A MILP-based decision procedure for the (Fuzzy) Description Logic $\mathcal{ALCB}$

Fernando Bobillo[1] and Umberto Straccia[2]

[1] Dpt. of Computer Science & Systems Engineering, University of Zaragoza, Spain
[2] Istituto di Scienza e Tecnologie dell'Informazione (ISTI - CNR), Pisa, Italy
Email: fbobillo@unizar.es, straccia@isti.cnr.it

Abstract. To overcome the inability of Description Logics (DLs) to represent vague or imprecise information, several fuzzy extensions have been proposed in the literature. In this context, an important family of reasoning algorithms for fuzzy DLs is based on a combination of tableau algorithms and Operational Research (OR) problems, specifically using Mixed Integer Linear Programming (MILP). In this paper, we present a MILP-based tableau procedure that allows to reason within fuzzy $\mathcal{ALCB}$, *i.e.*, $\mathcal{ALC}$ with individual value restrictions. Interestingly, unlike classical tableau procedures, our tableau algorithm is deterministic, in the sense that it defers the inherent non-determinism in $\mathcal{ALCB}$ to a MILP solver.

1 Introduction

Description Logics (DLs for short) [2] are a family of logics for representing structured knowledge. In the last two decades, DLs have gained even more popularity due to their application in the context of the *Semantic Web* [3]. Indeed, the current standard language for specifying ontologies is the Web Ontology Language (OWL 2) [18], which is based on the DL $\mathcal{SROIQ}(\mathbf{D})$ [23].

Fuzzy DLs have been proposed as an extension to classical DLs with the aim of dealing with *fuzzy/vague/imprecise concepts* by including elements of fuzzy logic [35]. In fuzzy DLs, the axioms may not be bivalent, but instead can be satisfied with a certain degree of truth (typically, a truth value in $[0, 1]$). Since the first work of J. Yen in 1991 [34], an important number of works can be found in the literature (good surveys on *fuzzy DLs* can be found on [25,33]).

Several families of algorithms to reason with fuzzy DLs have been proposed in the literature. The most important ones are tableau algorithms [15,28,29,30], tableau algorithms combined with Operational Research (OR) problems [8,17], automata-based algorithms [14,13], reduction to classical DLs [4,5,6,10,12], and reduction to fuzzy logics [16,19,22].

Some of the existing algorithms already support nominals. A tableau algorithm for Zadeh $\mathcal{SHOIQ}$ [28] and an algorithm to check subsumption in Gödel $\mathcal{EL}^{++}$ [26] are both able to deal with nominals. There are also reductions to classical DLs for fuzzy $\mathcal{SROIQ}(\mathbf{D})$ under Zadeh [4], finite Gödel [5], finite Łukasiewicz [10], and any finite t-norm [6]; but also for $\mathcal{SHOI}$ for every t-norm not starting with the Łukasiewicz t-norm [12].

However, to the best of our knowledge, none of the existing OR-based tableau algorithms is able to support nominals so far. This family of algorithms is interesting for several reasons: *(i)* it is very suitable to manage fuzzy datatypes [31] or fuzzy concepts without a counterpart in classical DLs, such as fuzzy modified concepts [31] or aggregated concepts [11]; *(ii)* it makes it possible to reason with other t-norms different than Gödel [8]; and *(iii)* the arguably most popular fuzzy ontology reasoner `fuzzyDL` implements one of these algorithms [7].

The objective of this paper is to start filling this gap by proposing a novel decision procedure for a fuzzy DL with individual value restrictions, namely $\mathcal{ALCB}$. $\mathcal{ALCB}$ is the basic DL language $\mathcal{ALC}$ [31] extended with individual value restrictions. Under the restriction that the TBox is acyclic, our algorithm applies for Łukasiewicz, Zadeh fuzzy DLs, and for classical DLs.

The rest of this paper is organised as follows. Section 2 includes some preliminary notions. Section 3 presents our reasoning algorithm and Section 4 a running example. Finally, Section 5 sets out some conclusions and ideas for future work.

2 Fuzzy DLs Basics

In this section we overview some basic definitions on mathematical fuzzy logic and the fuzzy DL $\mathcal{ALCB}$ (see [25,33] for a more in depth presentation).

Mathematical Fuzzy Logic. In *Mathematical Fuzzy Logic* [21], the usual convention prescribing that a statement is either true or false is changed and is a matter of degree measured on an ordered scale that is no longer $\{0, 1\}$, but e.g., $[0, 1]$. This degree is called *degree of truth* of the logical statement ϕ in the interpretation $\mathcal{I}$. For us, *fuzzy statements* have the form $\langle \phi, \alpha \rangle$, where $\alpha \in (0, 1]$ and ϕ is a statement, encoding that the degree of truth of ϕ is *greater than or equal to* α.

Fuzzy logics provide compositional calculi of degrees of truth. The conjunction, disjunction, complement and implication operations are performed in the fuzzy case by a t-norm function $\otimes$, a t-conorm function $\oplus$, a negation function $\ominus$ and an implication function $\Rightarrow$, respectively (see [21] for definitions and properties of these functions). A quadruple composed by a t-norm, a t-conorm, an implication function and a negation function determines a *fuzzy logic*. One usually distinguishes three fuzzy logics, namely Łukasiewicz, Gödel, and Product [21], due to the fact that any continuous t-norm can be obtained as a combination of Łukasiewicz, Gödel, and Product t-norm [27]. It is also usual to consider also Zadeh logic. The combination functions of these logics can be found in Table 1.

It is easy to see that Zadeh fuzzy logic can be expressed using Łukasiewicz fuzzy logic, as $\min(\alpha, \beta) = \alpha \otimes_{\text{Ł}} (\alpha \Rightarrow_{\text{Ł}} \beta), \max(\alpha, \beta) = 1 - \min(1 - \alpha, 1 - \beta)$, and $\alpha \Rightarrow_{KD} \beta = \max(1 - \alpha, \beta)$. This latter implication is called *Kleene-Dienes implication*. The name of Zadeh fuzzy logic is used following the tradition in the setting of fuzzy DLs, even if the name may lead to confusion because the logic does not usually include Rescher implication, sometimes called *Zadeh implication* as well. This implication is defined as $\alpha \Rightarrow \beta = 1$ if $\alpha \leq \beta$; 0 otherwise.

Table 1. Truth combination functions of various fuzzy logics.

	Łukasiewicz logic	Gödel logic	Product logic	Zadeh logic
$\alpha \otimes \beta$	$\max(\alpha+\beta-1,0)$	$\min(\alpha,\beta)$	$\alpha \cdot \beta$	$\min(\alpha,\beta)$
$\alpha \oplus \beta$	$\min(\alpha+\beta,1)$	$\max(\alpha,\beta)$	$\alpha+\beta-\alpha\cdot\beta$	$\max(\alpha,\beta)$
$\alpha \Rightarrow \beta$	$\min(1-\alpha+\beta,1)$	$\begin{cases}1 & \text{if } \alpha \leq \beta \\ \beta & \text{otherwise}\end{cases}$	$\min(1,\beta/\alpha)$	$\max(1-\alpha,\beta)$
$\ominus\alpha$	$1-\alpha$	$\begin{cases}1 & \text{if } \alpha = 0 \\ 0 & \text{otherwise}\end{cases}$	$\begin{cases}1 & \text{if } \alpha = 0 \\ 0 & \text{otherwise}\end{cases}$	$1-\alpha$

The Fuzzy DL $\mathcal{ALCB}$. This logic is obtained by extending fuzzy $\mathcal{ALC}$ with *individual value restrictions* (indicated with the letter $\mathcal{B}$). It is a sublogic of the fuzzy DL presented at [32]. Let $\mathbf{A}$ be a set of *concept names* (also called atomic concepts), $\mathbf{R}$ be a set of *role names*, and $\mathbf{I}$ be a set of *individual names*. Each role $R \in \mathbf{R}$ is also called an *object property*. The set of *concepts* (denoted C, D) is built from concept names $A \in \mathbf{A}$ using connectives and quantification constructs over roles R and individuals $a \in \mathbf{I}$, according to the following syntactic rule:

$$
\begin{array}{rcl}
C, D \rightarrow & A & \mid \text{(atomic concept)} \\
& \top & \mid \text{(universal concept)} \\
& \bot & \mid \text{(bottom concept)} \\
& \neg C & \mid \text{(concept negation)} \\
& C \sqcap D & \mid \text{(concept conjunction)} \\
& C \sqcup D & \mid \text{(concept disjunction)} \\
& \forall R.C & \mid \text{(universal restriction)} \\
& \exists R.C & \mid \text{(existential restriction)} \\
& \exists R.\{a\} & \mid (\textit{individual value restriction}) \ .
\end{array}
$$

Now we will define the *axioms* that can be expressed in a fuzzy ontology. A fuzzy *knowledge base* or fuzzy ontology is a tuple $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$, where $\mathcal{A}$ is an ABox with assertional axioms and $\mathcal{T}$ is a TBox with terminological axioms. In the axioms, we will use $\alpha \in (0,1]$ to denote a truth value. If α is omitted, $\alpha = 1$ is assumed.

An *ABox* $\mathcal{A}$ (Assertional Box) is a finite set of concept assertions or role assertions. A *concept assertion* $\langle a{:}C, \alpha \rangle$ states that a is an instance of concept C to degree at least α. Furthermore, a *role assertion* $\langle (a_1, a_2){:}R, \alpha \rangle$ indicates that (a_1, a_2) is an instance of role R to degree at least α.

A *TBox* $\mathcal{T}$ (Terminological Box) is a finite set of *General Concept Inclusion* (GCI) axioms of the form $\langle C \sqsubseteq D, \alpha \rangle$, indicating that C is a sub-concept of D to degree at least α. C is called the *head* and D is the *body* of the axiom. $C = D$ can be used as a shorthand for both $\langle C \sqsubseteq D, 1 \rangle$ and $\langle D \sqsubseteq C, 1 \rangle$. For a concept name A, we say that a *definitional axiom* is of the form $A = C$, while a *primitive inclusion axiom* is of the form $A \sqsubseteq C$. Furthermore, we say that A_1 *directly uses* A_2 w.r.t. $\mathcal{T}$ if A_1 is the head of some axiom and A_2 occurs in its body. Let *uses* be the transitive closure of the relation directly uses. $\mathcal{T}$ is *acyclic* if it has primitive or definitional axioms only, a concept name A is the head of at most one definitional axiom in $\mathcal{T}$, there is no concept name A in the head of both a definitional and primitive inclusion axiom, and there is no concept name A such that A uses A.

Example 1. We have built a fuzzy wine ontology [3] according to the FuzzyOWL 2 proposal [9]. Ontologies often use individual value restrictions to state geographical origins. For instance, the fuzzy wine ontology contains the following definition of Tuscan wines: $\mathsf{TuscanWine} = \mathsf{Wine} \sqcap \exists \mathsf{locatedIn}.\{\mathsf{TuscanyRegion}\}$.

Now, let us formally specify the semantics. Let us fix a fuzzy logic. In classical DLs, an interpretation $\mathcal{I}$ maps e.g., a concept C into a set of individuals $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, *i.e.*, $\mathcal{I}$ maps C into a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow \{0,1\}$ (either an individual belongs to the extension of C or does not belong to it). However, in fuzzy DLs, $\mathcal{I}$ maps C into a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow [0,1]$ and, thus, an individual belongs to the extension of C to some degree in $[0,1]$, *i.e.*, $C^{\mathcal{I}}$ is a fuzzy set. Specifically, a *fuzzy interpretation* is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a non-empty (crisp) set $\Delta^{\mathcal{I}}$ (the *domain*) and of a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$ that assigns:

1. to each atomic concept A a function $A^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \rightarrow [0,1]$;
2. to each object property R a function $R^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \rightarrow [0,1]$;
3. to each individual a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ such that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$ (called *Unique Name Assumption*, UNA). UNA is often not assumed in DLs.

Let $x, y \in \Delta^{\mathcal{I}}$ be elements of the domain. The fuzzy interpretation function is extended to complex concepts as follows:

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

The *satisfiability of axioms* is then defined by the following conditions:

1. $\mathcal{I}$ satisfies $\langle a \colon C, \alpha \rangle$ if $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq \alpha$;
2. $\mathcal{I}$ satisfies $\langle (a,b) \colon R, \alpha \rangle$ if $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq \alpha$;
3. $\mathcal{I}$ satisfies $\langle C \sqsubseteq D, \alpha \rangle$ if $\inf_{x \in \Delta^{\mathcal{I}}} \{C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x)\} \geq \alpha$.

Exceptionally, Zadeh fuzzy DLs use Zadeh implication in the semantics of GCIs, since Kleene-Dienes implication produce some counter-intuitive effects [4]. Note that classical $\mathcal{ALCB}$ is a particular case of this fuzzy $\mathcal{ALCB}$. Finally, $\mathcal{I}$ is a model of $\mathcal{K}$ iff $\mathcal{I}$ satisfies each axiom in $\mathcal{K}$.

[3] `http://www.straccia.info/software/FuzzyOWL/ontologies/FuzzyWine.1.0.owl`

Reasoning tasks. Let $\mathcal{K}$ be a fuzzy KB, C, D be fuzzy concepts, and α a degree of truth. We can define the following reasoning tasks:

– *Consistency.* $\mathcal{K}$ is consistent iff it has a model, i.e., there is a fuzzy interpretation $\mathcal{I}$ satisfying every axiom in $\mathcal{K}$.
– *Entailment*: $\mathcal{K}$ entails an axiom $\langle \tau, \alpha \rangle$ iff every model of $\mathcal{K}$ satisfies $\langle \tau, \alpha \rangle$.
– *Concept satisfiability.* C is satisfiable to at least degree α (or α-satisfiable) w.r.t. $\mathcal{K}$ iff there is a model $\mathcal{I}$ of $\mathcal{K}$ such that $C^{\mathcal{I}}(x) \geq \alpha$ for some $x \in \Delta^{\mathcal{I}}$.
– *Concept subsumption.* D subsumes C to at least degree α (or α-subsumes) w.r.t. $\mathcal{K}$ iff every model of $\mathcal{K}$ satisfies the axiom $\langle C \sqsubseteq D, \alpha \rangle$.
– *Best entailment degree* (BED). The BED (also called greatest lower bound) of an axiom $\phi \in \{a{:}C, (a,b){:}R, C \sqsubseteq D\}$ w.r.t. $\mathcal{K}$ is defined as $bed(\mathcal{K}, \phi) = \sup\{\alpha : \mathcal{K} \models \langle \phi, \alpha \rangle\}$, where $\sup \emptyset = 0$.

As usual in DLs and fuzzy DLs, reasoning tasks are often mutually inter-definable and each of the previous tasks can be reduced to the BED. For instance:

– $\mathcal{K}$ is not consistent iff $bed(\mathcal{K}, a{:}\bot) = 1$, where a is new individual. Recall that inconsistent ontologies entail everything.
– $\mathcal{K}$ entails $\langle \tau, \alpha \rangle$ iff $bed(\mathcal{K}, \tau) \geq \alpha$.
– C is α-satisfiable w.r.t. $\mathcal{K}$ iff $\mathcal{K} \cup \{\langle a{:}C, \alpha \rangle\}$ is consistent, where a is new individual. Note that consistency has already been reduced to the BED.
– D α-subsumes C w.r.t. $\mathcal{K}$ iff $bed(\mathcal{K}, C \sqsubseteq D) \geq \alpha$.

3 A MILP-based Reasoning Algorithm for $\mathcal{ALCB}$

It has recently been shown that reasoning is undecidable for several fuzzy DLs in the presence of GCIs. This is the case *e.g.* in Łukasiewicz [17] and Product fuzzy DLs [1]. However, this is not the case in Zadeh DLs or in Łukasiewicz DLs with an acyclic TBox [17]. In the following, we will restrict to fuzzy KBs with acyclic TBoxes in classical, Zadeh and Łukasiewicz $\mathcal{ALCB}$. Note that it is enough to study the BED of a concept assertion since $bed(\mathcal{K}, (a,b){:}R) = bed(\mathcal{K}, a{:}\exists R.\{b\})$ and, in our case, $bed(\mathcal{K}, C \sqsubseteq D) = bed(\mathcal{K}, a : \neg C \sqcup D)$ for a new individual a.

Our algorithm starts by applying some tableau rules that decompose complex concept expressions into simpler ones but also generate a system of inequation constraints [31,33]. These inequations have to hold in order to respect the semantics of the DL constructors. After all rules have been applied, an optimisation problem must be solved before obtaining the final solution. This problem has a solution iff the fuzzy KB is consistent. In our case, we will end up with a bounded Mixed Integer Linear Programming [20] (MILP) problem, although in other fuzzy DLs Non Linear optimisation problems can be obtained. A MILP 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. In our case, MILP problems will be a *bounded* with rational variables ranging over $[0, 1]$ and integer variables ranging over $\{0, 1\}$.

We can assume without loss of generality that role assertions $\langle (a,b)\!:\!R, \alpha\rangle$ are replaced by concept assertions $\langle a\!:\!\exists R.\{b\}, \alpha\rangle$, and that concepts are in *Negation Normal Form* (NNF), where the negation only appears before an atomic concept or an individual value restriction. In fact, a fuzzy concept $\neg C$ can be transformed into NNF by recursively applying these definitions: $\mathsf{nnf}(\neg A) = \neg A, \mathsf{nnf}(\neg\top) = \bot, \mathsf{nnf}(\neg\bot) = \top, \mathsf{nnf}(\neg\neg C) = C, \mathsf{nnf}(\neg(C \sqcap D)) = \neg C \sqcup \neg D, \mathsf{nnf}(\neg(C \sqcup D)) = \neg C \sqcap \neg D, \mathsf{nnf}(\neg\forall R.C) = \exists R.\neg C, \mathsf{nnf}(\neg\exists R.C) = \forall R.\neg C, \mathsf{nnf}(\neg\exists R.\{a\}) = \neg\exists R.\{a\}$.

Algorithm 1 shows how to compute $bed(\mathcal{K}, a\!:\!C)$. Essentially, we consider an expression of the form $\langle a\!:\!\neg C, 1-x\rangle$, where x is a $[0,1]$-valued variable; this implies $(a\!:\!C)^{\mathcal{I}} \leq x$. Then, we apply some satisfiability preserving tableau rules and *minimise* the original variable x such that all constraints are satisfied, that is, $bed(a\!:\!C, \mathcal{K}) := \inf x$ such that $\mathcal{K} \cup \{\langle a\!:\!\neg C, 1-x\rangle\}$ is consistent [7].

Our algorithm use *completion-forests* since an ABox might contain individuals with arbitrary roles connecting them. 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 labeled with a set $\mathcal{L}(v)$ containing $\mathcal{ALCB}$ concepts C or expressions of the forms $\{o\}$ and $\neg\{o\}$. If $C \in \mathcal{L}(v)$, we consider a variable $x_{v:C}$ meaning that v is an instance of C to degree greater than or equal to than the value of the variable $x_{v:C}$. A novelty of this algorithm is the fact that if $\{o\} \in \mathcal{L}(v)$ then we consider a binary variable $x_{v:\{o\}}$. The intuition is that $x_{v:\{o\}} = 1$ iff v is interpreted as individual o. Similarly, if $\neg\{o\} \in \mathcal{L}(v)$ then we consider a binary variable $x_{v:\neg\{o\}}$ with the opposite meaning, *i.e.*, $x_{v:\neg\{o\}} = 1$ iff v is *not* interpreted as individual o.
- Each edge $\langle v, w\rangle$ is labeled with a set $\mathcal{L}(\langle v, w\rangle)$ of roles R and if $R \in \mathcal{L}(\langle v, w\rangle)$ then we consider a variable $x_{(v,w):R}$ representing the degree of being $\langle v, w\rangle$ and instance of R.

A node v containing some $\{a\} \in \mathcal{L}(v)$ is called a *nominal node*. Specifically, if $\{a\} \in \mathcal{L}(v)$ then v is called an a-nominal node. Note that for a nominal node v, $\mathcal{L}(v)$ may contain several $\{a_i\}$ and $\neg\{a_j\}$, for $1 \leq i, j \leq n$, where n is the number of individuals occurring in $\mathcal{K}$. For example we can have $\mathcal{L}(v) = \{\{a_1\}, \{a_2\}, \neg\{a_3\}, \neg\{a_4\}\}$. The notion of successor is needed when dealing with edges. A node w is an *R-successor* of node v if $R \in \mathcal{L}(\langle v, w\rangle)$.

We associate to the forest a set $\mathcal{C}_{\mathcal{F}}$ of constraints of the form $l \leq l'$, $l = l'$, $x_i \in [0,1], y_i \in \{0,1\}$, where l, l' are linear expressions using the variables occurring in the forest. $\mathcal{F}$ is then expanded by repeatedly applying the tableau rules in Table 2. Each rule instance is applied at most once, and the rules (N2), (N3) have the lowest priority. This means that they can only be applied when no other rule can be applied, right before solving the MILP problem, and hence the node v has to be one of the a_i.

Let us explain Algorithm 1. Firstly, Line 1 adds to the fuzzy KB the assertion $\langle a\!:\!\neg C, 1-x\rangle$, involving the variable x that will be minimized. Then, there is some preprocessing: lines 2-4 transform the concepts into NNF, lines 5-7 replace the role assertions with equivalent value restrictions, and lines 8-13 initialise the forest from the axioms in the fuzzy KB by creating a nominal node for each individual. Then, Lines 14-17 apply the tableau rules until no other rule can be applied. Next, lines 18-27 introduce some additional constraints stating the range of each variable ($[0,1]$ or $\{0,1\}$). Finally, lines 28-33 return the minimised value of x if the optimisation problem has a solution; or a value 1 otherwise.

Algorithm 1 Computing $bed(a_0 : C_0, \mathcal{K})$ in fuzzy $\mathcal{ALCB}$.

Input: A concept assertion $a_0 : C_0$, a fuzzy KB $\mathcal{K}$
Output: BED of $a_0 : C_0$ with respect to $\mathcal{K}$
1: $\mathcal{K} \leftarrow \mathcal{K} \cup \{\langle a_0 : \mathsf{nnf}(\neg C_0), 1 - x\rangle\}$
2: **for** each concept assertion $\langle a : C, \alpha\rangle \in \mathcal{K}$ **do**
3: $\quad \mathcal{K} \leftarrow (\mathcal{K} \setminus \{\langle a : C, \alpha\rangle\}) \cup \{\langle a : \mathsf{nnf}(C), \alpha\rangle\}$
4: **end for**
5: **for** each role assertion $\langle (a,b) : R, \alpha\rangle \in \mathcal{K}$ **do**
6: $\quad \mathcal{K} \leftarrow (\mathcal{K} \setminus \{\langle (a,b) : R, \alpha\rangle\}) \cup \{\langle a : \exists R.\{b\}, \alpha\rangle\}$
7: **end for**
8: create an empty forest $\mathcal{F}$
9: $\mathcal{C}_{\mathcal{F}} := \emptyset$
10: **for** each individual a occurring in $\mathcal{K}$ **do**
11: create a node v
12: $\mathcal{L}(v) := \mathcal{L}(v) \cup \{\{a\}\}$
13: **end for**
14: **repeat**
15: apply a rule in Table 2 to a node v
16: mark the applied rule as not applicable anymore to node v
17: **until** no more rule can be applied to nodes in $\mathcal{F}$
18: **for** each variable of the form $x_{v:C} \in \mathcal{K}$ **do**
19: **if** the logic is Łukasiewicz or Zadeh $\mathcal{ALCB}$ **then**
20: $\mathcal{C}_{\mathcal{F}} := \mathcal{C}_{\mathcal{F}} \cup \{x_{v:C} \in [0,1]\}$
21: **else**
22: $\mathcal{C}_{\mathcal{F}} := \mathcal{C}_{\mathcal{F}} \cup \{x_{v:C} \in \{0,1\}\}$
23: **end if**
24: **end for**
25: **for** each variable $z \in \mathcal{K}$ of the forms $y, x_{v:\{o\}}, x_{v:\neg\{o\}}$ **do**
26: $\mathcal{C}_{\mathcal{F}} := \mathcal{C}_{\mathcal{F}} \cup \{z \in \{0,1\}\}$
27: **end for**
28: **if** $\mathcal{C}_{\mathcal{F}}$ has a solution **then**
29: solve the optimisation problem on x
30: **return** x
31: **else**
32: **return** 1
33: **end if**

Table 2. Rules for fuzzy $\mathcal{ALCB}$ with an acyclic TBox.

($\top$) IF $\top \in \mathcal{L}(v)$ THEN $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\top} = 1\}$.
($\bot$) IF $\bot \in \mathcal{L}(v)$ THEN $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\bot} = 0\}$.
($\bar{A}$) IF $\neg A \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{A\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:A} \leq 1 - x_{v:\neg A}\}$

($\sqcap$) IF $C_1 \sqcap C_2 \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{C_1, C_2\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:C_1} \otimes x_{v:C_2} \geq x_{v:C_1 \sqcap C_2}\}$

($\sqcup$) IF $C_1 \sqcup C_2 \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{C_1, C_2\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:C_1} \oplus x_{v:C_2} \geq x_{v:C_1 \sqcup C_2}\}$

($\forall$) IF $\forall R.C \in \mathcal{L}(v)$ AND w is an R-successor of v THEN
1. $\mathcal{L}(w) = \mathcal{L}(w) \cup \{C\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\forall R.C} \leq x_{(v,w):R} \Rightarrow x_{w:C}\}$

($\exists$) IF $\exists R.C \in \mathcal{L}(v)$ AND $C \neq \{a\}$ THEN
1. create a new node w
2. $\mathcal{L}(\langle v, w\rangle) = \mathcal{L}(\langle v, w\rangle) \cup \{R\}$
3. $\mathcal{L}(w) = \mathcal{L}(w) \cup \{C\}$
4. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{w:C} \otimes x_{(v,w):R} \geq x_{v:\exists R.C}\}$.

($\sqsubseteq$) IF $\langle A \sqsubseteq C, \alpha\rangle \in \mathcal{T}$ AND $A \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{C\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:A} \Rightarrow x_{v:C} \geq \alpha\}$

($=$) IF $A = C \in \mathcal{T}$ AND $A \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{C\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:A} = x_{v:C}\}$

($\neg =$) IF $A = C \in \mathcal{T}$ AND $\neg A \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{\neg C\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\neg A} = x_{v:\neg C}\}$

(Ass). IF $\langle a{:}C, \alpha\rangle \in \mathcal{K}$ AND v is an a-nominal node THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{C\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\{a\}} \Rightarrow x_{v:C} \geq \alpha\}$

($\forall_o$) IF $\neg\exists R.\{a\} \in \mathcal{L}(v)$ AND w is an R-successor of v THEN
1. $\mathcal{L}(w) = \mathcal{L}(w) \cup \{\neg\{a\}\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\neg\exists R.\{a\}} \leq x_{(v,w):R} \Rightarrow x_{w:\neg\{a\}}\}$.

($\exists_o$) IF $\exists R.\{a\} \in \mathcal{L}(v)$ AND w is an a-nominal node THEN
1. $\mathcal{L}(\langle v, w\rangle) = \mathcal{L}(\langle v, w\rangle) \cup \{R\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{w:\{a\}} \Rightarrow (x_{v:\exists R.\{a\}} \Rightarrow x_{(v,w):R}) \geq 1\}$.

(N1) IF $\neg\{a\} \in \mathcal{L}(v)$ THEN
1. $\mathcal{L}(v) = \mathcal{L}(v) \cup \{\{a\}\}$
2. $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\neg\{a\}} = 1 - x_{v:\{a\}}\}$.

(N2) IF $\{\{a_1\}, \ldots, \{a_n\}\} \subseteq \mathcal{L}(v)$ with $n \geq 2$ AND no other rule can be applied THEN $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v:\{a_1\}} + \ldots + x_{v:\{a_n\}} \leq 1\}$.

(N3) IF $\{a\} \in \mathcal{L}(v_i)$, for $1 \leq i \leq n$ AND no other rule can be applied THEN $\mathcal{C_F} := \mathcal{C_F} \cup \{x_{v_1:\{a\}} + \ldots + x_{v_n:\{a\}} = 1\}$.

Now, let us explain the rationale behind the rules. Rules $(\top), (\bot), (\bar{A}), (\sqcap), (\sqcup), (\forall), (\exists), (\sqsubseteq), (=)$, and $(\neg =)$ have directly been derived from [17,31], so we do not comment them further.

According to the (Ass) rule, given an axiom $\langle a:C, \alpha\rangle$ and an a-nominal node v, we add C to $\mathcal{L}(v)$ and then add the constraint $x_{v:\{a\}} \Rightarrow x_{v:C} \geq \alpha$ to $\mathcal{C}_{\mathcal{F}}$. Since $x_{v:\{a\}}$ is binary, this is equivalent to say that either $x_{v:\{a\}} = 0$ or $x_{v:C} \geq \alpha$. That is, if v is interpreted as a, then v must belong to concept C with at least degree α. Notice that we apply the rule not only to the nominal nodes created during the initialisation phase, but also to all nominal nodes created during the inference process. $(\forall_o)$ and $(\exists_o)$ are similar to the rules managing universal restrictions and existential restrictions. Note that in the $(\exists_o)$ rule, the equation $x_{w:\{a\}} \Rightarrow (x_{v:\exists R.\{a\}} \Rightarrow x_{(v,w):R}) \geq 1$ encodes the fact that if node w is interpreted as individual a then indeed the truth value of $x_{(v,w):R}$ has to be at least the truth value of $x_{v:\exists R.\{a\}}$.

(N1) avoids the inconsistency of having a node both interpreted and not interpreted as the same individual. (N2) deals with the UNA, and guarantees that a nominal node is not interpreted as more than one individual of the $a_i \in \mathcal{L}(v)$. Finally, (N3) makes sure that for every a-nominal node in the forest there is exactly one node interpreted as the individual a.

Concerning the MILP encoding of the fuzzy operators for classical, Łukasiewicz and Zadeh logics, we use the same ideas in *e.g.* [17,33]. Specifically, $x_1 \Rightarrow x_2 \geq z$ can be encoded as (denoted $\mapsto$) $\{1 - x_1 \oplus x_2 \geq z\}$. In Łukasiewicz, $x_1 \oplus x_2 \geq z \mapsto \{x_1 + x_2 \geq z\}$, while $x_1 \otimes x_2 \geq z \mapsto \{y \leq 1 - z, x_1 + x_2 - 1 \geq z - y, y \in \{0, 1\}\}$, where y is a new variable. In Zadeh, $x_1 \otimes x_2 \geq z \mapsto \{x_1 \geq z, x_2 \geq z\}$, while $x_1 \oplus x_2 \geq z \mapsto \{z \geq x_1, z \geq x_2, x_1 + y \geq z, x_2 + (1 - y) \geq z, y \in \{0, 1\}\}$, where y is a new variable. For classical DLs, any of the previous encodings is valid as long as we additionally force x_1, x_2 to be binary. It is convenient to chose those encodings that minimise the number of new variables.

Soundness, completeness, and termination can be proved similarly as in [17]:

Proposition 1. *Given a fuzzy KB $\mathcal{K}$ in Łukasiewicz, Zadeh, and classical $\mathcal{ALCB}$ with an acyclic TBox, Algorithm 1 terminates and correctly computes $bed(\mathcal{K}, a:C)$.*

4 A Running Example

Example 2. Consider the fuzzy KB $\mathcal{K} = \{b:A, c:B, a:\exists R.\{b\} \sqcup \exists R.\{c\}\}$ and let us show (the chosen fuzzy logic is irrelevant) that

$$bed(\mathcal{K}, a:\exists R.A \sqcup \exists R.\{c\}) = 1$$

The axioms of $\mathcal{K}$ are already in NNF and there are no role assertions. Furthermore, to compute $bed(\mathcal{K}, a:\exists R.A \sqcup \exists R.\{c\})$, the axiom

$$\langle a:\forall R.\neg A \sqcap \neg\exists R.\{c\}, 1 - x\rangle \quad (1)$$

is added to $\mathcal{K}$. The forest initialisation creates three nodes v_1, v_2, and v_3 being a b-nominal, c-nominal and a-nominal, respectively.

At first, the (Ass) rule is applied and, thus,

1. A is added to $\mathcal{L}(v_1)$;
2. B is added to $\mathcal{L}(v_2)$;
3. $\exists r.\{b\} \sqcup \exists r.\{c\}$ and $\forall R.\neg A \sqcap \neg\exists R.\{c\}$ are added to $\mathcal{L}(v_3)$;
4. $\mathcal{C}_\mathcal{F}$ contains so far expressions of the form $x_{v:\{a\}} \Rightarrow x_{v:C} \geq \alpha$, namely

$$\begin{array}{l} x_{v_1:\{b\}} \Rightarrow x_{v_1:A} \geq 1 \\ x_{v_2:\{c\}} \Rightarrow x_{v_2:B} \geq 1 \\ x_{v_3:\{a\}} \Rightarrow x_{v_3:\exists R.\{b\}\sqcup\exists R.\{c\}} \geq 1 \\ x_{v_3:\{a\}} \Rightarrow x_{v_3:\forall R.\neg A\sqcap\neg\exists R.\{c\}} \geq 1 - x \ . \end{array}$$

Next, the $(\sqcup)$ is applied to v_3 and, thus,

1. $\exists R.\{b\}$ and $\exists R.\{c\}$ are added to $\mathcal{L}(v_3)$;
2. $x_{v_3:\exists R.\{b\}} \oplus x_{v_3:\exists R.\{c\}} \geq x_{v_3:\exists R.\{b\}\sqcup\exists R.\{c\}}$ is added to $\mathcal{C}_\mathcal{F}$.

Then, the $(\sqcap)$ rule is applied to v_3 and, thus,

1. $\forall R.\neg A$ and $\neg\exists R.\{c\}$ are added to $\mathcal{L}(v_3)$;
2. $x_{v_3:\forall R.\neg A} \otimes x_{v_3:\neg\exists R.\{c\}} \geq x_{v_3:\forall R.\neg A\sqcap\neg\exists R.\{c\}}$ is added to $\mathcal{C}_\mathcal{F}$.

Next, we apply the $(\exists_o)$ rule to $\exists R.\{b\} \in \mathcal{L}(v_3)$ and, thus,

1. R is added to $\mathcal{L}(\langle v_3, v_1\rangle)$;
2. $x_{v_1:\{b\}} \Rightarrow (x_{v_3:\exists R.\{b\}} \Rightarrow x_{(v_3,v_1):R}) \geq 1$ is added to $\mathcal{C}_\mathcal{F}$.

Now we can apply the $(\forall)$ rule to $\mathcal{L}(v_3)$ and, thus,

1. $\neg A$ is added to $\mathcal{L}(v_1)$;
2. $x_{v_3:\forall R.\neg A} \leq x_{(v_3,v_1):R} \Rightarrow x_{v_1:\neg A}$ is added to $\mathcal{C}_\mathcal{F}$.

After that, we can apply the $(\forall_o)$ rule to $\mathcal{L}(v_3)$ and, thus,

1. $\neg\{c\}$ is added to $\mathcal{L}(v_1)$;
2. $x_{v_3:\neg\exists R.\{c\}} \leq x_{(v_3,v_1):R} \Rightarrow x_{v_1:\neg\{c\}}$ is added to $\mathcal{C}_\mathcal{F}$.

Then, the $(N1)$ rule adds $\{c\}$ to $\mathcal{L}(v_1)$ and $x_{v_1:\neg\{c\}} = 1 - x_{v_1:\{c\}}$ to $\mathcal{C}_\mathcal{F}$. Since v_1 is also a c-nominal node now, we may apply the (Ass) rule to it, so

1. B is added to $\mathcal{L}(v_1)$;
2. $x_{v_1:\{c\}} \Rightarrow x_{v_1:B} \geq 1$ is added to $\mathcal{C}_\mathcal{F}$.

The $(\exists_o)$ rule is applied next to $\exists R.\{c\} \in \mathcal{L}(v_3)$ and, thus,

1. R is added to $\mathcal{L}(\langle v_3, v_2\rangle)$;
2. $x_{v_1:\{c\}} \Rightarrow (x_{v_3:\exists R.\{c\}} \Rightarrow x_{(v_3,v_1):R}) \geq 1$ is added to $\mathcal{C}_\mathcal{F}$;
3. $x_{v_2:\{c\}} \Rightarrow (x_{v_3:\exists R.\{c\}} \Rightarrow x_{(v_3,v_2):R}) \geq 1$ is added to $\mathcal{C}_\mathcal{F}$.

As now node v_2 has become an R-successor of v_3, we may apply the $(\forall)$ and $(\forall_o)$ rules to $\mathcal{L}(v_3)$ and, thus,

1. $\neg A$ is added to $\mathcal{L}(v_2)$;
2. $x_{v_3:\forall R.\neg A} \leq x_{(v_3,v_2):R} \Rightarrow x_{v_2:\neg A}$ is added to $\mathcal{C}_\mathcal{F}$;

3. $\neg\{c\}$ is added to $\mathcal{L}(v_2)$;
4. $x_{v_3:\neg\exists R.\{a\}} \leq x_{(v_3,v_2):R} \Rightarrow x_{v_2:\neg\{c\}}$ is added to $\mathcal{C}_{\mathcal{F}}$.

The $(N1)$ rule adds now adds $\{c\}$ to $\mathcal{L}(v_2)$ and $x_{v_2:\neg\{c\}} = 1 - x_{v_2:\{c\}}$ to $\mathcal{C}_{\mathcal{F}}$. Eventually, rule $(N2)$ and $(N3)$ add the following constraints to $\mathcal{C}_{\mathcal{F}}$:

1. $x_{v_1:\{b\}} + x_{v_1:\{c\}} \leq 1$;
2. $x_{v_3:\{a\}} = 1$;
3. $x_{v_1:\{b\}} = 1$;
4. $x_{v_1:\{c\}} + x_{v_2:\{c\}} = 1$.

Finally, we have to determine the minimal value of x such that $\mathcal{C}_{\mathcal{F}}$ is satisfiable. Therefore, from the last equations, we get immediately that $x_{v_1:\{b\}} = 1$, $x_{v_3:\{a\}} = 1, x_{v_1:\{c\}} = 0$, and $x_{v_2:\{c\}} = 1$. Of course, then $x_{v_2:\neg\{c\}} = 0$ and $x_{v_1:\neg A} = 0$. The assertion $a{:}\exists R.\{b\} \sqcup \exists R.\{c\}$ implies that $x_{v_3:\exists R.\{b\}} = 1$ or $x_{v_3:\exists R.\{c\}} = 1$. If $x_{v_3:\exists R.\{b\}} = 1$ then from $x_{v_1:\{b\}} \Rightarrow (x_{v_3:\exists R.\{b\}} \Rightarrow x_{(v_3,v_1):R}) \geq 1$ we get that $x_{(v_3,v_1):R} = 1$, while if $x_{v_3:\exists R.\{c\}} = 1$ from $x_{v_2:\{c\}} \Rightarrow (x_{v_3:\exists R.\{c\}} \Rightarrow x_{(v_3,v_2):R}) \geq 1$ we get that $x_{(v_3,v_2):R} = 1$. Now, suppose $x < 1$. From Equation 1, $x_{v_3:\forall R.\neg A \sqcap \neg\exists R.\{c\}} > 0$, and thus $x_{v_3:\forall R.\neg A} > 0$ and $x_{v_3:\neg\exists R.\{c\}} > 0$. If $x_{(v_3,v_1):R} = 1$, we get a contradiction with $x_{v_1:\neg A} = 0$, while if $x_{(v_3,v_2):R} = 1$, we get a contradiction with $x_{v_2:\neg\{c\}} = 0$. However, for $x = 1$ $\mathcal{C}_{\mathcal{F}}$ is satisfiable, so our algorithm returns $x = 1$. Indeed, $bed(\mathcal{K}, a{:}\exists R.A \sqcup \exists R.\{c\}) = 1$ holds.

5 Conclusions and Future Work

In this paper we have proposed the first algorithm based on OR to reason with fuzzy DLs including individual value restrictions. In particular, we have proposed an algorithm to compute the BED in classical, Zadeh, and Łukasiewicz $\mathcal{ALCB}$ if the TBox is acyclic. The algorithm could be extended to the case of general TBoxes using blocking. A relevant property of the algorithm is that only one optimisation problem has to be solved, deferring the inherent non-determinism in $\mathcal{ALCB}$ to the optimisation problem solver.

Future work will include the extension of the algorithm to more expressive fuzzy DLs, such as $\mathcal{SHIFB}(\mathbf{D})$. Up to know, $\mathcal{SHIF}(\mathbf{D})$ is current the most expressive fuzzy DL for which there is an implementation of an OR-algorithm (the *fuzzyDL*system). On the one hand, adding inverse roles to the language seems challenging because of the possibility of having inverse functional role axioms. On the other hand, it would also be interesting to have unrestricted nominals to arbitrarily form complex concept expressions. Another possible extension is considering fuzzy nominals of the form $\{\alpha/o\}$ that can be used to describe fuzzy sets by enumeration of their elements [4].

Last but not least, we are aware that the current proposal is not optimal in the handling of nominal nodes, as several a-nominal nodes may occur in a completion-forest. We plan to adapt and implement the well-known *node merging* technique developed for classical DLs, such as for $\mathcal{SHOIQ}$ [24], to the fuzzy case as well, analyse the overall computational complexity and experiment the algorithm within *fuzzyDL*.

References

1. F. Baader, S. Borgwardt, and R. Peñaloza. On the decidability status of fuzzy $\mathcal{ALC}$ with general concept inclusions. *Journal of Philosophical Logic*, In press.
2. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.
3. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *The Scientific American*, 284(5):34–43, 2001.
4. F. Bobillo, M. Delgado, and J. Gómez-Romero. Crisp representations and reasoning for fuzzy ontologies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 17(4):501–530, 2009.
5. F. Bobillo, M. Delgado, J. Gómez-Romero, and U. Straccia. Fuzzy description logics under Gödel semantics. *International Journal of Approximate Reasoning*, 50(3):494–514, 2009.
6. F. Bobillo, M. Delgado, J. Gómez-Romero, and U. Straccia. Joining Gödel and Zadeh fuzzy logics in fuzzy description logics. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 20:475–508, 2012.
7. F. Bobillo and U. Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *Proceedings of the 17th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2008)*, pages 923–930, June 2008.
8. F. Bobillo and U. Straccia. Fuzzy description logics with general t-norms and datatypes. *Fuzzy Sets and Systems*, 160(23):3382–3402, 2009.
9. F. Bobillo and U. Straccia. Fuzzy ontology representation using OWL 2. *International Journal of Approximate Reasoning*, 52:1073–1094, 2011.
10. F. Bobillo and U. Straccia. Reasoning with the Łukasiewicz fuzzy description logic $\mathcal{SROIQ}$. *Information Sciences*, 181:758–778, 2011.
11. F. Bobillo and U. Straccia. Aggregation operators for fuzzy ontologies. *Applied Soft Computing*, 13:3816–3830, 2013.
12. S. Borgwardt, F. Distel, and R. Peñaloza. How fuzzy is my fuzzy description logic? In B. Gramlich, D. Miller, and U. Sattler, editors, *Proceedings of the 6th International Joint Conference on Approximated Reasoning (IJCAR 2012)*, volume 7364 of *Lecture Notes in Artificial Intelligence*, pages 82–96. Springer-Verlag, 2012.
13. S. Borgwardt, F. Distel, and R. Peñaloza. Decidable Gödel description logics without the finitely-valued model property. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR 2014)*. AAAI Press, 2014.
14. S. Borgwardt and R. Peñaloza. The complexity of lattice-based fuzzy description logics. *Journal on Data Semantics*, 2(1):1–19, 2013.
15. S. Borgwardt and R. Peñaloza. Consistency reasoning in lattice-based fuzzy description logics. *International Journal of Approximate Reasoning*, In press.
16. M. Cerami, A. García-Cerdaña, and F. Esteva. On finitely-valued fuzzy description logics. *International Journal of Approximate Reasoning*, In press.
17. M. Cerami and U. Straccia. On the (un)decidability of fuzzy Description Logics under Łukasiewicz t-norm. *Information Sciences*, 227:1–21, 2013.
18. B. Cuenca-Grau, I. Horrocks, B. Motik, B. Parsia, P. F. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.
19. A. García-Cerdaña, E. Armengol, and F. Esteva. Fuzzy Description Logics and t-norm based fuzzy logics. *International Journal of Approximate Reasoning*, 51(6):632–655, 2010.

20. M. K. H. Salkin. *Foundations of Integer Programming*. North-Holland, 1988.
21. P. Hájek. *Metamathematics of Fuzzy Logic*. Kluwer, 1998.
22. P. Hájek. Making fuzzy description logic more general. *Fuzzy Sets and Systems*, 154(1):1–15, 2005.
23. I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006.
24. I. Horrocks and U. Sattler. A tableau decision procedure for $\mathcal{SHOIQ}$. *Journal of Automated Reasoning*, 39(3):249–276, 2007.
25. T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Journal of Web Semantics*, 6:291–308, 2008.
26. T. P. Mailis, G. Stoilos, N. Simou, G. B. Stamou, and S. D. Kollias. Tractable reasoning with vague knowledge using fuzzy $\mathcal{EL}^{++}$. *Journal of Intelligent Information Systems*, 39:399–440, 2012.
27. P. S. Mostert and A. L. Shields. On the structure of semigroups on a compact manifold with boundary. *Annals of Mathematics*, 65(1):117–143, 1957.
28. G. Stoilos and G. Stamou. Reasoning with fuzzy extensions of OWL and OWL 2. *Knowledge and Information Systems*, In press.
29. G. Stoilos, G. Stamou, J. Z. Pan, V. Tzouvaras, and I. Horrocks. Reasoning with very expressive fuzzy description logics. *Journal of Artificial Intelligence Research*, 30(8):273–320, 2007.
30. U. Straccia. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.
31. U. Straccia. Description logics with fuzzy concrete domains. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI 2005)*, pages 559–567. AUAI Press, 2005.
32. U. Straccia. A fuzzy description logic for the semantic web. In E. Sanchez, editor, *Fuzzy Logic and the Semantic Web*, Capturing Intelligence, chapter 4, pages 73–90. Elsevier, 2006.
33. U. Straccia. *Foundations of Fuzzy Logic and Semantic Web Languages*. CRC Studies in Informatics Series. Chapman & Hall, 2013.
34. J. Yen. Generalizing term subsumption languages to fuzzy logic. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI 1991)*, pages 472–477, Sydney, Australia, 1991.
35. L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.