General Concept Inclusion Absorptions for Fuzzy Description Logics: A First Step

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. General Concept Inclusion (GCIs) absorption algorithms have shown to play an important role in classical Description Logics (DLs) reasoners, as they allow to transform GCIs into simpler forms to which apply specialised inference rules, resulting in an important performance gain. In this work, we develop a first absorption algorithm for fuzzy DLs, and evaluate it over some ontologies.

1 Introduction

GCI Absorption is a technique that allows to transform General Concept Inclusion axioms (GCIs) into simpler forms to which apply specialised inference rules [1, 11, 12, 18–20, 24], such as the so-called *lazy unfolding rules*.

While absorption algorithms have shown to provide an important performance gain for classical DLs reasoners, no such algorithms have been developed so far in the context of *fuzzy* DLs [14], *i.e.*, DLs in which the truth of axioms may not be bivalent, but graded instead, *e.g.*, a truth value in $[0, 1]$. Another benefit may consist in the posssibility to transform a non acyclic knowledge base into an acyclic one. This is important from a computational point of view, as *e.g.*, the satisfiability problem under Łukasiewicz logic is correct and complete for acyclic TBoxes, while the general GCIs case is undecidable [5] (see [4] for more undecidability results). Also note that the absorption method holds for the case of finite linearly-ordered truth space as well, for which the satisfiability problem is decidable if the same problem for the corresponding classical DL is. Therefore, the absorptions can be beneficial for finitely-valued DLs as well.

The aim of this paper is to work out an absorption algorithm for fuzzy DLs, and evaluate it over some ontologies using the *fuzzyDL* reasoner [2][3]. In the following, we first recap some basic fuzzy DLs definitions, then in Section 3 we illustrate our GCI absorption algorithm, while Section 4 provides an evaluation of it. Eventually, Section 5 concludes and illustrates some future research directions.

2 Fuzzy DLs Basics

We recap here some basic definitions we rely on. We refer the reader to *e.g.*, [14], for a more in depth presentation.

[3] http://www.straccia.info/software/fuzzyDL/fuzzyDL.html

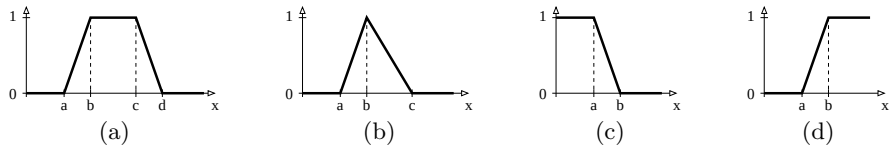


Fig. 1. (a) Trapezoidal function $trz(a, b, c, d)$, (b) triangular function $tri(a, b, c)$, (c) left shoulder function $ls(a, b)$, and (d) right shoulder function $rs(a, b)$.

Mathematical Fuzzy Logic. Fuzzy Logic is the logic of fuzzy sets. A *fuzzy set* R over a countable crisp set X is a function $R\colon X \to [0, 1]$. The trapezoidal (Fig. 1 (a)), the triangular (Fig. 1 (b)), the L-function (left-shoulder function, Fig. 1 (c)), and the R-function (right-shoulder function, Fig. 1 (d)) are frequently used to specify membership functions of fuzzy sets.

Although fuzzy sets have a far greater expressive power than classical crisp sets, its usefulness depends critically on the capability to construct appropriate membership functions for various given concepts in different contexts. The problem of constructing meaningful membership functions is a difficult one and we refer the interested reader to, *e.g.*, [13, Chapter 10]. However, one easy and typically satisfactory method to define the membership functions is to uniformly partition the range of, *e.g.*, salary values (bounded by a minimum and maximum value), into 5 or 7 fuzzy sets using either trapezoidal functions (*e.g.*, as illustrated on the left in Figure 2), or using triangular functions (as illustrated on the right in Figure 2). The latter is the more used one, as it has less parameters.

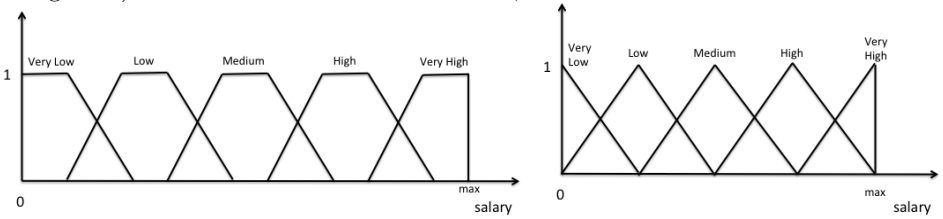


Fig. 2. Fuzzy sets over salaries using trapezoidal or triangular functions.

In *Mathematical Fuzzy Logic* [9], the 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 or equal* α. Usually, the *truth space* is $L = [0, 1]$. Another popular truth space is the finite truth space $L_n = \{0, \frac{1}{n-1}, \ldots, \frac{n-2}{n-1}, 1\}$ for some natural number $n > 1$. Of course, L_2 is the usual classical two-valued case.

A *fuzzy interpretation* $\mathcal{I}$ maps each atomic statement p_i into $[0, 1]$ and is then extended inductively to all statements: $\mathcal{I}(\phi \wedge \psi) = \mathcal{I}(\phi) \otimes \mathcal{I}(\psi)$, $\mathcal{I}(\phi \vee \psi) = \mathcal{I}(\phi) \oplus \mathcal{I}(\psi)$, $\mathcal{I}(\phi \rightarrow \psi) = \mathcal{I}(\phi) \Rightarrow \mathcal{I}(\psi)$, $\mathcal{I}(\neg\phi) = \ominus\mathcal{I}(\phi)$, $\mathcal{I}(\exists x.\phi(x)) = \sup_{y \in \Delta^{\mathcal{I}}} \mathcal{I}(\phi(y))$, $\mathcal{I}(\forall x.\phi(x)) = \inf_{y \in \Delta^{\mathcal{I}}} \mathcal{I}(\phi(y))$, where $\Delta^{\mathcal{I}}$ is the domain of $\mathcal{I}$, and $\otimes$, $\oplus$, $\Rightarrow$, and $\ominus$ are so-called *t-norms*, *t-conorms*, *implication functions*, and *negation functions*, respectively, which extend the Boolean conjunction, disjunction, implication, and negation, respectively, to the fuzzy case.

One usually distinguishes three different logics, namely Łukasiewicz, Gödel, and Product logics [9][4], with the following combination functions:

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

We will also use an optional subscript $X \in \{l, g, p\}$ to identify the logic they refer to (*e.g.*, $\alpha \otimes_g \beta$ refers to Gödel conjunction). Note that the operators for Zadeh logic, namely $\alpha \otimes \beta = \min(\alpha, \beta)$, $\alpha \oplus \beta = \max(\alpha, \beta)$, $\ominus \alpha = 1 - \alpha$ and $\alpha \Rightarrow \beta = \max(1 - \alpha, \beta)$, can be expressed in Łukasiewicz logic. More precisely, $\min(\alpha, \beta) = \alpha \otimes_l (\alpha \Rightarrow_l \beta), \max(\alpha, \beta) = 1 - \min(1 - \alpha, 1 - \beta)$. Furthermore, the implication $\alpha \Rightarrow_{kd} \beta = \max(1 - \alpha, b)$ is called *Kleene-Dienes implication* (denoted $\Rightarrow_{kd}$), while *Zadeh implication* (denoted $\Rightarrow_z$) is the implication $\alpha \Rightarrow_z \beta = 1$ if $\alpha \leq \beta$; 0 otherwise.

An *r-implication* is an implication function obtained as the residuum of a continuous t-norm $\otimes$ *i.e.*, $\alpha \Rightarrow \beta = \max\{\gamma \mid \alpha \otimes \gamma \leq \beta\}$. Note that Łukasiewicz, Gödel and Product implications are r-implications, while Kleene-Dienes implication is not. Note also, that given an r-implication $\Rightarrow_r$, we may also define its related negation $\ominus_r \alpha$ by means of $\alpha \Rightarrow_r 0$ for every $\alpha \in [0, 1]$.

Some additional properties of truth combination functions are the following:

Property	Łukasiewicz logic	Gödel	Product	Zadeh [23]
$\alpha \otimes \ominus \alpha = 0$	•	•	•	
$\alpha \oplus \ominus \alpha = 1$	•			
$\alpha \otimes \alpha = \alpha$		•		•
$\alpha \oplus \alpha = \alpha$		•		•
$\ominus \ominus \alpha = \alpha$	•			•
$\alpha \Rightarrow \beta = \ominus \alpha \oplus \beta$	•			•
$\alpha \Rightarrow \beta = \ominus \beta \Rightarrow \ominus \alpha$	•			•
$\ominus (\alpha \Rightarrow \beta) = \alpha \otimes \ominus \beta$	•			•
$\ominus (\alpha \otimes \beta) = \ominus \alpha \oplus \ominus \beta$	•	•	•	•
$\ominus (\alpha \oplus \beta) = \ominus \alpha \otimes \ominus \beta$	•	•	•	•
$\alpha \otimes (\beta \oplus \gamma) = (\alpha \otimes \beta) \oplus (\alpha \otimes \gamma)$		•		•
$\alpha \oplus (\beta \otimes \gamma) = (\alpha \oplus \beta) \otimes (\alpha \oplus \gamma)$		•		•

The notions of satisfiability and logical consequence are defined in the standard way, where a fuzzy interpretation $\mathcal{I}$ *satisfies* a fuzzy statement $\langle \phi, \alpha \rangle$ or $\mathcal{I}$ is a *model* of $\langle \phi, \alpha \rangle$, denoted as $\mathcal{I} \models \langle \phi, \alpha \rangle$, iff $\mathcal{I}(\phi) \geq \alpha$.

Fuzzy $\mathcal{ALC}(\mathbf{D})$ basics. We recap here the fuzzy variant of the DL $\mathcal{ALC}(\mathbf{D})$ [17].

A *fuzzy concrete domain* or *fuzzy datatype theory* $\mathbf{D} = \langle \Delta^{\mathbf{D}}, \cdot^{\mathbf{D}} \rangle$ consists of a datatype domain $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each data value an element of $\Delta^{\mathbf{D}}$, and to every n-ary datatype predicate d an n-ary fuzzy relation over $\Delta_{\mathbf{D}}$. We will restrict to unary datatypes as usual in fuzzy DLs. Therefore, $\cdot^{\mathbf{D}}$ maps indeed each datatype predicate into a function from $\Delta^{\mathbf{D}}$ to $[0, 1]$. Typical examples of datatype predicates $\mathbf{d}$ are the well known membership functions

[4] The main reason is that any other t-norm can be obtained as a combination of them.

$$\mathbf{d} := ls(a,b) \mid rs(a,b) \mid tri(a,b,c) \mid trz(a,b,c,d) \mid \geq_v \mid \leq_v \mid =_v ,$$

where *e.g.*, $ls(a,b)$ is the left-shoulder membership function and $\geq_v$ corresponds to the crisp set of data values that are greater or equal than the value v.

Now, let $\mathbf{A}$ be a set of *concept names* (also called atomic concepts), $\mathbf{R}$ be a set of *role names*. Each role is either an *object property* or a *datatype property*. The set of *concepts* are built from concept names A using connectives and quantification constructs over object properties R and datatype properties T, as described by the following syntactic rules:

$$C \to \top \mid \bot \mid A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \neg C \mid C_1 \to C_2 \mid \exists R.C \mid \forall R.C \mid \exists T.\mathbf{d} \mid \forall T.\mathbf{d} .$$

Each of the connectives $\sqcap$ and $\sqcup$ may also have a subscript $X \in \{g, l, p\}$, $\to$ may have a subscript $X \in \{kd, g, l, p, z\}$ and, $\neg$ may have a subscript $Y \in \{g, l\}$. The subscript indicates the fuzzy logic the connectives refers to (see Section 2). For instance, $C \sqcap (D \to_l \forall R.\neg_g E)$ is a concept (if a subscript is missing, then we assume that a a priori selected fuzzy logic $X \in \{g, l, p, z\}$ has been selected).

An *ABox* $\mathcal{A}$ consists of a finite set of assertions of the forms $\langle a{:}C, \alpha\rangle$ (meaning that a is an instance of C to degree at least α), or $\langle (a,b){:}R, \alpha\rangle$ (meaning that a and b are related via R with degree degree at least α), where a, b are individual names, C is a concept, R is a role name and $\alpha \in (0, 1]$ is a truth value.

A *Terminological Box* or *TBox* $\mathcal{T}$ is a finite set of GCIs and constraints. A *General Concept Inclusion* (GCI) axiom is of the form $\langle C_1 \sqsubseteq C_2, \alpha\rangle$ (C_1 is a sub-concept of C_2 to degree at least α), where C_i is a concept and $\alpha \in (0, 1]$. A *primitive* GCI is one of the form $\langle A \sqsubseteq C, \alpha\rangle$, where A is a concept name and C is a concept. In both cases above, $\sqsubseteq$ may also have a subscript $X \in \{g, l, p, z\}$. Note that $\sqsubseteq_{kd}$ is not allowed. A *definitional* GCI is one of the form $A \doteq C$. A is called the *head* of a primitive/definitional axiom, and C is the *body*. A *synonym* GCI is of the form $A \doteq B$, where both A and B are concept names. A *generalised definitional* GCI is of the form $C \doteq D$, where both C and D are concepts.

A *constraint* axiom has one of the following form (R is an object property): *(i)* $\mathsf{domain}(R, C)$, called *domain restriction*, that restricts the domain of role R to be concept C; *(ii)* $\mathsf{range}(R, C)$, called *range restriction*, that restricts the range of role R to be concept C; and *(iii)* $\mathsf{disjoint}(A, B)$, called *disjoint restriction*, that restricts the concept names A and B to be disjoint.

We may omit the truth degree α of an axiom; in this case $\alpha = 1$ is assumed.

A *knowledge base* is a pair $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}\rangle$, where $\mathcal{A}$ is an ABox and $\mathcal{T}$ is a TBox.

Let $\mathcal{T}$ be a TBox consisting of definitional GCIs only. Concept name A *directly uses* concept name B w.r.t. $\mathcal{T}$, if A is the head of some axiom $\tau \in \mathcal{T}$ such that B occurs in the body of τ. Let *uses* be the transitive closure of the relation directly uses. $\mathcal{T}$ is *acyclic* if no concept name A is the head of more than one definitional axiom in $\mathcal{T}$ and there is no concept name A such that A uses A.

Concerning the semantics, let us fix a fuzzy logic $X \in \{l, g, p, z\}$. Unlike classical DLs in which 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 $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to \{0, 1\}$ (either an individual belongs to the extension of C or does not belong to it), in fuzzy

DLs, $\mathcal{I}$ maps C into a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0,1]$ 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 nonempty (crisp) set $\Delta^{\mathcal{I}}$ (the *domain*) and of a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$ that assigns: *(i)* to each atomic concept A a function $A^{\mathcal{I}}\colon \Delta^{\mathcal{I}} \to [0,1]$; *(ii)* to each object property R a function $R^{\mathcal{I}}\colon \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1]$; *(iii)* to each data type property T a function $T^{\mathcal{I}}\colon \Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}} \to [0,1]$; *(iv)* to each individual a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$; and *(v)* to each concrete value v an element $v^{\mathcal{I}} \in \Delta^{\mathbf{D}}$.

Now, a fuzzy interpretation function is extended to concepts as specified below (where $x, y \in \Delta^{\mathcal{I}}$ are elements of the domain), so for every concept C we get a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0,1]$:

$$\bot^{\mathcal{I}}(x) = 0, \quad \top^{\mathcal{I}}(x) = 1,$$
$$(C \sqcap D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \otimes D^{\mathcal{I}}(x), \quad (C \sqcap_X D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \otimes_X D^{\mathcal{I}}(x),$$
$$(C \sqcup D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \oplus D^{\mathcal{I}}(x), \quad (C \sqcup_X D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \oplus_X D^{\mathcal{I}}(x),$$
$$(\neg C)^{\mathcal{I}}(x) = \ominus C^{\mathcal{I}}(x), (\neg_X C)^{\mathcal{I}}(x) = \ominus_X C^{\mathcal{I}}(x),$$
$$(C \to D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x), \quad (C \to_X D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \Rightarrow_X D^{\mathcal{I}}(x),$$
$$(\forall R.C)^{\mathcal{I}}(x) = \inf_{y \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)\}, (\exists R.C)^{\mathcal{I}}(x) = \sup_{y \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)\},$$
$$(\forall T.\mathbf{d})^{\mathcal{I}}(x) = \inf_{y \in \Delta^{\mathcal{I}}} \{T^{\mathcal{I}}(x,y) \Rightarrow \mathbf{d}^{\mathbf{D}}(y)\}, \ (\exists T.\mathbf{d})^{\mathcal{I}}(x) = \sup_{y \in \Delta^{\mathcal{I}}} \{T^{\mathcal{I}}(x,y) \otimes \mathbf{d}^{\mathbf{D}}(y)\} \ .$$

The *satisfiability of axioms* is then defined by the following conditions: *(i)* $\mathcal{I}$ satisfies an axiom $\langle a{:}C, \alpha\rangle$ if $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq \alpha$; *(ii)* $\mathcal{I}$ satisfies an axiom $\langle (a,b){:}R, \alpha\rangle$ if $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq \alpha$; *(iii)* $\mathcal{I}$ satisfies an axiom $\langle C \sqsubseteq D, \alpha\rangle$ if $(C \sqsubseteq D)^{\mathcal{I}} \geq \alpha$ where[5] $(C \sqsubseteq D)^{\mathcal{I}} = \inf_{x \in \Delta^{\mathcal{I}}} \{C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x)\}$; *(iv)* $\mathcal{I}$ satisfies an axiom $\langle C \sqsubseteq_X D, \alpha\rangle$ if $(C \sqsubseteq_X D)^{\mathcal{I}} \geq \alpha$ where $(C \sqsubseteq_X D)^{\mathcal{I}} = \inf_{x \in \Delta^{\mathcal{I}}} \{C^{\mathcal{I}}(x) \Rightarrow_X D^{\mathcal{I}}(x)\}$; *(v)* $\mathcal{I}$ satisfies an axiom $\mathsf{domain}(R, C)$ if $\mathcal{I}$ satisfies $\exists R.\top \sqsubseteq C$; *(vi)* $\mathcal{I}$ satisfies an axiom $\mathsf{range}(R, C)$ if $\mathcal{I}$ satisfies $\top \sqsubseteq \forall R.C$; and *(vii)* $\mathcal{I}$ satisfies an axiom $\mathsf{disjoint}(A, B)$ if $\mathcal{I}$ satisfies $A \sqcap B \sqsubseteq \bot$.

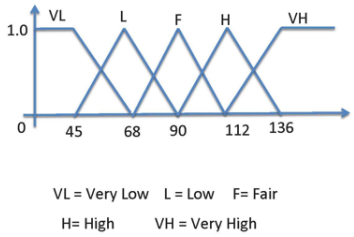
$\mathcal{I}$ is a model of a $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}\rangle$ iff $\mathcal{I}$ satisfies each axiom in $\mathcal{K}$. We say that $\mathcal{K}$ *entails* and axiom α, denoted $\mathcal{K} \models \alpha$, if any model of $\mathcal{K}$ satisfies α. Two TBoxes $\mathcal{T}, \mathcal{T}'$ are equivalent (denoted $\mathcal{T} \equiv \mathcal{T}'$) iff they entail the same set of axioms.

Example 1. We have built a fuzzy wine ontology[6], according the FuzzyOWL 2 proposal[7]. One of the GCIs in there is of the form $SparklingWine \sqcap (\exists hasSugar.\ tri(32, 41, 50)) \sqsubseteq DemiSecSparklingWine$ where $hasSugar$ is a datatype property whose values are measured in g/L.

Example 2. fuzzyDL-Learner[8] is a system that illustrates how one may learn graded GCIs. For instance, consider the case of hotel finding in a possible tourism application, where an ontology is used to describe the meaningful entities of the domain. Now, one may fix a city, say Pisa, extract the characteristic of the hotels from Web sites and the graded hotel judgements of the users, e.g., from Trip

[5] However, note that under Zadeh logic $\sqsubseteq$ is interpreted as $\Rightarrow_z$ and not as $\Rightarrow_{kd}$.
[6] `http://www.straccia.info/software/FuzzyOWL/ontologies/FuzzyWine.1.0.owl`
[7] `http://www.straccia.info/software/FuzzyOWL`
[8] `http://straccia.info/software/FuzzyDL-Learner`

Advisor[9] and asks about what characterises good hotels. Then one may learn that, *e.g.*, $\langle \exists hasPrice.High \sqsubseteq GoodHotel, 0.569 \rangle$ where $hasPrice$ is a datatype property whose values are measured in euros and the price concrete domain has been automatically fuzzified as illustrated below.



Now, it can be verified that for hotel *verdi*, whose room price is 105 euro, *i.e.*, we have the assertion $verdi{:}\exists hasPrice.{=_{105}}$ in the KB, we infer under Product logic that $\mathcal{K} \models \langle verdi{:}GoodHotel, 0.18 \rangle$ (note that $0.18 = 0.318 \cdot 0.569$, where $0.318 = tri(90, 112, 136)(105)$).

3 GCI Absorptions for Fuzzy DLs

The aim of our GCI absorption algorithm is to build from a TBox $\mathcal{T}$ an equivalent TBox $\mathcal{T}'$, which is the union of two disjoint sets of axioms $\mathcal{T}_g$ and $\mathcal{T}_u$, where:

1. $\mathcal{T}_g$ is a set of GCIs of the form $\langle \top \sqsubseteq C, \alpha \rangle$,
2. $\mathcal{T}_u = T_{def} \cup \mathcal{T}_{inc} \cup \mathcal{T}_{dr} \cup \mathcal{T}_{disj} \cup \mathcal{T}_{syn}$ is the disjoint union of
   (a) T_{def}, which is acyclic and contains definitional GCIs only;
   (b) T_{inc}, which contains primitive GCIs only;
   (c) T_{dr}, which contains domain and range restrictions only;
   (d) T_{disj}, which contains disjoint restrictions only;
   (e) T_{syn}, which contains synonym GCIs only; and
3. there cannot be a concept name A that is a head of axioms in T_{def} and T_{inc}.

This partitioning makes it possible to apply lazy unfolding rules to axioms in $\mathcal{T}_u$ only. Now, we explain now how to compute it: Sections 3.1–3.4 present some auxiliary steps and then Section 3.5 describes the partitioning algorithm.

The transformations, if left unspecified, hold always, *i.e.*, for connectives taken from the same logic; for the other cases, the semantics under which they hold is specified explicitly. The transformations are semantics preserving in the sense that they transform a TBox $\mathcal{T}$ into an equivalent TBox $\mathcal{T}'$.

3.1 Concept Simplifications

The procedure $Simplify(C)$ performs the following concept simplifications, replacing the left-hand part with the right-hand part. The simplifications are applied recursively considering that $\sqcap, \sqcup$ are n-ary, *commutative* and *associative*:

[9] `http://www.tripadvisor.com`

1. $C \sqcap \bot \mapsto \bot$
2. $C \sqcap \top \mapsto C$
3. $C \sqcap_G C \mapsto C$, and for classical logic
4. $C \sqcup_G C \mapsto C$, and for classical logic
5. $C \sqcup \bot \mapsto C$
6. $C \sqcup \top \mapsto \top$
7. $\exists R.\bot \mapsto \bot$
8. $\forall R.\top \mapsto \top$
9. $\neg\top \mapsto \bot$
10. $\neg\bot \mapsto \top$, and for classical logic
11. $\neg_l\neg_l C \mapsto C$, and for classical logic
12. $\neg(C \sqcap D) \mapsto \neg C \sqcup \neg D$
13. $\neg(C \sqcup D) \mapsto \neg C \sqcap \neg D$
14. $\neg\forall R.C \mapsto \exists R.\neg C$, for Zadeh, Łukasiewicz, and classical logic
15. $\neg\exists R.C \mapsto \forall R.\neg C$, for Zadeh, Łukasiewicz, and classical logic
16. $C_1 \rightarrow C_2 \mapsto \neg C_1 \sqcup C_2$, for Zadeh, Łukasiewicz, and classical logic
17. $\neg(C_1 \rightarrow C_2) \mapsto C_1 \sqcap \neg C_2$, for Zadeh, Łukasiewicz, and classical logic
18. $C \sqcap \neg C \mapsto \bot$, for Łukasiewicz, Gödel , Product and classical logic
19. $C \sqcup \neg C \mapsto \top$, for Łukasiewicz, and classical logic
20. $C \sqcap (C \sqcup D) \mapsto C$, for Gödel, Zadeh, and for classical logic
21. $C \sqcup (C \sqcap D) \mapsto C$, for Gödel, Zadeh, and for classical logic
22. $C \sqcap (\neg C \sqcup D) \mapsto C \sqcap D$, for Gödel, and for classical logic
23. $C \sqcup (\neg C \sqcap D) \mapsto C \sqcup D$, for Gödel, and for classical logic
24. $\forall R.C \sqcap_G \forall R.D \mapsto \forall R.(C \sqcap_G D)$, and for classical logic
25. $\exists R.C \sqcup \exists R.D \mapsto \exists R.(C \sqcup D)$, for Gödel, Zadeh, and for classical logic
26. $(C \sqcup_G D) \rightarrow E \mapsto (C \rightarrow E) \sqcap_G (D \rightarrow E)$, and for classical logic
27. $C \sqcup_G (D \sqcap_G E) \mapsto (C \sqcup_G D) \sqcap_G (C \sqcup_G E)$, and for classical logic
28. $\exists R.C \sqcap_G \exists R.\top \mapsto \exists R.C$, and for classical logic
29. $C \sqcap (D \sqcap E) \mapsto C \sqcap D \sqcap E$
30. $C \sqcup (D \sqcup E) \mapsto C \sqcup D \sqcup E$
31. $C \sqcap (D \sqcup_G E) \mapsto (C \sqcap D) \sqcup_G (C \sqcap E)$
32. $C \sqcup (D \sqcap_G E) = (C \sqcap D) \sqcap_G (C \sqcap E)$
33. $C \rightarrow \top \mapsto \top$
34. $\top \rightarrow C \mapsto C$
35. $\bot \rightarrow C \mapsto \top$
36. $C \rightarrow_r \bot \mapsto \neg_r C$, where $\rightarrow_r$ is an r-implication

3.2 Redundant GCIs Elimination

The following axioms can safely be removed, since they are trivially satisfied:

1. $\langle \bot \sqsubseteq C, \alpha\rangle$, $\langle C \sqsubseteq \top, \alpha\rangle$ and $C = C$
2. $\langle C \sqsubseteq C, \alpha\rangle$, for any r-implication and Zadeh implication
3. $\langle \sqcap_{i=1}^{n} C_i \sqsubseteq A, l\rangle$, if some C_i is A and $\sqsubseteq$ is an r-implication or Zadeh implication.
4. $\langle A \sqsubseteq \sqcup_{i=1}^{n} C_i, l\rangle$, if some C_i is A and $\sqsubseteq$ is an r-implication or Zadeh implication.

3.3 Role Absorptions

Basic Role Absorption. As in the classical case, $\top \sqsubseteq \forall R.C$ and $\exists R.\top \sqsubseteq C$ are equivalent to $\mathsf{range}(R, C)$ and $\mathsf{domain}(R, C)$, respectively, so we have the following rules [18] to transform these GCIs into domain and range restrictions:

(RB1) Replace any GCI $\exists R.\top \sqsubseteq C \in \mathcal{T}$ with $\mathsf{domain}(R, C)$
(RB2) Replace any GCI $\top \sqsubseteq \forall R.C \in \mathcal{T}$ with $\mathsf{range}(R, C)$

Extended Role Absorptions. In the classical case, $\exists R.C \sqsubseteq D$ can be replaced with $\mathsf{domain}(R, \exists R.C \rightarrow D)$ and $D \sqsubseteq \forall R.C$ can be replaced with $\mathsf{domain}(R, \exists R.\neg C \rightarrow \neg D)$, where $C \rightarrow D$ is $\neg C \sqcup D$. In the fuzzy case we have the following rules:

(RE1) Replace any GCI $\exists R.C \sqsubseteq D \in \mathcal{T}$ with $\mathsf{domain}(R, \exists R.C \rightarrow_G D)$
(RE2) Replace any GCI $D \sqsubseteq \forall R.C \in \mathcal{T}$ with $\mathsf{domain}(R, \exists R.\neg C \rightarrow_G \neg D)$, for Łukasiewicz and Zadeh logics.

3.4 Concept Absorption

In the following, C, C_i are concepts, A, A_j are atomic concepts.

Classical Case. For classical DLs we have the following definitions:

GCI transformation rules.

(CT1) Replace $C \sqsubseteq \sqcap_{i=1}^{n} C_i$ with $C \sqsubseteq C_i$, for $i \leq n$;
(CT2) Replace $\sqcup_{i=1}^{n} C_i \sqsubseteq C$ with $C_i \sqsubseteq C$, for $i \leq n$.

Primitive concept absorption.

(CA0) $A \sqsubseteq C$ is a *possible absorbable* GCI, A is the defined atom and $A \sqsubseteq C$ is its rewriting;
(CA1) $C \sqsubseteq \neg A$ is a *possible absorbable* GCI, A is the defined atom and $A \sqsubseteq \neg C$; is its rewriting
(CA2) If some C_j is a negated atomic concept $\neg A$, then $C \sqsubseteq \sqcup_{i=1}^{n} C_i$ is a *possible absorbable* GCI, A is the defined atom and $A \sqsubseteq \neg C \sqcup \sqcup_{i=1, i\neq j}^{n} C_i$ is its rewriting;
(CA3) If some C_j is an atomic concept A, then $\sqcap_{i=1}^{n} C_i \sqsubseteq C$ is a *possible absorbable* GCI, A is the defined atom and $A \sqsubseteq C \sqcup \sqcup_{i=1, i\neq j}^{n} \neg C_i$ is its rewriting.

Fuzzy Case. For fuzzy DLs we have the following definitions instead:

GCI transformation rules.

(FT1) Replace $\langle C \sqsubseteq C_1 \sqcap_G \ldots \sqcap_G C_n, \alpha\rangle$ with $\langle C \sqsubseteq C_i, \alpha\rangle$, for $i \leq n$;
(FT2) Replace $\langle C_1 \sqcup_G \ldots \sqcup_G C_n \sqsubseteq C, \alpha\rangle$ with $\langle C_i \sqsubseteq C, \alpha\rangle$, for $i \leq n$.

Primitive concept absorption.

(FA0) $\langle A \sqsubseteq C, \alpha\rangle$ is a *possible absorbable* GCI, A is the defined atom and $\langle A \sqsubseteq C, \alpha\rangle$ is its rewriting;
(FA1) $C \sqsubseteq \neg_l A$ is a *possible absorbable* GCI, A is the defined atom and $A \sqsubseteq \neg_l C$ is its rewriting;
(FA2.1) If some C_j is an atomic concept $\neg A$, then $\langle C \sqsubseteq \sqcup_{i=1}^{n} C_i, \alpha\rangle$ is a *possible absorbable* GCI, A is the defined atom and $\langle A \sqsubseteq \neg C \sqcup \sqcup_{i=1, i\neq j}^{n} C_i, \alpha\rangle$ is its rewriting (for Łukasiewicz or Zadeh implication and Łukasiewicz t-conorm);
(FA2.2) $\langle C \sqsubseteq A \rightarrow D, \alpha\rangle$ is a *possible absorbable* GCI, A is the defined atom and $\langle A \sqsubseteq C \rightarrow D, \alpha\rangle$ is its rewriting (if $\sqsubseteq$ and $\rightarrow$ are interpreted as the same r-implication);
(FA3) If some C_j is an atomic concept A, then $\langle \sqcap_{i=1}^{n} C_i \sqsubseteq C, \alpha\rangle$ is a *possible absorbable* GCI, A is the defined atom and $\langle A \sqsubseteq (\sqcap_{i=1, i\neq j}^{n} C_i) \rightarrow C, \alpha\rangle$ is its rewriting (for $\sqsubseteq$ and $\rightarrow$ interpreted as the residuum of $\sqcap$, or for the triple $\langle \sqsubseteq_z, \sqcap_g, \rightarrow_g\rangle$, or $\langle \sqcap, \rightarrow\rangle$ is the pair $\langle \sqcap_g, \rightarrow_g\rangle$, $\alpha = 1$ and $\sqsubseteq$ is any r-implication or Zadeh implication).
(FA4) If some C_j is an atomic concept A, then $\langle \sqcup_{i=1}^{n} C_i \sqsubseteq C, \alpha\rangle$ is a *possible absorbable* GCI, A is the defined atom and $\langle A \sqsubseteq (\sqcap_{i=1, i\neq j}^{n} \neg C_i) \rightarrow C, \alpha\rangle$ is its rewriting (for Łukasiewicz logic).

3.5 GCI Absorption Algorithm

Our algorithm has 3 phases: Phase A, a looping Phase B and a final Phase C.

Phase A: This phase has the following steps:

1. Simplify the GCIs in $\mathcal{T}$ using the concept simplifications in Section 3.1
2. Remove redundant GCIs (Section 3.2)
3. Initialise $\mathcal{T}_g = \mathcal{T}_{def} = \mathcal{T}_{inc} = \mathcal{T}_{dr} = \mathcal{T}_{disj} = \mathcal{T}_{syn} = \emptyset$
4. Remove any range and domain axiom in $\mathcal{T}$ and add it to $\mathcal{T}_{dr}$
5. Remove any disjointness axiom in $\mathcal{T}$ and add it to $\mathcal{T}_{disj}$
6. Remove any synonym axiom in $\mathcal{T}$ and add it to $\mathcal{T}_{syn}$
7. For any axiom $\tau \in \mathcal{T}$ do
   (a) if τ is of the form $\langle C \sqsubseteq D, \alpha\rangle$ then add τ to $\mathcal{T}_g$
   (b) if τ is of the form $C \doteq D$ then add $C \sqsubseteq D$ and $D \sqsubseteq C$ to $\mathcal{T}_g$

Phase B: Apply iteratively the following steps to axioms $\tau \in \mathcal{T}_g$, in the order specified below, until none of these steps can be applied to any axiom in $\mathcal{T}_g$. As soon as one step is applied once, we restart Phase B.

Redundant GCIs removal. Remove redundant GCIs (Section 3.2).

GCI transformations. If a GCI transformation rule can be applied to an axiom $\tau \in \mathcal{T}_g$, remove τ from $\mathcal{T}_g$ and add the obtained GCIs to $\mathcal{T}_g$ (see Section 3.4).

Synonym absorption. If for $\tau \in \mathcal{T}_g$ there is $\tau' \in \mathcal{T}_g \cup \mathcal{T}_{inc}$ such that $\{\tau, \tau'\}$ is $\{A \sqsubseteq B, B \sqsubseteq A\}$ with A, B atomic, then remove τ, τ' from the sets they are in and add $A \doteq B$ to $\mathcal{T}_{syn}$.

Primitive concept absorption. If there is a possible absorbable GCI $\tau \in \mathcal{T}_g$, with defined atom A, A not defined in $\mathcal{T}_{def}$, then remove τ from $\mathcal{T}_g$ and add the rewriting of τ to $\mathcal{T}_{inc}$ (see Section 3.4).

Definition absorption. If for some $\tau \in \mathcal{T}_g$ there is $\tau' \in \mathcal{T}_g \cup \mathcal{T}_{inc}$ such that $\{\tau, \tau'\}$ is $\{A \sqsubseteq C, C \sqsubseteq A\}$ with A atomic, C non-atomic, A not defined in $\mathcal{T}_{def}$ or $\mathcal{T}_{inc} \setminus \{\tau'\}$, and $\mathcal{T}_{def} \cup \{A \doteq C\}$ acyclic, then remove τ, τ' from the sets they are in and add $A \doteq C$ to $\mathcal{T}_{def}$.

Role absorption. If for some $\tau \in \mathcal{T}_g$ a role absorption rule can be applied (Section 3.3), then remove τ from $\mathcal{T}_g$ and move the obtained restriction to $\mathcal{T}_{dr}$.

Phase C: Once none of the above steps in Phase B can be applied anymore, replace any axiom $\langle C \sqsubseteq D, \alpha\rangle \in \mathcal{T}_g$ with an equivalent axiom $\langle \top \sqsubseteq E, \alpha\rangle$ where E is the simplification of $C \rightarrow D$ using the rules in Section 3.1, and return the TBox partitioning $\langle \mathcal{T}_u, \mathcal{T}_g\rangle$, where $\mathcal{T}_u = \mathcal{T}_{def} \cup \mathcal{T}_{inc} \cup \mathcal{T}_{dr} \cup \mathcal{T}_{disj} \cup \mathcal{T}_{syn}$.

Example 3. Consider the TBox $\mathcal{T} = \{A = B \sqcup C, A \sqsubseteq D\}$. Then, it can be verified that under classical and Zadeh logic, the absorbed TBox is given by $\mathcal{T}_{inc} = \{C \sqsubseteq A, B \sqsubseteq A, A \sqsubseteq D, A \sqsubseteq B \sqcup C\}$ and $\mathcal{T}_g = \mathcal{T}_{def} = \mathcal{T}_{disj} = \mathcal{T}_{syn} = \mathcal{T}_{dr} = \emptyset$, while under Łukasiewicz logic the absorbed TBox is given by $\mathcal{T}_{inc} = \{A \sqsubseteq D, A \sqsubseteq B \sqcup C, B \sqsubseteq (\neg C \sqcap A)\}$ and $\mathcal{T}_g = \mathcal{T}_{def} = \mathcal{T}_{disj} = \mathcal{T}_{syn} = \mathcal{T}_{dr} = \emptyset$.

Note that Example 3 also illustrates the usefulness of giving less priority to the "definition absorption" step. Usually, this step has highest priority. In that case Example 3 will not be absorbed completely. Also, the example shows that a non acyclic KB may be transformed into an acyclic one and, thus, avoiding the undecidability problem mentioned in the introduction.

4 Evaluation

We have implemented the fuzzy GCI absorption algorithm in our *fuzzyDL* reasoner, under Zadeh, Łukasiewicz, and classical logics.

To evaluate our algorithm, we have selected 7 well-known classical ontologies: Economy, LUBM, FBbt_XP, GALEN doctored (or GALEN-d), NCI, process and Transportation. We also have considered the fuzzy wine ontology, FuzzyWine.

Table 1 shows some information for each ontology:the expressivity (column expressivity) and the number of classes (classes), primitive GCIs (subsCls), definitional GCIs (equivCls), domain restrictions (domain), range restrictions (range), disjoint restrictions, (disjoint) and general concept inclusions (GCIs).

Table 1. Statistics of the considered ontologies.

ontology	expressivity	classes	subsCls	equivCls	domain	range	disjoint	GCIs
Economy	$\mathcal{ALCH}(\mathbf{D})$	337	409	0	47	41	142	0
FBbt_XP	$\mathcal{SHI}$	7225	12043	1028	8	6	126	0
FuzzyWine	$\mathcal{SHIF}(\mathbf{D})$	177	212	57	12	8	2	9
GALEN-d	$\mathcal{ALEHIF}+$	2748	2881	681	0	0	0	357
LUBM	$\mathcal{ALEHI}+(\mathbf{D})$	43	36	6	25	18	0	0
NCI	$\mathcal{ALE}$	27652	46800	0	70	70	0	0
process	$\mathcal{SHOIN}(\mathbf{D})$	2294	2746	12	176	159	2	0
Transportation	$\mathcal{ALCH}(\mathbf{D})$	444	452	0	81	72	634	0

These ontologies were loaded in fuzzyDL using a parser translating from OWL 2 into fuzzyDL syntax [3], discarding the elements that fuzzyDL is not able to process, such as nominals. For this reason, the number of axioms in Table 1 may be smaller than in the original ontologies. The evaluation dataset can be downloaded from `http://nmis.isti.cnr.it/~straccia/ftp/DL13.testonotologies.zip`.

Table 2 shows the same values after running the absorption algorithm for each of the three fuzzy logics considered. We include the semantics of the reasoner (column semantics with values z for Zadeh, l for Łukasiewicz and c for classical), the number of classes (classes), the sizes of $\mathcal{T}_{inc}, \mathcal{T}_{def}, \mathcal{T}_{syn}, \mathcal{T}_{dr}, \mathcal{T}_{disj}, \mathcal{T}_g$ and the running time (in seconds) of the absorption algorithm. The tests run under Mac OS X, 10.7.5, Mac Pro 2 x 3 GHz Dual-Core Intel Xeon, 9GB Ram.

Note that that the number of disjoint constraints is constant, as our absorption algorithm does not create new restrictions of these type. yet

The results are similar for the 3 semantics. In terms of running time, the algorithm behaves similarly in the 3 cases. In terms of number of absorbed axioms, the semantics is not significant in Economy, NCI and Transportation, there are small differences in FBbt_XP, GALEN-d, LUBM, FuzzyWine and process.

In all the considered ontologies $\mathcal{T}_g$ is empty, which is important from a reasoning effectiveness point of view. For instance, the reasoner gets out of memory when solving a simple query in some ontologies such as GALEN-d, but using the absorbed one our preliminary subsumption tests perform in a fraction of second.

Table 2. Results of the absorption algorithm.

semantics	ontology	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dr}$	$\mathcal{T}_{disj}$	$\mathcal{T}_{g}$	time
c	Economy	409	0	0	88	142	0	0.069
l	Economy	409	0	0	88	142	0	0.069
z	Economy	409	0	0	88	142	0	0.068
c	FBbt_XP	15187	0	0	14	126	0	1.959
l	FBbt_XP	14099	0	0	14	126	0	1.809
z	FBbt_XP	15187	0	0	14	126	0	1.678
l	FuzzyWine	328	0	0	27	2	0	0.079
z	FuzzyWine	392	0	0	27	2	0	0.092
c	GALEN-D	5507	0	18	0	0	0	1.601
l	GALEN-D	4600	0	18	0	0	0	1.295
z	GALEN-D	5507	0	18	0	0	0	1.315
c	LUBM	54	0	0	43	0	0	0.02
l	LUBM	48	0	0	43	0	0	0.017
z	LUBM	54	0	0	43	0	0	0.02
c	NCI	46800	0	0	140	0	0	2.177
l	NCI	46800	0	0	140	0	0	2.174
z	NCI	46800	0	0	140	0	0	2.187
c	process	2785	1	235	338	2	0	1.18
l	process	2769	1	235	338	2	0	1.214
z	process	2785	1	235	338	2	0	1.214
c	Transportation	452	0	0	153	634	0	0.075
l	Transportation	452	0	0	153	634	0	0.075
z	Transportation	452	0	0	153	634	0	0.074

5 Conclusions

We have worked out a first absorption algorithm for fuzzy DLs. We implemented it into the *fuzzyDL* reasoner and evaluated it over some ontologies, and our preliminary results seem to be very encouraging.

There are several directions for future research related to optimised fuzzy DL reasoning that need to be addressed:

Absorption. We plan to investigate and evaluate more deeply our absorption algorithm considering more ontologies and several heuristics (*e.g.*, which atom to select for absorption and concept name unfolding) such as those reported in the literature [1, 11, 12, 18–20, 24].

Classification. While we already have implemented in *fuzzyDL* all fuzzy lazy unfolding rules related to absorbed TBoxes and preliminary subsumption tests perform in a fraction of second, a non trivial optimised fuzzy classification algorithm in the style of *e.g.*, [6] has still to be worked out. It seems that classification is more involved in the fuzzy case (if concrete domains are involved or GCIs are graded) because, contrary to the crisp case, one may have $\mathcal{T} \models \langle A \sqsubseteq B, n_1\rangle$, $\mathcal{T} \models \langle B \sqsubseteq A, \beta_2\rangle$, $\mathcal{T} \models \langle B \sqsubseteq C, \beta_3\rangle$, $\mathcal{T} \models \langle A \sqsubseteq C, \beta_4\rangle$, with $0 < \beta_1 \neq \beta_2 \leq 1$ and $0 < \beta_1 \otimes \beta_3 < \beta_4 \leq 1$.

Instance retrieval. Other important tasks to optimise are instance checking, *i.e.*, determining the *best entailment degree* $bed(\mathcal{K}, a{:}C) = \sup\{\alpha \mid \mathcal{K} \models \langle a{:}C, \alpha\rangle\}$, and instance retrieval, *i.e.*, determining the set $ans(\mathcal{K}, C) = \{\langle a, \alpha\rangle \mid \alpha = bed(\mathcal{K}, a{:}C)\}$ [7, 8, 10, 16, 21, 22]. In that direction, *fuzzyDL* already implements a fuzzy variant of anywhere blocking [15].

References

1. 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.
2. F. Bobillo and U. Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *2008 International Conference on Fuzzy Systems (FUZZ-IEEE 2008)*, pp. 923–930. IEEE Computer Society, 2008.
3. F. Bobillo and U. Straccia. Fuzzy ontology representation using OWL 2. *International Journal of Approximate Reasoning*, 52:1073–1094, 2011.
4. S. Borgwardt and R. Peñaloza. Undecidability of fuzzy description logics. In *Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR 2012)*, pp. 232–242, 2012. AAAI Press.
5. M. Cerami and U. Straccia. On the (un)decidability of fuzzy description logics under lukasiewicz t-norm. *Information Sciences*, 227:1–21, 2013.
6. B. Glimm, I. Horrocks, B. Motik, R. Shearer, and G. Stoilos. A novel approach to ontology classification. *Journal of Web Semantics*, 14:84–101, 2012.
7. V. Haarslev and R. Möller. On the scalability of description logic instance retrieval. *Journal of Automated Reasoning*, 41(2):99–142, 2008.
8. V. Haarslev, H.-I. Pai, and N. Shiri. Optimizing tableau reasoning in alc extended with uncertainty. In *Proceedings of the 2007 International Workshop on Description Logics (DL 2007)*, 2007.
9. P. Hájek. *Metamathematics of Fuzzy Logic*. Kluwer, 1998.
10. I. Horrocks, L. Li, D. Turi, and S. Bechhofer. The instance store: DL reasoning with large numbers of individuals. In *Proceedings of the 2004 Description Logic Workshop (DL 2004)*, pp. 31–40, 2004.
11. I. Horrocks and S. Tobies. Reasoning with axioms: Theory and practice. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR 2000)*, pp. 285–296. Morgan Kaufman, 2000.
12. A. K. Hudek and G. E. Weddell. Binary absorption in tableaux-based reasoning for description logics. In *Proceedings of the 2006 International Workshop on Description Logics (DL 2006)*, volume 189 of *CEUR Workshop Proceedings*, 2006.
13. G. J. Klir and B. Yuan. *Fuzzy sets and fuzzy logic: theory and applications*. Prentice-Hall, Inc., 1995.
14. T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Journal of Web Semantics*, 6:291–308, 2008.
15. B. Motik, R. Shearer, and I. Horrocks. Optimized reasoning in description logics using hypertableaux. In *Proceedings of the 21st International Conference on Automated Deduction: Automated Deduction*, CADE-21, pp. 67–83, 2007. Springer-Verlag.
16. N. Simou, T. P. Mailis, G. Stoilos, and G. B. Stamou. Optimization techniques for fuzzy description logics. In *Proceedings of the 23rd International Workshop on Description Logics (DL 2010)*, volume 573 of CEUR Electronic Workshop Proceedings, 2010.
17. U. Straccia. Description logics with fuzzy concrete domains. In *21st Conference on Uncertainty in Artificial Intelligence (UAI 2005)*, pp. 559–567, 2005. AUAI Press.
18. D. Tsarkov and I. Horrocks. Efficient reasoning with range and domain constraints. In *Proceedings of the 2004 Description Logic Workshop (DL 2004)*, pp. 41–50, 2004.
19. D. Tsarkov, I. Horrocks, and P. F. Patel-Schneider. Optimizing terminological reasoning for expressive description logics. *Journal of Automated Reasoning*, 39(3):277–316, 2007.

20. J. Wu and V. Haarslev. Planning of axiom absorption. In *Proceedings of the 21st International Workshop on Description Logics* (DL 2008), volume 353 of *CEUR Workshop Proceedings*, 2008.
21. J. Wu, A. K. Hudek, D. Toman, and G. E. Weddell. Absorption for aboxes. In *Proceedings of the 2012 International Workshop on Description Logics (DL 2012)*, volume 846 of *CEUR Workshop Proceedings*, 2012.
22. J. Wu, A. K. Hudek, D. Toman, and G. E. Weddell. Assertion absorption in object queries over knowledge bases. In *Proceedings of the 13th International Conference Principles of Knowledge Representation and Reasoning (KR 2012)*. AAAI Press, 2012.
23. L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.
24. M. Zuo and V. Haarslev. High performance absorption algorithms for terminological reasoning. In *Proceedings of the 2006 International Workshop on Description Logics (DL 2006)*, volume 189 of *CEUR Workshop Proceedings*, 2006.