Example 5.1.** Consider a fuzzy KB with the following axioms:

$$a : \neg\forall R.A \;\geqslant\; 1, \tag{1}$$

$$a : \forall R.B \;\geqslant\; 1, \tag{2}$$

$$\langle B \sqcap B \;\sqsubseteq\; A, 1\rangle. \tag{3}$$

Let us show that there is an infinite model, but no witnessed model. Consider an interpretation $\mathcal{I}$ such that its domain is $\Delta^{\mathcal{I}} = \{a\} \cup \{b_n \mid n \in \mathbb{N}\}$ and

$$R^{\mathcal{I}}(a, b_n) = 1/n, \tag{4}$$

$$A^{\mathcal{I}}(b_n) = 1/n^2, \tag{5}$$

$$B^{\mathcal{I}}(b_n) = 1/n. \tag{6}$$

For all other cases the value 0 is assumed. Let us verify that $\mathcal{I}$ is indeed a model:

$\mathcal{I} \models a : \neg\forall R.A \geqslant 1$: $\mathcal{I} \models a : \neg\forall R.A \geqslant 1$ iff $(\forall R.A)^{\mathcal{I}}(a) = 0$ iff $0 = \inf_{b_n} R^{\mathcal{I}}(a, b_n) \Rightarrow A^{\mathcal{I}}(b_n)$ iff $0 = \inf_{b_n} 1/n \Rightarrow 1/n^2$. As $1/n > 1/n^2$ we have that $1/n \Rightarrow 1/n^2 = 1/n$ and, hence, $\inf_{b_n} 1/n \Rightarrow 1/n^2 = \inf_{b_n} 1/n = 0$. Therefore, $\mathcal{I} \models a : \neg\forall R.A \geqslant 1$.

$\mathcal{I} \models a : \forall R.B \geqslant 1$: $\mathcal{I} \models a : \forall R.B \geqslant 1$ iff $(\forall R.B)^{\mathcal{I}}(a) = 1$ iff $1 = \inf_{b_n} R^{\mathcal{I}}(a, b_n) \Rightarrow B^{\mathcal{I}}(b_n) = \inf_{b_n} 1/n \Rightarrow 1/n = \inf_{b_n} 1 = 1$. Therefore, $\mathcal{I} \models a : \forall R.B \geqslant 1$.

$\mathcal{I} \models \langle B \sqcap B \sqsubseteq A, 1\rangle$: $\mathcal{I} \models \langle B \sqcap B \sqsubseteq A, 1\rangle$ iff $1 = \inf_c B^{\mathcal{I}}(c) \cdot B^{\mathcal{I}}(c) \Rightarrow A^{\mathcal{I}}(c)$ iff for all objects $c$ of the domain $B^{\mathcal{I}}(c) \cdot B^{\mathcal{I}}(c) \leqslant A^{\mathcal{I}}(c)$. Now, for all individuals $c \neq b_n$ we have $B^{\mathcal{I}}(c) = 0$ so the condition holds trivially. For $c = b_n$ we have $B^{\mathcal{I}}(b_n) = 1/n$ and, thus, $B^{\mathcal{I}}(b_n) \cdot B^{\mathcal{I}}(b_n) = 1/n^2 \leqslant A^{\mathcal{I}}(b_n)$. Therefore, $\mathcal{I} \models \langle B \sqcap B \sqsubseteq A, 1\rangle$.

Therefore, $\mathcal{I}$ is a model of the KB. Now, let us show that the KB does not have a witnessed model. Assume to the contrary that there is such a model $\mathcal{I}$. Then:

1. From Eq. (1), $0 = \inf_b R^{\mathcal{I}}(a^{\mathcal{I}}, b) \Rightarrow A^{\mathcal{I}}(b)$. As $\mathcal{I}$ is witnessed, there is $b$ such that $0 = R^{\mathcal{I}}(a^{\mathcal{I}}, b) \Rightarrow A^{\mathcal{I}}(b)$, i.e., $A^{\mathcal{I}}(b) = 0$ and $R^{\mathcal{I}}(a^{\mathcal{I}}, b) > 0$.
2. From Eq. (3), for this $b$ we have that $B^{\mathcal{I}}(b) \cdot B^{\mathcal{I}}(b) \leqslant A^{\mathcal{I}}(b) = 0$. That is, $B^{\mathcal{I}}(b) = 0$.
3. Finally, from Eq. (2), we have that $1 = \inf_c R^{\mathcal{I}}(a^{\mathcal{I}}, c) \Rightarrow B^{\mathcal{I}}(c) \leqslant (R^{\mathcal{I}}(a^{\mathcal{I}}, b) \Rightarrow B^{\mathcal{I}}(b)) = 0$, as $R^{\mathcal{I}}(a^{\mathcal{I}}, b) > 0$ and $B^{\mathcal{I}}(b) = 0$, which is absurd.

Hence, $\mathcal{I}$ cannot be a witnessed model of the KB.

[8] $Ł\Pi$ is a very expressive logic combining Łukasiewicz and Product fuzzy operators and containing not only Łukasiewicz and Product logics, but also Gödel logic and many others.

[9] We are grateful to Lluís Godo, who significantly contributed to figure out the example.

We point out that in the paper [9] we claim that this logic has the witnessed model property, but this may not be the case as we have just seen. Hence, that paper also needs to restrict the reasoning tasks to witnessed models and to restrict the concept inclusion such as we do in this paper.

In this case, $glb(\mathcal{K}, C \sqsubseteq D)$ can be computed as the minimal value of $x$ such that $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \cup \{\langle a : C, x_1\rangle\} \cup \{\langle a : \neg D, 1 - x_2\rangle\}\rangle$ is satisfiable under the constraints $\{x \geqslant 1 - y, x_1 + (1 - y) \cdot x_2 \geqslant \cdot x_2, y \cdot x_2 \leqslant x \cdot x_1, x_1 \in [0, 1], x_2 \in [0, 1], y \in \{0, 1\}\}$, where $a$ is new abstract individual.

### 5.2. An algorithm for building a fuzzy tableau

In this case, the inference rules are the following:

- $(A)$ If (i) $\langle A, l\rangle \in \mathcal{L}(v)$, and (ii) $l \neq x_{v:A}$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:A} \geqslant l\} \cup \{x_{v:A} \in [0, 1]\}$.
- $(\bar{A})$ If $\langle \neg A, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:A} \leqslant 1 - l\} \cup \{x_{v:A} \in [0, 1]\}$.
- $(C)$ If (i) $\langle C, l\rangle \in \mathcal{L}(v)$, and (ii) $l \neq x_{v:C}$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} \geqslant l\} \cup \{x_{v:C} \in [0, 1]\}$.
- $(R)$ If $\langle R, l\rangle \in \mathcal{L}(\langle v, w\rangle)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{(v,w):R} \geqslant l\} \cup \{x_{(v,w):R} \in [0, 1]\}$.
- $(\bar{\top})$ If $\langle \neg\top, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{l = 0\}$.
- $(\bot)$ If $\langle \bot, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C_F} = \mathcal{C_F} \cup \{l = 0\}$.
- $(\neg\neg)$ If $\langle \neg\neg C, l\rangle \in \mathcal{L}(v)$ then $\mathcal{L}(v) = \mathcal{L}(v) \cup \langle C, l\rangle$.
- $(\sqcap)$ If (i) $\langle C \sqcap D, l\rangle \in \mathcal{L}(v)$, and (ii) not both $\langle C, x_{v:C}\rangle \in \mathcal{L}(v)$ and $\langle D, x_{v:D}\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} \cdot x_{v:D} \geqslant l\} \cup \{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
- $(\bar{\sqcap})$ If $\langle \neg(C \sqcap D), l\rangle \in \mathcal{L}(v)$ then append $\langle \neg C \sqcup \neg D, l\rangle$ to $\mathcal{L}(v)$.
- $(\sqcup)$ If (i) $\langle C \sqcup D, l\rangle \in \mathcal{L}(v)$, and (ii) not both $\langle C, x_{v:C}\rangle \in \mathcal{L}(v)$ and $\langle D, x_{v:D}\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} + x_{v:D} - x_{v:C} \cdot x_{v:D} \geqslant l\} \cup \{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
- $(\bar{\sqcup})$ If $\langle \neg(C \sqcup D), l\rangle \in \mathcal{L}(v)$ then append $\langle \neg C \sqcap \neg D, l\rangle$ to $\mathcal{L}(v)$.
- $(\rightarrow)$ If $\langle C \rightarrow D, l\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:D} \geqslant x_{v:C} \cdot l\} \cup \{x_{v:C} \in [0, 1]\} \cup \{x_{v:D} \in [0, 1]\}$.
- $(\bar{\rightarrow})$ If $\langle \neg(C \rightarrow D), l\rangle \in \mathcal{L}(v)$ then (i) append $\langle C, x_{v:C}\rangle$ and $\langle D, x_{v:D}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} + (1 - y) \cdot x_{v:D} \geqslant x_{v:D}, l \leqslant y, y \cdot x_{v:D} \leqslant x_{v:C} - l \cdot x_{v:C}\} \cup \{x_{v:D} \in [0, 1]\} \cup \{y \in \{0, 1\}\}$, where $y$ is a new variable.
- $(\forall)$ If (i) $\langle \forall R.C, l_1\rangle \in \mathcal{L}(v)$, $\langle R, l_2\rangle \in \mathcal{L}(\langle v, w\rangle$, and (ii) the rule has not been already applied to this pair then (i) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{w:C} \geqslant l_1 \cdot l_2\} \cup \{x_{w:C} \in [0, 1]\}$.
- $(\bar{\exists})$ If (i) $\langle \neg\exists R.C, l_1\rangle \in \mathcal{L}(v)$, $\langle R, l_2\rangle \in \mathcal{L}(\langle v, w\rangle$, and (ii) the rule has not been already applied to this pair then (i) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{(v,w):R} \cdot x_{w:C} \leqslant 1 - l_1\} \cup \{x_{w:C} \in [0, 1]\}$.
- $(\sqsubseteq)$ If (i) $\langle A \sqsubseteq C, n\rangle \in \mathcal{T}$, (ii) $\langle A, x_{v:A}\rangle \in \mathcal{L}(v)$, and (iii) $v$ is a node to which this rule has not yet been applied then (i) append $\langle C, x_{v:C}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:C} \geqslant x_{v:A} \cdot n\} \cup \{x_{v:C} \in [0, 1]\}$.
- $(\bar{\sqsubseteq})$ If (i) $\langle C \sqsubseteq A, 1\rangle \in \mathcal{T}$, (ii) $\langle \neg A, x_{v:\neg A}\rangle \in \mathcal{L}(v)$, and (iii) $v$ is a node to which this rule has not yet been applied then (i) append $\langle \neg C, x_{v:\neg C}\rangle$ to $\mathcal{L}(v)$, and (ii) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{v:\neg A} \geqslant x_{v:\neg C}\} \cup \{x_{v:\neg C} \in [0, 1]\}$.
- $(\exists)$ If $\langle \exists R.C, l\rangle \in \mathcal{L}(v)$ then (i) create a new node $w$, and (ii) append $\langle R, x_{(v,w):R}\rangle$ to $\mathcal{L}(\langle v, w\rangle)$, and (iii) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (iv) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{w:C} \cdot x_{(v,w):R} \geqslant l\} \cup \{x_{w:C} \in [0, 1]\}$.
- $(\bar{\forall})$ If $\langle \neg\forall R.C, l\rangle \in \mathcal{L}(v)$ then (i) create a new node $w$, and (ii) append $\langle R, x_{(v,w):R}\rangle$ to $\mathcal{L}(\langle v, w\rangle)$, and (iii) append $\langle C, x_{w:C}\rangle$ to $\mathcal{L}(w)$, and (iv) $\mathcal{C_F} = \mathcal{C_F} \cup \{x_{(v,w):R} + (1 - y) \cdot x_{w:C} \geqslant x_{w:C}, l \leqslant y, y \cdot x_{w:C} \leqslant x_{(v,w):R} - l \cdot x_{(v,w):R}\} \cup \{x_{w:C} \in [0, 1]\} \cup \{y \in \{0, 1\}\}$, where $y$ is a new variable.

**Example 5.2.** Let us comment the $(\bar{\forall})$-rule. If $\langle \neg\forall R.C, l\rangle \in \mathcal{L}(v)$ then $\inf_w R^{\mathcal{I}}(v, w) \Rightarrow C^{\mathcal{I}}(w) \leqslant 1 - l$. For a new abstract individual $w$, there are two possibilities to satisfy this restriction:

- $l = 0$, since obviously $\langle \neg\forall R.C, 0\rangle \in \mathcal{L}(v)$ holds;
- $R^{\mathcal{I}}(v, w) \geqslant C^{\mathcal{I}}(w)$ and $C^{\mathcal{I}}(w) \leqslant R^{\mathcal{I}}(v, w) - R^{\mathcal{I}}(v, w) \cdot l$.

The binary variable $y$ simulates the two branches:

- if $y = 0$ then $l \leqslant 0$;
- if $y = 1$ then $x_{(v,w):R} \geqslant x_{w:C}$ (so $R^{\mathcal{I}}(a, b) \geqslant C^{\mathcal{I}}(b)$) and $x_{w:C} \leqslant x_{(v,w):R} - x_{(v,w):R} \cdot l$ (so $C^{\mathcal{I}}(w) \leqslant R^{\mathcal{I}}(v, w) - R^{\mathcal{I}}(v, w) \cdot l$). $\square$

Notice that every constructor of the logic is bMICQP representable i.e., it generates a set of bMICQP representable constraints. Hence, we end up solving a bMICQP problem, instead of a bMINLP problem as it occurs in the general case.

**Example 5.3.** Consider again the fuzzy KB in Example 4.1 $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T} = \emptyset$ and $\mathcal{A} = \{\langle (a, b) : R, 0.7\rangle, \langle b\colon C, 0.8\rangle\}$. Let us show that $glb(\mathcal{K}, a\colon \exists R.C) = 0.56$. To this end, we have to determine the minimal value for $x$ such that $\langle \mathcal{T}, \mathcal{A} \cup \{\langle a\colon \neg\exists R.C, 1 - x\rangle\}\rangle$ is satisfiable.

To start with, we construct a forest $\mathcal{F}$ with two root nodes $a$ and $b$ (one for each individual in $\mathcal{A}$). We process first $\langle (a, b)\colon R, 0.7\rangle$, then $\langle b\colon C, 0.8\rangle$ and finally $\langle a\colon \neg\exists R.C, 1 - x\rangle$. Therefore, we set $\mathcal{L}(a) = [\langle \neg\exists R.C, 1 - x\rangle]$, $\mathcal{L}(\langle a, b\rangle) = [\langle R, 0.7\rangle]$, $\mathcal{L}(b) = [\langle C, 0.8\rangle]$ and $\mathcal{C}_{\mathcal{F}} = \{x \in [0, 1]\}$.

We first process $\langle R, 0.7\rangle \in \mathcal{L}(\langle a, b\rangle)$, apply rule $(R)$ and, thus, add $x_{(a,b):R} \geqslant 0.7$ and $x_{(a,b):R} \in [0, 1]$ to $\mathcal{C}_{\mathcal{F}}$. Then we process $\langle C, 0.8\rangle \in \mathcal{L}(b)$, apply rule $(A)$ and, thus, add $x_{b:C} \geqslant 0.8$ and $x_{b:C} \in [0, 1]$ to $\mathcal{C}_{\mathcal{F}}$. We next process $\langle \neg\exists R.C, 1-x\rangle \in \mathcal{L}(a)$, apply rule $(\bar{\exists})$ and, thus, we set $\mathcal{L}(b) = \langle C, x_{b:C}\rangle$ and add $\{x_{(a,b):R} \cdot x_{b:C} \leqslant x\}$ and $\{x_{b:C} \in [0, 1]\}$ to $\mathcal{C}_{\mathcal{F}}$. Next we process $\langle \neg C, 1 - x_1\rangle \in \mathcal{L}(b)$, apply rule $(\bar{A})$ and, thus, add $x_{b:C} \leqslant x_1$ to $\mathcal{C}_{\mathcal{F}}$.

Now the forest $\mathcal{F}$ is complete as no more rules are applicable and we consider the set of inequations $\mathcal{C}_{\mathcal{F}}$. It remains to solve the bMIQCP problem on $\mathcal{C}_{\mathcal{F}}$. Indeed, it holds that $glb(\mathcal{K}, a\colon D) = \min x.\mathcal{C}_{\mathcal{F}}$. It can be verified that this value is 0.56.

## 6. Concrete features

The aim of this section is to add the possibility to deal with reals, integers and strings within the language. Essentially, we allow to specify some constraints on the allowed values of an attribute, i.e., functional role such as *hasAge* and *hasName*. For instance, we may want to express the concept of person who are not old enough to vote as those having an age of less than 18 years, i.e.,

$$Person \sqcap (< hasAge\ 18)$$

or the concept of people whose first name is "umberto":

$$Person \sqcap (= hasName\ \text{"umberto"})$$

It is interesting to note that this approach is independent of the family of operators, and hence can be combined with the Product logic described in Section 5, but also with Łukasiewicz and Zadeh families [43,47].

We start this section by proposing a fuzzy extension of $\mathcal{ALCF}(\mathbf{D})$ including reals, integer and strings. Once the syntax and the semantics are defined, we address the reasoning algorithm.

### 6.1. $\mathcal{ALCF}(D)$ Syntax

$\mathcal{ALCF}(\mathbf{D})$ extends $\mathcal{ALC}$ with functional roles (also called attributes or features) and concrete domains [31] allowing to deal with datatypes such as strings and integers. We will also allow modifiers. For the sake of brevity, in this section we will focus on the differences with respect to fuzzy $\mathcal{ALC}$ as defined in Section 3.

A *fuzzy data type theory* $\mathbf{D} = \langle \Delta_{\mathbf{D}}, \cdot_{\mathbf{D}}\rangle$ is such that $\cdot_{\mathbf{D}}$ assigns to every $n$-ary data type predicate an $n$-ary fuzzy relation over $\Delta_{\mathbf{D}}$. For instance, as for $\mathcal{ALCF}(\mathbf{D})$, the predicate $\leqslant_{18}$ may be a unary crisp predicate over the natural numbers denoting the set of integers smaller or equal to 18. On the other hand, concerning non-crisp fuzzy domain predicates, we recall that in fuzzy set theory and practice, there are many functions for specifying fuzzy set membership degrees. However, the triangular, the trapezoidal, the $L$-function (left-shoulder function), and the $R$-function (right-shoulder function) are simple, but most frequently used to specify membership degrees. The functions are defined over the set of non-negative rationals $\mathbb{Q}^+ \cup \{0\}$ (see Fig. 1). Using these functions, we may then define, for instance, *Young*: *Natural* $\rightarrow [0, 1]$ to be a fuzzy concrete predicate over the natural numbers denoting the degree of youngness of a person's age. The concrete fuzzy predicate *Young* may be defined as $Young(x) = L(x; 10, 30)$.

**Example 6.1.** Assume, that a car seller sells an Audi TT for \$31 500, as from the catalog price. A buyer is looking for a sports car, but wants to pay no more than around \$30 000. In classical DLs no agreement can be found. The

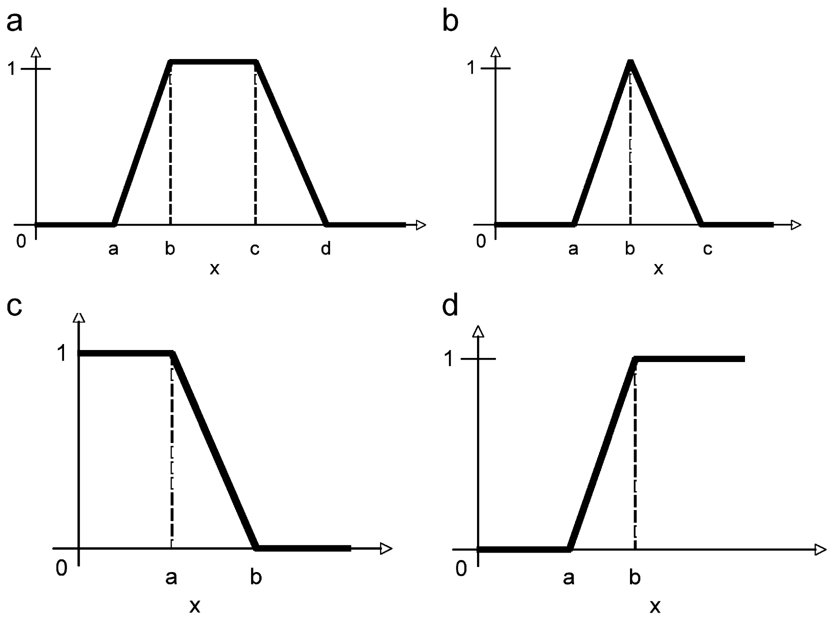


Fig. 1. (a) Trapezoidal function; (b) triangular function; (c) $L$-function; (d) $R$-function.

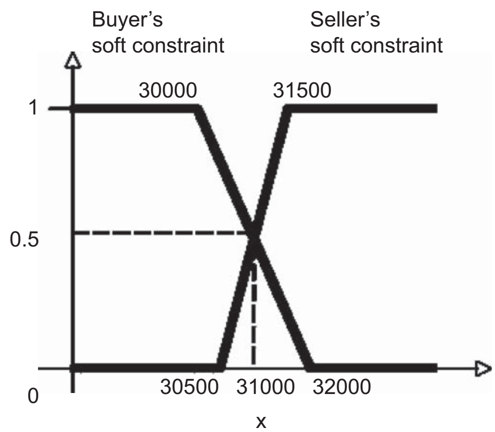


Fig. 2. The soft price constraints.

problem relies on the crisp condition on the seller's and the buyer's price. A more fine grained approach would be (and usually happens in matchmaking) to consider prices as concrete fuzzy sets instead. For instance, the seller may consider optimal to sell above \$31 500, but can go down to \$30 500. The buyer prefers to spend less than \$30 000, but can go up to \$32 000. We may represent these statements using two axioms (see Fig. 2):

$$\begin{aligned} AudiTT &= SportsCar \sqcap \exists hasPrice.R(x; 30\,500, 31\,500), \\ Query &= SportsCar \sqcap \exists hasPrice.L(x; 30\,000, 32\,000), \end{aligned}$$

where *hasPrice* is a concrete feature (a car has only one price, which is a number). Then we may find out that the highest degree to which the concept $AudiTT \sqcap Query$ is satisfiable is 0.5 (the *possibility* that the Audi TT and the query matches is 0.5). That is, $glb(\mathcal{K}, C) = 0.5$ and corresponds to the point where both requests intersects (i.e., the car may be sold at \$31 000).

We also allow fuzzy modifiers in fuzzy $\mathcal{ALCF}(\mathbf{D})$, like *very*, *more_or _less* and *slightly*, apply to fuzzy sets to change their membership function. Formally, a *modifier* is a function $f_m: [0,1] \to [0,1]$. For instance, we may define $very(x) = x^2$ and $slightly(x) = \sqrt{x}$. Modifiers have been considered, for instance, in [22,48].

Now, let $\mathbf{A}, \mathbf{R}_A, \mathbf{R}_C, \mathbf{I}, \mathbf{I}_c$ and $\mathbf{M}$ be non-empty enumerable and pair-wise disjoint sets of *concept names* (denoted $A$), *abstract role names* (denoted $R$), i.e., binary predicates *concrete role names* (denoted $T$), *abstract individual names* (denoted $a$), *concrete individual names* (denoted $c$) and *modifiers* (denoted $m$). $\mathbf{R}_A$ also contains a non-empty subset $\mathbf{F}_a$ of *abstract feature names* (denoted $r$), while $\mathbf{R}_C$ contains a non-empty subset $\mathbf{F}_c$ of *concrete feature names* (denoted $t$). Features are functional roles.

Firstly, concrete features should be defined using the syntax (cf *name type* [*range*]), where:

- *type* of the related datatypes can be *real*, *integer* or *string*.
- *range* (only for reals and integers) is optional and has the form $k_1$ $k_2$, where $k_1, k_2$ is a real or an integer (depending on the type);
- *name* is the name of the concrete feature.

For instance, some valid declarations are (cf hasAge integer 0 150) and (cf hasName string).

Now, $\mathcal{ALCF}(\mathbf{D})$ concepts can be built according to the following syntax rule (for ease, we assume that fuzzy domain predicates are unary):

$$\begin{aligned} C &:= \top \mid \bot \mid A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid C_1 \to C_2 \mid \neg C \mid \forall R.C \mid \exists R.C \mid \\ &\quad\ m(C) \mid \forall T.D \mid \exists T.D \mid CDC, \\ D &:= d \mid \neg d, \\ CDC &:= (\geqslant t\ n) \mid (\leqslant t\ n) \mid (= t\ n), \end{aligned}$$

where $d$ is a unary fuzzy domain predicate, $n$ is a value for feature $t$ of the appropriate type (currently, a real, an integer or a string). The syntax of concepts $(\geqslant t\ n)$ and $(\leqslant t\ n)$ should not be confused with the similar syntax of unqualified cardinality restrictions (in more expressive DLs).

### 6.2. $\mathcal{ALCF}(\mathbf{D})$ semantics

In $\mathcal{ALCF}(\mathbf{D})$ a *fuzzy interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ relative to a fuzzy data type theory $\mathbf{D} = \langle \Delta_{\mathbf{D}}, \cdot_{\mathbf{D}} \rangle$ consists of a non-empty set $\Delta^{\mathcal{I}}$ (the *domain*), disjoint from $\Delta_{\mathbf{D}}$, and of a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$ that coincides with $\cdot_{\mathbf{D}}$ on every data value, data type, and fuzzy data type predicate, and it assigns:

1. to each abstract concept $C$ a function $C^{\mathcal{I}}: \Delta^{\mathcal{I}} \to [0,1]$;
2. to each abstract role $R$ a function $R^{\mathcal{I}}: \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1]$;
3. to each abstract feature $r$ a partial function $r^{\mathcal{I}}: \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to \{0,1\}$ such that for all $u \in \Delta^{\mathcal{I}}$ there is an unique $w \in \Delta^{\mathcal{I}}$ on which $r^{\mathcal{I}}(u, w)$ is defined;
4. to each concrete role $T$ a function $R^{\mathcal{I}}: \Delta^{\mathcal{I}} \times \Delta_{\mathbf{D}} \to [0,1]$;
5. to each concrete feature $t$ a partial function $t^{\mathcal{I}}: \Delta^{\mathcal{I}} \times \Delta_{\mathbf{D}} \to \{0,1\}$ such that for all $u \in \Delta^{\mathcal{I}}$ there is an unique $o \in \Delta_{\mathbf{D}}$ on which $t^{\mathcal{I}}(u, o)$ is defined;
6. to each modifier $m$ the modifier function $f_m: [0,1] \to [0,1]$;
7. to each abstract individual $a$ an element in $\Delta^{\mathcal{I}}$;
8. to each concrete individual $c$ an element in $\Delta_{\mathbf{D}}$.
9. to each $n$-ary concrete predicate $d$ the interpretation $d_{\mathbf{D}} \subseteq \Delta^n_{\mathbf{D}}$.

Notice that we force features to be crisp. In our opinion, the notion of functionality induces a crisp interpretation over a concrete feature and hence saying that e.g., the degree of truth of $hasAge(x, 18)$ is 0.5 is rather unrealistic. [10]

The fuzzy interpretation function is extended as shown in Table 3, where $x, y \in \Delta^{\mathcal{I}}$, $v \in \Delta_{\mathbf{D}}$ and $=$ denotes the crisp equality. Notice that CDC are crisp concepts and, accordingly, $\otimes, \oplus$ and $\ominus$ denote the classical intersection, union and complement, respectively. Furthermore, it is easy to see that $(= t\ n) = (\leqslant t\ n) \otimes (\geqslant t\ n)$.

[10] Personal communication with Lluís.

Table 3
Fuzzy DL semantics for $\mathcal{ALCF}(\mathbf{D})$.

$$
\begin{aligned}
\bot^{\mathcal{I}}(x) &= 0\\
\top^{\mathcal{I}}(x) &= 1\\
(C_1 \sqcap C_2)^{\mathcal{I}}(x) &= C_1{}^{\mathcal{I}}(x) \otimes C_2{}^{\mathcal{I}}(x)\\
(C_1 \sqcup C_2)^{\mathcal{I}}(x) &= C_1{}^{\mathcal{I}}(x) \oplus C_2{}^{\mathcal{I}}(x)\\
(C_1 \rightarrow C_2)^{\mathcal{I}}(x) &= C_1{}^{\mathcal{I}}(x) \Rightarrow C_2{}^{\mathcal{I}}(x)\\
(\neg C)^{\mathcal{I}}(x) &= \ominus\, C^{\mathcal{I}}(x)\\
(\forall R.C)^{\mathcal{I}}(x) &= \inf_{y \in \Delta^{\mathcal{I}}}\ R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)\\
(\exists R.C)^{\mathcal{I}}(x) &= \sup_{y \in \Delta^{\mathcal{I}}}\ R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)\\
(m(C))^{\mathcal{I}}(x) &= f_m(C^{\mathcal{I}}(x))\\
(\forall T.D)^{\mathcal{I}}(x) &= \inf_{y \in \Delta_{\mathbf{D}}} (T^{\mathcal{I}}(x,y)) \Rightarrow D^{\mathcal{I}}(y)\\
(\exists T.D)^{\mathcal{I}}(x) &= \sup_{y \in \Delta_{\mathbf{D}}} (T^{\mathcal{I}}(x,y)) \otimes D^{\mathcal{I}}(y)\\
(\geqslant t\ n)^{\mathcal{I}}(a) &= \sup_{b \in \Delta_D} [t^{\mathcal{I}}(a,b) \otimes (b \geqslant n)]\\
\neg(\geqslant t\ n)^{\mathcal{I}}(a) &= 1 - (\geqslant t\ n)^{\mathcal{I}}(a) = \inf_{b \in \Delta_D} [\ominus t^{\mathcal{I}}(a,b) \oplus (b < n)]\\
(\leqslant t\ n)^{\mathcal{I}}(a) &= \sup_{b \in \Delta_D} [t^{\mathcal{I}}(a,b) \otimes (b \leqslant n)]\\
\neg(\leqslant t\ n)^{\mathcal{I}}(a) &= 1 - (\leqslant t\ n)^{\mathcal{I}}(a) = \inf_{b \in \Delta_D} [\ominus t^{\mathcal{I}}(a,b) \oplus (b > n)]\\
(= t\ n)^{\mathcal{I}}(a) &= \sup_{b \in \Delta_D} [t^{\mathcal{I}}(a,b) \otimes (b = n)]\\
\neg(= t\ n)^{\mathcal{I}}(a) &= 1 - (= t\ n)^{\mathcal{I}}(a) = \inf_{b \in \Delta_D} [\ominus t^{\mathcal{I}}(a,b) \oplus (b \neq n)]
\end{aligned}
$$

### 6.3. Reasoning

Reasoning within $\mathcal{ALCF}(\mathbf{D})$ relies on the construction of a fuzzy tableau. In this section we concentrate on the differences with respect to the algorithm described in Section 5.

Firstly, we need a technical definition involving feature roles (see [31]). Let $\mathcal{F}$ be a forest, $r$ an abstract or concrete feature such that we have two edges $\langle v, w_1 \rangle$ and $\langle v, w_1 \rangle$ such that $\langle r, l_1 \rangle$ and $\langle r, l_2 \rangle$ occur in $\mathcal{L}(\langle v, w_1 \rangle)$ and $\mathcal{L}(\langle v, w_2 \rangle)$, respectively (informally, $\mathcal{F}$ contains $\langle (v, w_1){:}\, r, l_1 \rangle$ and $\langle (v, w_2){:}\, r, l_2 \rangle$). Then we call such a pair a *fork*. As $r$ is a function, such a fork means that $w_1$ and $w_2$ have to be interpreted as the same individual. Such a fork can be deleted by adding both $\mathcal{L}(\langle v, w_2 \rangle)$ to $\mathcal{L}(\langle v, w_1 \rangle)$ and $\mathcal{L}(w_2)$ to $\mathcal{L}(w_1)$, and then deleting node $w_2$. A similar argument applies to concrete feature roles. At the beginning, we remove the forks from the initial forest. We assume that forks are eliminated as soon as they appear (as part of a rule application) with the proviso that newly generated nodes are replaced by older ones and not vice versa.

Then, we have to extend the notion of fuzzy tableau with the following additional conditions (associated to the new constructs):

- If $\mathcal{L}(s, m(C)) \geqslant n$, then $\mathcal{L}(s, C) \geqslant m$ and $n = f_m(m)$ for some $m$.
- If $\mathcal{L}(s, \forall T.D) \geqslant n$, then $D^{\mathcal{I}}(c) \geqslant \mathcal{E}(R, \langle s, c \rangle) \otimes n$, for all $c \in \mathbf{S}$.
- If $\mathcal{L}(s, \neg\forall T.D) \geqslant n$, then there exists $c \in \mathbf{S}$ such that $\mathcal{E}(R, \langle s, c \rangle) \Rightarrow D^{\mathcal{I}}(c) \leqslant 1 - n$.
- If $\mathcal{L}(s, \exists T.D) \geqslant n$, then there exists $c \in \mathbf{S}$ such that $\mathcal{E}(R, \langle s, c \rangle) \geqslant m_1$, $D^{\mathcal{I}}(c) \geqslant m_2$ and $n = m_1 \otimes m_2$, for some $m_1$ and $m_2$.
- If $\mathcal{L}(s, \neg\exists T.D) \geqslant n$, then $\mathcal{E}(R, \langle s, c \rangle) \otimes D^{\mathcal{I}}(c) \leqslant 1 - n$, for all $c \in \mathbf{S}$.
- $\mathcal{L}(s, (\geqslant t\ n)) \in \{0, 1\}$.
- If $\mathcal{L}(s, (\geqslant t\ n)) = 1$, then there exists $c \in \mathbf{S}$ such that $\mathcal{E}(t, \langle s, c \rangle) = 1$ and $c \geqslant n$.
- If $\mathcal{L}(s, (\neg\geqslant t\ n)) = 1$, then $\mathcal{E}(R, \langle c, t \rangle) = 0$ or $\mathcal{L}(t, C) < n$, for all $c \in \mathbf{S}$.
- $\mathcal{L}(s, (\leqslant t\ n)) \in \{0, 1\}$.
- If $\mathcal{L}(s, (\leqslant t\ n)) = 1$, then there exists $c \in \mathbf{S}$ such that $\mathcal{E}(t, \langle s, c \rangle) = 1$ and $c \leqslant n$.
- If $\mathcal{L}(s, (\neg\geqslant t\ n)) = 1$, then $\mathcal{E}(R, \langle c, t \rangle) = 0$ or $\mathcal{L}(t, C) > n$, for all $c \in \mathbf{S}$.
- $\mathcal{L}(s, (= t\ n)) \in \{0, 1\}$.
- If $\mathcal{L}(s, (= t\ n)) = 1$, then there exists $c \in \mathbf{S}$ such that $\mathcal{E}(t, \langle s, c \rangle) = 1$ and $c = n$.
- If $\mathcal{L}(s, (\neg\geqslant t\ n)) = 1$, then $\mathcal{E}(R, \langle c, t \rangle) = 0$ or $\mathcal{L}(t, C) > n$ or $\mathcal{L}(t, C) < n$, for all $c \in \mathbf{S}$.

Now, we introduce the inference rules, where $x_\tau$ will take the truth value associated to $\tau$, while with $x_c$ we denote the variable associated to the concrete individual $c$. Note that these rules generate a set of constraints which is a bMILP problem. Hence, if we combine concrete features with fuzzy operators from Łukasiewicz and Zadeh families, we may rely on a bMILP solver. However, if we want to use it with Product logic, we still need a bMIQCP solver.

We will start by showing how to deal with modifiers and fuzzy concrete roles, then we consider integers and reals and finally we extend the work to strings.

### 6.3.1. Inference rules for modifiers and fuzzy membership functions

In order to represent the membership functions of modifiers and concrete fuzzy predicates, we use a linear function (or a combination of linear functions) $L$. In order to represent it as a bMILP, we have to define the graph $g(L) = \{\langle x1, x2\rangle : L(x1) \geqslant x2\}$ as the solutions of a bMILP. Similarly, as we may have negation in front of modifiers and fuzzy domain predicates, we also need to define $\bar{g}(L) = \{\langle x1, x2\rangle : L(x1) \leqslant x2\}$.

The inference rules are the following:

- $(T)$ If $\langle T, l\rangle \in \mathcal{L}(\langle v, w\rangle)$ the case is similar as in rule $(R)$, but considering concrete roles.
- $(r)$ If $\langle r, l\rangle \in \mathcal{L}(\langle v, w\rangle)$ then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{x_{(v,w):r} \geqslant l\} \cup \{x_{(v,w):r} \in \{0, 1\}\}$. The case for concrete features $t$ is similar.
- $(m)$ If $\langle m(C), l\rangle \in \mathcal{L}(v)$ then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{\gamma(v : C, l)\}$, where the set $\gamma(v : C, l)$ is obtained from the bMILP representation of $g(m)$ as follows: replace in $g(m)$ all occurrences of $x_2$ with $l$. Then resolve for $x_1$ and replace all occurrences of the form $x_1 \geqslant l'$ with $\langle v : C, l'\rangle$, while replace all occurrences the form $x_1 \leqslant l'$ with $\langle v : (\neg C), 1-l'\rangle$.
- $(\bar{m})$ The case $\langle \neg m(C), l\rangle \in \mathcal{L}(v)$ is similar as in rule $(m)$, but using the bMILP representation of $\bar{g}(m)$ in place of $g(m)$.
- $(d)$ If $\langle d, l\rangle \in \mathcal{L}(v)$ then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{\gamma(v : d, l)\}$, where the set $\gamma(v : d, l)$ is obtained from the bMILP representation of $g(d)$ by replacing all occurrences of $x_2$ with $l$ and $x_1$ with $x_v$.
- $(\bar{d})$ The case $\langle \neg d, l\rangle \in \mathcal{L}(v)$ is similar as in rule $(d)$, where we use the bMILP representation of $\bar{g}(d)$ in place of $g(d)$.
- $(\forall_{\mathbf{D}})$ If $\langle \forall T.D, l\rangle \in \mathcal{L}(v)$, the case is similar as in rule $(\forall)$.
- $(\bar{\exists}_{\mathbf{D}})$ If $\langle \neg\exists T.D, l\rangle \in \mathcal{L}(v)$, the case is similar as in rule $(\bar{\exists})$.
- $(\exists_{\mathbf{D}})$ If $\langle \exists T.D, l\rangle \in \mathcal{L}(v)$, the case is similar as in rule $(\exists)$.
- $(\bar{\forall}_{\mathbf{D}})$ If $\langle \neg\forall T.D, l\rangle \in \mathcal{L}(v)$, the case is similar as in rule $(\bar{\forall})$.

Note that rules for modifiers and concrete roles are similar to [43], but in rule $(m)$ we write $v : (\neg C)$ instead of $v : (\mathrm{nnf}(\neg C))$. The reason is that we want to express this rules independently from the family of fuzzy operators, and in general (e.g., in the Product logic in Section 5) we cannot suppose that concepts are in NNF.

For the sake of a concrete illustration of the meaning of $g$ and $\bar{g}$, consider a left-shoulder function with $k_1 = 0$ and $k_2 = b$. Then,

$$\bar{g}(m) = \{\langle x1, x2\rangle : x_1 \geqslant a(1-y) + by, x_2 \geqslant 1-y, x_1 \geqslant ay, x_1 \leqslant b, x_1 + (b-a)x_2 \geqslant by\}.$$

It can be verified that the control variable $y$ simulates two branches: if $y = 0$ then $x_2 = 1$, whereas if $y = 1$ then $x_2 \geqslant (b - x_1)/(b - a)$.

### 6.3.2. Inference rules for integers and reals

We will start by showing the inference rules, and then we will illustrate them with a pair of examples.

- $(\geqslant_{\Delta_D})$ If $\langle (\geqslant t\ n), \alpha_1\rangle \in \mathcal{L}(v)$ then: (i) if $\langle t(v, c), \beta\rangle \notin \mathcal{L}(v, c)$ then add $\langle t(v, c), \alpha_1\rangle$ to $\mathcal{L}(v, c)$, (ii) for some $c$ such that $\langle t(v, c), \beta\rangle \in \mathcal{L}(v, c)$, $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{\alpha_1 + y \geqslant \varepsilon\} \cup \{\alpha_1 + y \leqslant 1\} \cup \{x_{t(v,c)} \geqslant 1 - y\} \cup \{x_c \geqslant n - (n - k_1)y\} \cup \{y \in \{0, 1\}\}$, where $y$ is a new variable.
- $(\neg\geqslant_{\Delta_D})$ If (i) $\langle \neg(\geqslant t\ n), \alpha_1\rangle \in \mathcal{L}(v)$ and (ii) $\langle t(v, c), \alpha_2\rangle \in \mathcal{L}(v, c)$, then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{y_1 \leqslant y_2\} \cup \{x_{t(v,c)} = 1 - y_1\} \cup \{\alpha_1 + y_1 + y_2 \geqslant \varepsilon\} \cup \{\alpha_1 - y_1 + y_2 \leqslant 1\} \cup \{\alpha_2 \leqslant 1 - y_1\} \cup \{x_c \leqslant (n-\varepsilon) - (n - \varepsilon - k_2)y_2\} \cup \{y_i \in \{0, 1\}\}$, where $y_i$ are new variables.
- $(\leqslant_{\Delta_D})$ If $\langle (\leqslant t\ n), \alpha_1\rangle \in \mathcal{L}(v)$ then: (i) if $\langle t(v, c), \beta\rangle \notin \mathcal{L}(v, c)$ then add $\langle t(v, c), \alpha_1\rangle$ to $\mathcal{L}(v, c)$, (ii) for some $c$ such that $\langle t(v, c), \beta\rangle \in \mathcal{L}(v, c)$, $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{\alpha_1 + y \geqslant \varepsilon\} \cup \{\alpha_1 + y \leqslant 1\} \cup \{x_{t(v,c)} \geqslant 1 - y\} \cup \{x_c \leqslant n - (n - k_2)y\} \cup \{y \in \{0, 1\}\}$, where $y$ is a new variable.
- $(\neg\leqslant_{\Delta_D})$ If (i) $\langle \neg(\geqslant t\ n), \alpha_1\rangle \in \mathcal{L}(v)$ and (ii) $\langle t(v, c), \alpha_2\rangle \in \mathcal{L}(v, c)$, then $\mathcal{C}_{\mathcal{F}} = \mathcal{C}_{\mathcal{F}} \cup \{y_1 \leqslant y_2\} \cup \{x_{t(v,c)} = 1 - y_1\} \cup \{\alpha_1 + y_1 + y_2 \geqslant \varepsilon\} \cup \{\alpha_1 - y_1 + y_2 \leqslant 1\} \cup \{\alpha_2 \leqslant 1 - y_1\} \cup \{x_c \geqslant (n+\varepsilon) - (n + \varepsilon - k_1)y_2\} \cup \{y_i \in \{0, 1\}\}$, where $y_i$ are new variables.

($=_{\Delta_D}$) If $\langle(= t\ n), \alpha_1\rangle \in \mathcal{L}(v)$ then: (i) if $\langle t(v, c), \beta\rangle \notin \mathcal{L}(v, c)$ then add $\langle t(v, c), \alpha_1\rangle$ to $\mathcal{L}(v, c)$, (ii) for some $c$ such that $\langle t(v, c), \beta\rangle \in \mathcal{L}(v, c), \mathcal{C_F} = \mathcal{C_F} \cup \{\alpha_1 + y \geqslant \varepsilon\} \cup \{\alpha_1 + y \leqslant 1\} \cup \{x_{t(v,c)} \geqslant 1 - y\} \cup \{x_c \geqslant n - (n - k_1)y\} \cup \{x_c \leqslant n - (n - k_2)y\} \cup \{y \in \{0, 1\}\}$, where $y$ is a new variable.

($\neg =_{\Delta_D}$) If (i) $\langle\neg(= t\ n), \alpha_1\rangle \in \mathcal{L}(v)$ and (ii) $\langle t(v, c), \alpha_2\rangle \in \mathcal{L}(v, c)$, then: $\mathcal{C_F} = \mathcal{C_F} \cup \{y_2 \geqslant y_1\} \cup \{y_3 \geqslant y_2\} \cup \{x_{t(v,c)} = 1 - y_1\} \cup \{\alpha_1 + y_1 + y_2 \geqslant \varepsilon\} \cup \{\alpha_1 - y_1 + y_2 \leqslant 1\} \cup \{\alpha_2 \leqslant 1 - y_1\} \cup \{x_c \leqslant (n - \varepsilon) - (n - \varepsilon - k_2)y_2 - (n - \varepsilon - k_2)(1 - y_3)\} \cup \{x_c \geqslant (n + \varepsilon) - (n + \varepsilon - k_1)y_3\} \cup \{y_i \in \{0, 1\}\}$, where $y_i$ are new variables.

Rules ($\geqslant_{\Delta_D}$), ($\leqslant_{\Delta_D}$) and ($=_{\Delta_D}$) start by checking if there exists a concrete individual $c$ such that it is related with $v$ by means of feature $t$ and otherwise such a relation is created. Hence, before step (ii) is applied, there exists a relation $\langle t(v, c), \beta\rangle \in \mathcal{L}(v, c)$ for some $\beta \in [0, 1]$, to which these rules are applied.

On the other hand, rules ($\neg\geqslant_{\Delta_D}$), ($\neg\leqslant_{\Delta_D}$) and ($\neg =_{\Delta_D}$) are applied to every concrete individual $c$ such that $\langle t(v, c), \alpha_2\rangle \in \mathcal{L}(v, c)$, but since $t$ is functional, $c$ is unique for every node $v$.

Notice that strict inequalities are not allowed, and hence for any linear expression $\tau$, the inequality $\tau > 0$ is expressed as $\tau \geqslant \varepsilon$, for a sufficiently small $\varepsilon > 0$.

**Example 6.2.** Let us comment the ($\neg\geqslant_{\Delta_D}$)-rule. If $\langle\neg(\geqslant \mathit{hasAge}\ 18), \alpha_1\rangle \in \mathcal{L}(v)$ and, for some $c$, $\langle t(v, c), \alpha_2\rangle \in \mathcal{L}(v, c)$, there are three possibilities to satisfy these formulae:

1. $\alpha_1 = 0$, because $\langle\neg(\geqslant \mathit{hasAge}\ 18), 0\rangle$ is always true.
2. $x_{\mathit{hasAge}(v,c)} = 0$, since it implies $[\ominus \mathit{hasAge}(a, b) \oplus (b < n)] = 1$ and hence $\langle\ominus(\geqslant \mathit{hasAge}\ 18), \alpha_1\rangle$ is true. In order to satisfy $\mathit{hasAge}(v, c) \geqslant \alpha_2$, $\alpha_2$ must be 0.
3. $x_{\mathit{hasAge}(v,c)} = 1$ and $x_c < n$, because this implies $[\ominus \mathit{hasAge}(a, b) \oplus (b < n)] = 1$ and hence $\langle\ominus(\geqslant \mathit{hasAge}\ 18), \alpha_1\rangle$ is true.

In order to cover all these possibilities, the control variables $y_1$ and $y_2$ simulate three branches:

1. $y_1 = 0,\ y_2 = 1$. In this case, $\alpha_1 = 0$ and hence the other variables are not constrained ($x_{\mathit{hasAge}(v,c)} = 1$, $\alpha_2 \leqslant 1$, $x_c \leqslant k_2$).
2. $y_1 = 1,\ y_2 = 1$. Now, $x_{\mathit{hasAge}(v,c)} = 0$ and $\alpha_2 = 0$. $x_c$ is not constrained ($x_c \leqslant k_2$).
3. $y_1 = 0,\ y_2 = 0$. Then, $x_{\mathit{hasAge}(v,c)} = 1$ and $x_c < 18$.
4. Since $y_1 \leqslant y_2$, the case $y_1 = 1$ and $y_2 = 0$ is not possible.

**Example 6.3.** Let us consider now the ($\neg =_{\Delta_D}$)-rule. The scenario is similar to the one in Example 6.2 but now there are four possibilities. In fact, $(b \neq n)$ is true if either $b > n$ or $b < n$. Now, variables $y_1$, $y_2$ and $y_3$ simulate four branches:

1. $y_1 = 0,\ y_2 = 1,\ y_3 = 1$. In this case, $\alpha_1 = 0$ and hence the other variables are not constrained.
2. $y_1 = 1,\ y_2 = 1,\ y_3 = 1$. Now, $x_{\mathit{hasAge}(v,c)} = 0$ and $\alpha_2 = 0$.
3. $y_1 = 0,\ y_2 = 0,\ y_3 = 0$. Then, $x_{\mathit{hasAge}(v,c)} = 1$ and $x_c > n$.
4. $y_1 = 0,\ y_2 = 0,\ y_3 = 1$. Then, $x_{\mathit{hasAge}(v,c)} = 1$ and $x_c < n$.
5. Since $y_1 \leqslant y_2$ and $y_2 \leqslant y_3$, the other cases are not possible.

*6.3.3. Inference rules for strings*

First of all, some preprocessing is needed. Essentially, we collect all strings in the KB, order them in alphabetical order and assign a progressive natural number to them. The algorithm is:

1. Get the set of strings $S$ appearing in the fuzzy KB $\mathcal{K}$ i.e., for every concrete feature $t$ such that (cf $t$ string) $\in \mathcal{K}$, consider $\{s : s \text{ is a string}, (\bowtie t\ s) \in \mathcal{K}, \bowtie\ = \{\geqslant, \leqslant, =\}\}$.
2. Order $S = \{s_1, \ldots, s_n\}$ in alphabetical order.
3. Compute for every string $s_i$ its order $\mathit{order}(s_i)$ (which is obviously an integer in $[1, n]$).

Then, we add the following rule (which should have higher priority than inference rules for integer and reals):

($\mathit{string}$) If $\langle(\bowtie t\ \ s), \alpha\rangle \in \mathcal{L}(v)$, with $s$ being a string then: (i) replace $\langle(\bowtie t\ \ s), \alpha\rangle$ with $\langle(\bowtie t\ \ \mathit{order}(s)), \alpha\rangle$ where $\bowtie = \{\geqslant, \leqslant, =\}$ and $\mathit{order}(s)$ is the ordinal assigned to $s$, (ii) change the type of $t$, becoming real.

**Example 6.4.** Suppose we want to know if the KB $\mathcal{K} = \{\langle x\colon(\geqslant hasName \text{ "}u\text{"}), 1\rangle, \langle x\colon(\leqslant hasName \text{ "}v\text{"}), 1\rangle, \langle x\colon(= hasName \text{ "}umberto\text{"}), 1\rangle\}$ is consistent.

To start with, the order of every string in $\mathcal{K}$ is computed: *order*("*u*") = 1, *order*("*umberto*") = 2, *order*("*v*") = 3. Then, we construct a forest $\mathcal{F}$ with a root node $x$ and $\mathcal{L}(x) = [\langle(\geqslant hasName \text{ "}u\text{"}), 1\rangle, \langle(\leqslant hasName \text{ "}v\text{"}), 1\rangle], \langle(= hasName \text{ "}umberto\text{"}), 1\rangle]$.

Now, we apply (*string*) rule to $\mathcal{L}(x)$. We consider the tuple $\langle(\geqslant hasName \text{ "}u\text{"}), 1\rangle$ and we replace the string "*u*" with the number $order(u) = 1$. Once applied the rule two more times, we have that $\mathcal{L}(x) = [\langle(\geqslant hasName\ 1), 1\rangle, \langle(\leqslant hasName\ 3), 1\rangle], \langle(= hasName \text{ "}umberto\text{"}), 2\rangle]$.

Next, $(\geqslant_{\Delta_D})$, $(\leqslant_{\Delta_D})$ and $(=_{\Delta_D})$ are applied. It is easy to see that the set of constraints which is generated has a solution, so $\mathcal{K}$ is satisfiable.

Finally, we will discuss correctness, completeness and termination results of the reasoning procedure for $\mathcal{ALCF}(\mathbf{D})$.

**Proposition 6.1.** *A $\mathcal{ALCF}(\mathbf{D})$ KB $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ is satisfiable iff there exists a fuzzy tableau for $\mathcal{K}$.*

**Proposition 6.2** (*Termination*). *For each $\mathcal{ALCF}(\mathbf{D})$ KB $\mathcal{K}$, the tableau algorithm terminates.*

**Proposition 6.3** (*Soundness*). *If the expansion rules can be applied to an $\mathcal{ALCF}(\mathbf{D})$ KB $\mathcal{K}$ such that they yield a complete completion-forest $\mathcal{F}$ such that $\mathcal{C}_{\mathcal{F}}$ has a solution, then $\mathcal{K}$ has a fuzzy tableau for $\mathcal{K}$.*

**Proposition 6.4** (*Completeness*). *Consider an $\mathcal{ALCF}(\mathbf{D})$ KB $\mathcal{K}$. If $\mathcal{K}$ has a fuzzy tableau, then the expansion rules can be applied in such a way that the tableaux algorithm yields a complete completion-forest for $\mathcal{K}$ such that $\mathcal{C}_{\mathcal{F}}$ has a solution.*

The proofs of these propositions can straightforwardly be obtained by extending the ones described in Section 4.

## 7. Conclusions

In this work we have provided a reasoning algorithm for a general family of fuzzy DLs which extend a general t-norm with an involutive negation (Łukasiewicz negation). In general, universal and existential quantifiers are not inter-definable. The reasoning algorithm combines tableaux rules with a reduction to a bMINLP problem. The algorithm uses a fuzzy version of the lazy expansion technique, as well as a more sophisticated technique with respect to the related work which makes possible to save some of the generated variables. In some particular cases, we can obtain an easier optimization problem. For example, for Zadeh and Łukasiewicz t-norms we end up with a bMILP problem [43,44,47], and for Product t-norm we end up with a bMIQCP problem. We have provided the set of rules that make possible to reason with Product t-norm.

As already mentioned, there is a significative difference with other similar algorithms for fuzzy DLs combining tableau algorithms with optimization problems [9,43,47]. In those algorithms, every time a concept $C$ appears in the list of expressions of a node $v$, a new variable $x$ is created. In this paper, we introduce a variable $x_{v:C}$ once, and reuse it the following times. This reduction in the number of generated variables is important because it makes the bMINLP problem easier to solve.

We have also proposed a fuzzy extension of $\mathcal{ALCF}(\mathbf{D})$ which allows to use fuzzy modifiers, fuzzy membership functions and concrete features relating individuals with strings, real or integer numbers. This extension is general in the sense that does not depend on any particular semantics of the fuzzy operators. Moreover, the constraints that these features introduce are linear, so they do not add extra complexity, that is, in Łukasiewicz/Zadeh, Product or a general t-norm-based fuzzy DL, we still end up with a bMILP, a bMIQCP or a bMINLP problem, respectively.

The algorithms to reason with general t-norms are currently under implementation as an extension of the *fuzzyDL* reasoner [10]. Concrete features are already supported, as well as Łukasiewicz, Zadeh and Gödel fuzzy logics.

Using Hájek's translation [18], it is possible to reduce fuzzy $\mathcal{ALC}$ to fuzzy propositional logics with an involutive negation [14], although the reduction is exponential in size. For example, our fuzzy DL under Product logic can be reduced to $\Pi_{\sim}$ (propositional Product logic with an involutive negation [14]). The advantage of our approach is that we can also support $\mathcal{ALCF}(\mathbf{D})$, with fuzzy modifiers, fuzzy membership functions and concrete features.

The results could be extended to more expressive fuzzy DLs, such as $\mathcal{SHIF}(D)$ and $\mathcal{SHOIN}(D)$, the logics behind OWL-DL and OWL-Lite, by using an appropriate blocking condition similarly as done in [25]. It would also be interesting to support fuzzy general (i.e., unrestricted) concept inclusion axioms.


## Acknowledgement

We would like to thank to the anonymous referees for their very valuable comments on an earlier version of this paper.


## References


[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P.F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, Cambridge, 2003.
[2] F. Baader, E. Franconi, B. Hollunder, B. Nebel, H.J. Profitlich, An empirical analysis of optimization techniques for terminological representation systems, or: Making KRIS get a move on, Applied Artificial Intelligence 4 (1994) 109–132.
[3] F. Baader, I. Horrocks, U. Sattler, Description logics as ontology languages for the semantic web, in: Mechanizing Mathematical Reasoning: Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday, Lecture Notes in Artificial Intelligence, Vol. 2605, Springer, Berlin, 2005, pp. 228–248.
[4] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Scientific American 284 (5) (2001) 34–43.
[5] F. Bobillo, M. Delgado, J. Gómez-Romero, A crisp representation for fuzzy $\mathcal{SHOIN}$ with fuzzy nominals and general concept inclusions, in: Uncertainty Reasoning for the Semantic Web I, Lecture Notes in Artificial Intelligence, Vol. 5327, Springer, Berlin, 2008, pp. 174–188.
[6] F. Bobillo, M. Delgado, J. Gómez-Romero, Optimizing the crisp representation of the fuzzy Description Logic $\mathcal{SROIQ}$, in: Uncertainty Reasoning for the Semantic Web I, Lecture Notes in Artificial Intelligence, Vol. 5327, Springer, Berlin, 2008, pp. 189–206.
[7] F. Bobillo, M. Delgado, J. Gómez-Romero, Crisp representations and reasoning for fuzzy ontologies, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, in press.
[8] F. Bobillo, M. Delgado, J. Gómez-Romero, U. Straccia, Fuzzy description logics under Gödel semantics, International Journal of Approximate Reasoning 50 (3) (2009) 494–514.
[9] F. Bobillo, U. Straccia, A fuzzy description logic with product t-norm, in: Proc. 16th IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE 2007), 2006, pp. 652–657.
[10] F. Bobillo, U. Straccia, fuzzyDL: an expressive fuzzy Description Logic reasoner, in: Proc. 17th IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE 2008), 2008, pp. 923–930.
[11] F. Bobillo, U. Straccia, Towards a crisp representation of fuzzy Description Logics under Łukasiewicz semantics, in: Proc. 17th Internat. Sympos. on Methodologies for Intelligent Systems (ISMIS 2008), Lecture Notes in Artificial Intelligence, Vol. 4994, Springer, Berlin, 2008, pp. 309–318.
[12] P. Cintula, An alternative approach to the Ł$\Pi$ logic, Neural Network World 11 (6) (2001) 561–572.
[13] B. Cuenca-Grau, I. Horrocks, B. Motik, B. Parsia, P.F. Patel-Schneider, U. Sattler, OWL 2: the next step for OWL, Journal of Web Semantics 6 (4) (2008) 309–322.
[14] F. Esteva, L. Godo, P. Hájek, M. Navara, Residuated fuzzy logics with an involutive negation, Archive for Mathematical Logic 39 (2) (2000) 103–124.
[15] F. Esteva, L. Godo, F. Montagna, The Ł$\Pi$ and Ł$\Pi\frac{1}{2}$ logics: two complete fuzzy systems joining Łukasiewicz and Product logics, Archive for Mathematical Logic 40 (1) (2001) 39–67.
[16] C.A. Floudas, Nonlinear and Mixed-integer Optimization, Oxford University Press, Oxford, 1995.
[17] R. Hähnle, Advanced many-valued logics, in: Handbook of Philosophical Logic, Vol. 2, second ed., Kluwer Academic Publishers, Dordrecht, 2001.
[18] P. Hájek, Making fuzzy description logics more general, Fuzzy Sets and Systems 154 (1) (2005) 1–15.
[19] P. Hájek, Metamathematics of Fuzzy Logic, Kluwer Academic Publishers, Dordrecht, 2001.
[20] P. Hájek, On witnessed models in fuzzy logic, Mathematical Logic Quarterly 53 (1) (2007) 66–77.
[21] P. Hájek, What does mathematical fuzzy logic offer to description logic? in: Capturing Intelligence: Fuzzy Logic and the Semantic Web, Elsevier, Amsterdam, 2006, pp. 91–100.
[22] S. Hölldobler, H.P. Störr, T.D. Khang, The fuzzy description logic $\mathcal{ALC}_{FH}$ with hedge algebras as concept modifiers, Journal of Advanced Computational Intelligence 7 (3) (2003) 294–305.
[23] I. Horrocks, P.F. Patel-Schneider, Reducing OWL entailment to description logic satisfiability, Journal of Web Semantics 1 (4) (2004) 345–357.
[24] I. Horrocks, P.F. Patel-Schneider, From $\mathcal{SHIQ}$ and RDF to OWL: the making of a Web ontology language, Journal of Web Semantics 1 (1) (2003) 7–26.
[25] I. Horrocks, U. Sattler, S. Tobies, Practical reasoning for very expressive description logics, Logic Journal of the Interest Group in Pure and Applied Logic 8 (3) (2000) 239–263.
[26] I. Horrocks, U. Sattler, S. Tobies, Reasoning with individuals for the description logic $\mathcal{SHIQ}$, in: Proc. 17th Internat. Conf. on Automated Deduction (CADE-17), Lecture Notes in Artificial Intelligence, Vol. 1831, Springer, Berlin, 2000, pp. 482–496.

[27] R.G. Jeroslow, Logic-based Decision Support. Mixed Integer Model Formulation, Elsevier, Amsterdam, 1989.
[28] R. Knappe, H. Bulskov, T. Andreasen, Perspectives on ontology-based querying, International Journal of Intelligent Systems 22 (7) (2006) 739–761.
[29] T. Łukasiewicz, Expressive probabilistic description logics, Artificial Intelligence 172 (6–7) (2008) 852–883.
[30] T. Łukasiewicz, U. Straccia, Managing uncertainty and vagueness in description logics for the semantic web, Journal of Web Semantics 6 (4) (2008) 291–308.
[31] C. Lutz, Description logics with concrete domains—A survey, Advances in Modal Logics, Vol. 4, King's College Publications, 2003.
[32] D. L. McGuinness, F. van Harmelen, OWL Web Ontology Language overview, 2004, Retrieved October 1, 2007, from ⟨http://www.w3.org/TR/owl-features⟩.
[33] B. Nebel, Terminological reasoning is inherently intractable, Artificial Intelligence 43 (2) (1990) 235–249.
[34] H. Salkin, M. Kamlesh, Foundations of Integer Programming, North-Holland, Amsterdam, 1988.
[35] M. Schmidt-Schauß, G. Smolka, Attributive concept descriptions with complements, Artificial Intelligence 48 (1991) 1–26.
[36] G. Stoilos, G. Stamou, Extending fuzzy description logics for the semantic web, in: Proc. 3rd Internat. Workshop on OWL: Experiences and Directions (OWLED 2007), 2007.
[37] G. Stoilos, G. Stamou, J.Z. Pan, Handling imprecise knowledge with fuzzy description logic, in: Proc. 2006 Internat. Workshop on Description Logics (DL 2006), 2006, pp. 119–126.
[38] G. Stoilos, G. Stamou, V. Tzouvaras, J.Z. Pan, I. Horrocks, Fuzzy OWL: uncertainty and the semantic web, in: Proc. 1st Internat. Workshop on OWL: Experiences and Directions (OWLED 2005), 2005.
[39] G. Stoilos, G. Stamou, V. Tzouvaras, J.Z. Pan, I. Horrocks, Reasoning with very expressive fuzzy description logics, Journal of Artificial Intelligence Research 30 (8) (2007) 273–320.
[40] G. Stoilos, U. Straccia, G. Stamou, J.Z. Pan, General concept inclusions in fuzzy description logics, in: Proc. 17th European Conf. on Artificial Intelligence (ECAI-06), IOS Press, 2006, pp. 457–461.
[41] U. Straccia, A fuzzy description logic, in: Proc. 15th National Conf. on Artificial Intelligence (AAAI-98), 1998, pp. 594–599.
[42] U. Straccia, A fuzzy description logic for the semantic web, in: Fuzzy Logic and the Semantic Web, Capturing Intelligence, Elsevier, Amsterdam, 2006, pp. 73–90 (Chapter 4).
[43] U. Straccia, Description logics with fuzzy concrete domains, in: Proc. 21st Conf. on Uncertainty in Artificial Intelligence (UAI-05), AUAI Press, 2005, pp. 559–567.
[44] U. Straccia, Reasoning in Ł-$\mathcal{SHIF}$: an expressive fuzzy description logic under Łukasiewicz semantics, Technical Report ISTI-018/2007, ISTI-CNR, 2007.
[45] U. Straccia, Reasoning within fuzzy description logics, Journal of Artificial Intelligence Research 14 (2001) 137–166.
[46] U. Straccia, Transforming fuzzy description logics into classical description logics, in: Proc. 9th European Conf. on Logics in Artificial Intelligence (JELIA-04), Lecture Notes in Computer Science, Vol. 3229, Springer, Berlin, 2004, pp. 385–399.
[47] U. Straccia, F. Bobillo, Mixed integer programming, general concept inclusions and fuzzy description logics, Mathware & Soft Computing 14 (3) (2007) 247–259.
[48] C. Tresp, R. Molitor, A description logic for vague knowledge, in: Proc. 13th European Conf. on Artificial Intelligence (ECAI-98), 1998.
[49] J. Yen, Generalizing term subsumption languages to fuzzy logic, in: Proc. 12th Internat. Joint Conf. on Artificial Intelligence (IJCAI-91), 1991, pp. 472–477.
[50] L. Zadeh, Fuzzy sets, Information and Control 8 (3) (1965) 338–353.