Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems 292 (2016) 98–129

FUZZY sets and systems

www.elsevier.com/locate/fss

Optimising fuzzy description logic reasoners with general concept inclusion absorption ☆

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 31 January 2014; received in revised form 23 September 2014; accepted 30 October 2014

Available online 7 November 2014

Abstract

General Concept Inclusion (GCI) absorption algorithms have shown to play an important role in classical Description Logic (DL) reasoners. They allow to transform GCIs into simpler forms to which we may apply specialised inference rules, returning important performance gains. In this work, we develop the first absorption algorithm for fuzzy DLs, implement it in the *fuzzyDL* reasoner and evaluate it extensively over both classical and fuzzy ontologies. The results show that our algorithm improves the performance of the reasoner significantly.
© 2014 Elsevier B.V. All rights reserved.

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

1. Introduction

Description Logics (DLs for short) [1] is a well-known 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* [2]. Indeed, the current standard language for specifying ontologies is the Web Ontology Language (OWL 2) [3], which is based on the DL $\mathcal{SROIQ}(\mathbf{D})$ [4].

Fuzzy DLs have been proposed as an extension to classical DLs with the aim of dealing with *fuzzy, vague*, and *imprecise* concepts. In these logics, 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 [5], an important number of works can be found in the literature (good surveys on *fuzzy DLs* can be found in [6,7]).

However, little effort has been paid so far to the study and implementation of optimization techniques, which is essential to reason with real-world scenarios in practice. Up to now, we are only aware of two works [8,9], but neither of them considers reasoning with a general TBox.

☆ This paper is a revised and extended version of "General Concept Inclusion Absorptions for Fuzzy Description Logics: A First Step", published in the Proceedings of the 26th International Workshop on Description Logics (DL 2013).

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

http://dx.doi.org/10.1016/j.fss.2014.10.029
0165-0114/© 2014 Elsevier B.V. All rights reserved.

In classical DLs, a TBox contains *General Concept Inclusion* axioms (GCIs) of the form

$$C \sqsubseteq D,$$

intuitively encoding that an instance of class C is also an instance of class D. *GCI absorption* is a technique that allows to transform GCIs into simpler forms to which we may apply then specialised inference rules: the so-called *lazy unfolding rules* [1,10–16]. The basic principle in absorption is to avoid the *internalisation* of GCIs, which is the naive way to reason with these axioms. For example, the GCI

$$A \sqcap C \sqsubseteq D \tag{1}$$

is usually internalised into the canonical form

$$\top \sqsubseteq \neg A \sqcup \neg C \sqcup D.$$

The original GCI encodes the idea that an instance of both classes A and C is also an instance of class D, while the canonical form encodes that any object is either not an instance of A, or not an instance of C, or an instance of D. The consequence of this transformation is that the inference rule handling GCIs will be applied to *every* object that occurs during the reasoning process. If A is an atomic concept, it is a smarter option to rewrite Eq. (1) as

$$A \sqsubseteq \neg C \sqcup D.$$

The subsequent idea behind the latter axiom is that an instance of class A is either not an instance of class C or an instance of class D. The consequence of this transformation is that the inference rule for such axiom applies (among other conditions) *only if* an object has been inferred as being an instance of concept A. This implies a reduction of the number of applications of the axiom and hence of the reasoning time.

While absorption algorithms have experimentally shown to provide a very important performance gain for classical DLs reasoners, no such algorithms have been investigated so far in the context of fuzzy DLs. It is expected that absorption would provide a reduction in the reasoning time within fuzzy DLs.

Another benefit of absorption for fuzzy DLs is the possibility of transforming a non-acyclic ontology [17] into an acyclic one. This is important from a computational point of view since reasoning problems are in general undecidable in the presence of GCIs for several fuzzy DLs [18,19,21] (for instance in Łukasiewicz and Product $\mathcal{ALC}$), but are decidable if the TBox is acyclic [18,20]. That is, in some cases it may be possible to transform a non-acyclic fuzzy ontology (for which reasoning problems are not guaranteed to be decidable) into an equivalent acyclic one (and thus for which reasoning problems are guaranteed to be decidable).

It is also worth to note that the absorption method holds for the case of a finite linearly-ordered truth space as well (for which the decidability of the satisfiability problem is always guaranteed). Therefore, the absorptions can be beneficial for finitely valued DLs as well [22,23].

The aim of this paper is to propose the first absorption algorithm for fuzzy DLs. This algorithm is implemented in the *fuzzyDL* reasoner[1] [24] and evaluated over several existing ontologies. Our results show that our algorithm significantly improves the performance of the reasoner.

The rest of this paper is organised as follows. Section 2 recalls some preliminaries on fuzzy logic and fuzzy DLs. Section 3 presents our GCI absorption algorithm. Next, Section 4 discusses an experimental evaluation of the algorithm. Finally, Section 5 sets out some conclusions and ideas for future research.

2. Fuzzy DLs basics

In this section we recap some basic definitions needed during the rest of the paper. We refer the reader to [6,7] for a more in depth presentation.

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

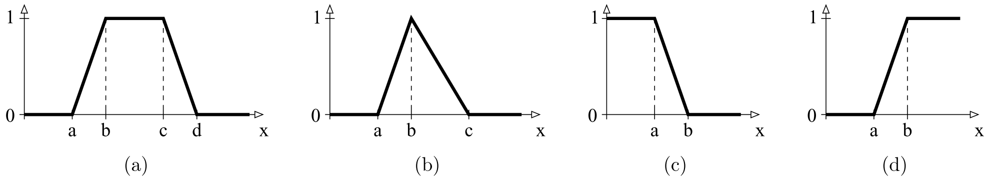


Fig. 1. (a) Trapezoidal function $trz(a,b,c,d)$, (b) triangular function $tri(a,b,c)$, (c) left shoulder function $ls(a,b)$, and (d) right shoulder function $rs(a,b)$.

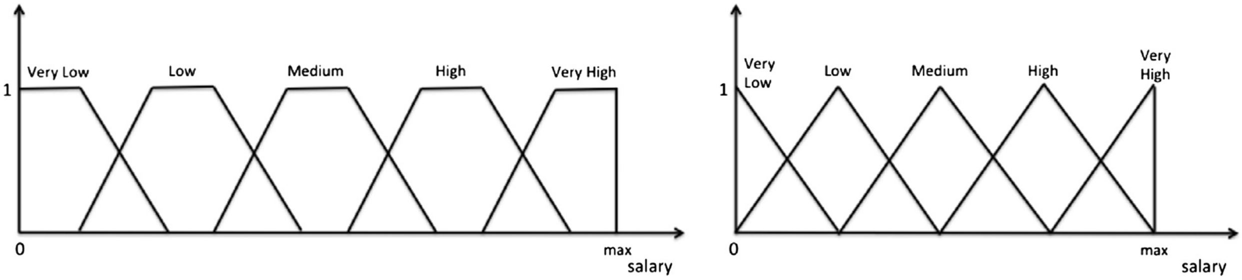


Fig. 2. Fuzzy sets over the domain of salaries using trapezoidal or triangular functions.

2.1. Mathematical fuzzy logic

In *Mathematical Fuzzy Logic* [25], 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 (usually) $[0,1]$. This degree is called *degree of truth* of the logical statement ϕ in the interpretation $\mathcal{I}$. The *truth space* is usually $L=[0,1]$, but another popular choice 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 classical two-valued case.

The main idea behind fuzzy logic is that of fuzzy set. Let X be a set of elements called the reference set. A *fuzzy subset* A of X is defined by a membership function $\mu_A(x)$, or simply $A(x)$, which assigns to every $x \in X$ a value in a truth space L.

Some popular membership functions, commonly used to define fuzzy sets, are the trapezoidal (Fig. 1 (a)), the triangular (Fig. 1 (b)), the left-shoulder (Fig. 1 (c)), and the right-shoulder function (Fig. 1 (d)).[2] 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 [26, Chapter 10]. However, an easy and typically satisfactory method to define the membership functions is to uniformly partition the range of possible values (bounded by a minimum and maximum value) into 5 or 7 fuzzy sets using either trapezoidal functions (as illustrated on the left in Fig. 2), or using triangular functions (as illustrated on the right in Fig. 2). The latter is the more used one, as it involves less parameters.

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 the 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.

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 [25], 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 Zadeh logic [28]. The combination functions of these logics can be found in Table 1, while Table 2 shows some important properties of these functions.

[2] Note that these membership functions assume that the domain of a fuzzy set is a dense total ordering, which is not always the case.

Table 1
Combination functions of various fuzzy logics.

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

Table 2
Some important properties of truth combination functions.

Property	Łukasiewicz	Gödel	Product	Zadeh
$\alpha \otimes \ominus\alpha = 0$	✓	✓	✓	
$\alpha \oplus \ominus\alpha = 1$	✓			
$\alpha \otimes \alpha = \alpha$		✓		✓
$\alpha \oplus \alpha = \alpha$		✓		✓
$\ominus\ominus\alpha = \alpha$	✓			✓
$\alpha \Rightarrow \beta = \ominus\alpha \oplus \beta$	✓			✓
$\alpha \Rightarrow \beta = \ominus\beta \Rightarrow \ominus\alpha$	✓			✓
$\ominus(\alpha \Rightarrow \beta) = \alpha \otimes \ominus\beta$	✓			✓
$\ominus(\alpha \otimes \beta) = \ominus\alpha \oplus \ominus\beta$	✓	✓	✓	✓
$\ominus(\alpha \oplus \beta) = \ominus\alpha \otimes \ominus\beta$	✓	✓	✓	✓
$\alpha \otimes (\beta \oplus \gamma) = (\alpha \otimes \beta) \oplus (\alpha \otimes \gamma)$		✓		✓
$\alpha \oplus (\beta \otimes \gamma) = (\alpha \oplus \beta) \otimes (\alpha \oplus \gamma)$		✓		✓

We will often use an optional subscript $X \in \{\text{Ł}, \text{G}, \Pi, \text{Z}\}$ to indicate that an operator belongs to Łukasiewicz, Gödel, Product and Zadeh fuzzy logics, respectively. For instance, $\alpha \otimes_G \beta$ refers to Gödel conjunction.

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 $\alpha \Rightarrow_{KD} \beta = \max(1-\alpha, b)$. This latter implication is called *Kleene-Dienes implication* and is denoted $\Rightarrow_{KD}$.

The name of Zadeh fuzzy logic is used following the tradition in the setting of fuzzy DLs, even if it might lead to confusion because the logic does not include the *Zadeh implication* (or Rescher implication), denoted $\Rightarrow_Z$ and defined as:

$$\alpha \Rightarrow_Z \beta = \begin{cases} 1, & \text{if } \alpha \leq \beta \\ 0 & \text{if } \alpha > \beta. \end{cases}$$

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

Truth combination functions are not arbitrarily combined to form a fuzzy logic; in practice, one fixes the t-norm $\otimes$, builds an implication from $\otimes$ (for example, its residuum), considers the dual t-conorm of $\otimes$, and concludes by selecting either the involutive or the residuated negation.

Our *fuzzy statements* will have the form $\langle \phi, \alpha \rangle$, where ϕ is a statement and $\alpha \in (0,1]$, encoding that the degree of truth of ϕ is *greater than or equal to* α. 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 $\mathcal{I}(\phi) \geq \alpha$.

2.2. A fuzzy DL

To illustrate our absorption algorithm and our experimentation, we introduce a fuzzy variant of the DL $\mathcal{SHOIF}_g(\mathbf{D})$ [1,29], already presented in [7,30]. Fuzzy $\mathcal{SHOIF}_g(\mathbf{D})$ is obtained by extending fuzzy $\mathcal{ALC}$ with transitive roles,[3] inverse roles (indicated with the letter $\mathcal{I}$), role hierarchies (indicated with the letter $\mathcal{H}$), *nominals* (indicated with the letter $\mathcal{O}$), global functional roles (indicated with the letter $\mathcal{F}_g$), and concrete domains (indicated with the letter **D**). In the particular case where nominals are restricted to *object property value restrictions*, $\mathcal{O}$ is replaced with letter $\mathcal{B}$.

Syntax Let **A**, **R**, and **I** be pairwise disjoint sets of *concept names* (also called atomic concepts), *role names*, and *individual names*, respectively. Each role is either an *object property* (denoted by R) or a *data type property* (denoted by T). Object properties link pairs of individuals, whereas data type properties relate an individual with a *data value* or a *data type predicate*. The *inverse* of an object property R is an object property indicated with R^-. Object properties are defined according to the following syntax rule, where $r \in \mathbf{R}$:

$$
\begin{array}{rll}
R \rightarrow & r & \text{(atomic object property)} \\
& R^- & \text{(inverse object property)}
\end{array}
$$

In classical DLs, the range of data type properties are data values (numerical or textual, among many other possibilities). In the fuzzy case, in addition to relating an individual with a data value, it is possible to relate it with a fuzzy membership function. Typically, in fuzzy DLs data type predicates (denoted by **d**) are unary and examples of them over a dense total ordered concrete domain are the following known membership functions:

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

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

The set of *concepts* (denoted C, D) is built from concept names $A \in \mathbf{A}$ using connectives and quantification constructors over object properties R, data type properties T and individuals $a \in \mathbf{I}$, according to the following syntactic rule:

$$
\begin{array}{rcll}
C, D \rightarrow & A & \mid & \text{(atomic concept)} \\
& \top & \mid & \text{(universal concept)} \\
& \bot & \mid & \text{(bottom concept)} \\
& \neg C & \mid & \text{(concept negation)} \\
& C \sqcap D & \mid & \text{(concept conjunction)} \\
& C \sqcup D & \mid & \text{(concept disjunction)} \\
& C \rightarrow D & \mid & \text{(concept implication)} \\
& \{a\} & \mid & \text{(nominal)} \\
& \forall R.C & \mid & \text{(object property universal restriction)} \\
& \exists R.C & \mid & \text{(object property existential restriction)} \\
& \forall T.\mathbf{d} & \mid & \text{(data type property universal restriction)} \\
& \exists T.\mathbf{d} & & \text{(data type property existential restriction).}
\end{array}
$$

Concepts of the form $\exists R.\{a\}$ are called *object property value restrictions*. In the following, we will use the expression $\{a_1, \ldots, a_n\}$ (called *concept enumeration*) as a macro for the concept expression $\{a_1\} \sqcup \ldots \sqcup \{a_n\}$. Please also note that, strictly speaking concept implications are not syntactically part of classical $\mathcal{SHOIF}_g(\mathbf{D})$, but we included it here for convenience within fuzzy $\mathcal{SHOIF}_g(\mathbf{D})$: it plays an important role in the absorption algorithm in some cases. Under classical semantics, the concept $C \rightarrow D$ is a macro for $\neg C \sqcup D$.

A fuzzy *knowledge base* (KB), also called *fuzzy ontology*, is a set of axioms. Formally, a fuzzy KB $\mathcal{K}$ is a tuple $\langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where $\mathcal{A}$ is an ABox, $\mathcal{T}$ is a TBox and $\mathcal{R}$ is an RBox. We may also omit one or more among $\mathcal{A}$, $\mathcal{T}$ and $\mathcal{R}$ if they are empty.

Now we will define the *axioms* that can be expressed in a fuzzy ontology. We will use $\alpha \in (0, 1]$ to denote a truth value. If α is omitted, $\alpha = 1$ is assumed. If $\alpha \neq 1$, we say that the axiom is *graded*.

An *ABox* $\mathcal{A}$ (Assertional Box) consists of a finite set of assertion axioms. There are two types of assertion axioms:

[3] $\mathcal{ALC}$ with transitive roles is called $\mathcal{S}$.

- A *concept assertion* $\langle a{:}C, \alpha\rangle$ states that a is an instance of concept C to degree at least α.
- A *role assertion* $\langle (a_1, a_2){:}R, \alpha\rangle$ states that (a_1, a_2) is an instance of role R to degree at least α. For data type properties, we have $\langle (a, v){:}T, \alpha\rangle$.

An *RBox* $\mathcal{R}$ (Role Box) can have three types of axioms:

- A *Role Inclusion Axiom* (RIA) is of the form $\langle R_1 \sqsubseteq R_2, \alpha\rangle$ and states that object property R_1 is a sub-role of object property R_2 to degree at least α.
- A *transitivity axiom* $\mathsf{trans}(R)$ states that object property R is transitive.
- A *functional axiom* $\mathsf{func}(R)$ forces role R to be functional.

A *TBox* $\mathcal{T}$ (Terminological Box) can have the following axioms:

- A *General Concept Inclusion* (GCI) axiom is of the form $\langle C \sqsubseteq D, \alpha\rangle$ and states that C is a sub-concept of D to degree at least α. $\langle C \sqsubseteq D, 1\rangle$ will often be shortened to $C \sqsubseteq D$.
- A *primitive* GCI is a particular case of GCI having the form $\langle A \sqsubseteq C, \alpha\rangle$, where A is atomic.
- A *generalised definitional* GCI is of the form $C \doteq D$, where both C and D are concepts. This axiom can be seen as a shorthand for both $C \sqsubseteq D$ and $D \sqsubseteq C$.
- A *definitional* GCI is a particular case of generalised definitional GCI having the form $A \doteq C$, with A atomic.
- A *synonym* GCI is a particular case of definitional GCI having the form $A \doteq B$, where both A and B are atomic.
- A *constraint* axiom is a particular case of GCI with one of the following forms:
  - $\mathsf{domain}(R, C)$, called *domain restriction*, that restricts the domain of object property R to be concept C;
  - $\mathsf{range}(R, C)$, called *range restriction*, that restricts the range of object property R to be concept C; and
  - $\mathsf{disjoint}(A, B)$, called *disjoint restriction*, that restricts the concept names A and B to be disjoint.

In primitive and definitional axioms, A is called the *head* and C is the *body*.

The notions of transitivity, functionality, domain, range, and disjointness are standard in classical DLs. However, in the fuzzy case, several definitions would be possible. Later on we will describe our chosen semantics for these axioms.

Example 2.1. We have built a fuzzy wine ontology[4] according to the FuzzyOWL 2 proposal [33]. One of the GCIs in there is of the form

$$SparklingWine \sqcap \big(\exists hasSugar.tri(32, 41, 50)\big) \sqsubseteq DemiSecSparklingWine$$

where *hasSugar* is a data type property whose values are measured in g/L (grams per litre).

Semantics Let us fix a fuzzy logic $X \in \{Ł, G, \Pi, Z\}$ (see Section 2.1). The semantics of the logic is given by an abstract interpretation $\mathcal{I}$ (for the individuals) and a *fuzzy concrete domain* or *fuzzy data type theory* [32] $\mathbf{D}$. This latter consists of a tuple $\mathbf{D} = \langle \Delta^{\mathbf{D}}, \cdot^{\mathbf{D}}\rangle$, with data type domain $\Delta^{\mathbf{D}}$, and a function $\cdot^{\mathbf{D}}$ that assigns to each data type predicate $\mathbf{d}$ a function $\mathbf{d}^{\mathbf{D}} : \Delta^{\mathbf{D}} \to [0, 1]$ (we are restricting to unary data types).

In classical DLs, an interpretation $\mathcal{I}$ maps 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). However, 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* $\mathcal{I}$ with respect to $\mathbf{D}$ is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a non-empty (crisp) set $\Delta^{\mathcal{I}}$ (the *domain*) disjoint with $\Delta^{\mathbf{D}}$ and of a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$ that assigns:

1. to each atomic concept A a function $A^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0, 1]$;
2. to each object property R a function $R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0, 1]$;
3. to each data type property T a function $T^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}} \to [0, 1]$;

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

4. to each concrete value v an element $v^{\mathcal{I}} \in \Delta^{\mathbf{D}}$; and
5. 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).

Let $x, y \in \Delta^{\mathcal{I}}$ denote elements of the domain. A fuzzy interpretation function is extended to inverse roles by imposing

$$\left(R^{-}\right)^{\mathcal{I}}(x, y) = R^{\mathcal{I}}(y, x).$$

Moreover, a fuzzy interpretation function is extended to concepts as follows:

$$\begin{aligned}
&\top^{\mathcal{I}}(x) = 1 \\
&\bot^{\mathcal{I}}(x) = 0 \\
&(\neg C)^{\mathcal{I}}(x) = \ominus_X C^{\mathcal{I}}(x) \\
&(C \sqcap D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \otimes_X D^{\mathcal{I}}(x) \\
&(C \sqcup D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \oplus_X D^{\mathcal{I}}(x) \\
&(C \to D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \Rightarrow_X D^{\mathcal{I}}(x) \\
&\{a\}^{\mathcal{I}}(x) = 1 \text{ if } a^{\mathcal{I}} = x, \text{ else } 0 \\
&(\forall R.C)^{\mathcal{I}}(x) = \inf_{y \in \Delta^{\mathcal{I}}} \left\{R^{\mathcal{I}}(x, y) \Rightarrow_X C^{\mathcal{I}}(y)\right\} \\
&(\exists R.C)^{\mathcal{I}}(x) = \sup_{y \in \Delta^{\mathcal{I}}} \left\{R^{\mathcal{I}}(x, y) \otimes_X C^{\mathcal{I}}(y)\right\} \\
&(\forall T.\mathbf{d})^{\mathcal{I}}(x) = \inf_{y \in \Delta^{\mathcal{I}}} \left\{T^{\mathcal{I}}(x, y) \Rightarrow_X \mathbf{d}^{\mathbf{D}}(y)\right\} \\
&(\exists T.\mathbf{d})^{\mathcal{I}}(x) = \sup_{y \Delta^{\mathcal{I}}} \left\{T^{\mathcal{I}}(x, y) \otimes_X \mathbf{d}^{\mathbf{D}}(y)\right\}.
\end{aligned}$$

Hence, for every concept C we get a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0, 1]$.

Remark 1. Gödel and Zadeh fuzzy DLs include by definition Gödel concept conjunction and Gödel concept disjunction. Note that these constructors (denoted by $\sqcap_G$ and $\sqcup_G$, respectively) can be defined under Łukasiewicz and Product logic as well:

- $C \sqcap_G D := C \sqcap (C \to D)$.
- $C \sqcup_G D := (C \to D) \to D$.

In classical DLs, the semantics of conjunction and Gödel concept conjunction coincide, and the same holds for disjunction and Gödel disjunction.

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

1. $\mathcal{I}$ satisfies $\langle a{:}C, \alpha\rangle$ if $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq \alpha$;
2. $\mathcal{I}$ satisfies $\langle (a, b){:}R, \alpha\rangle$ if $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq \alpha$;
3. $\mathcal{I}$ satisfies $\langle (a, v){:}T, \alpha\rangle$ if $T^{\mathcal{I}}(a^{\mathcal{I}}, v^{\mathcal{I}}) \geq \alpha$;
4. $\mathcal{I}$ satisfies $\langle C \sqsubseteq D, \alpha\rangle$ if $\inf_{x \in \Delta^{\mathcal{I}}}\{C^{\mathcal{I}}(x) \Rightarrow_X D^{\mathcal{I}}(x)\} \geq \alpha$;
5. $\mathcal{I}$ satisfies $C \doteq D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$;
6. $\mathcal{I}$ satisfies $\mathsf{domain}(R, C)$ if $\mathcal{I}$ satisfies $\exists R.\top \sqsubseteq C$;
7. $\mathcal{I}$ satisfies $\mathsf{range}(R, C)$ if $\mathcal{I}$ satisfies $\top \sqsubseteq \forall R.C$;
8. $\mathcal{I}$ satisfies $\mathsf{disjoint}(A, B)$ if $\mathcal{I}$ satisfies $A \sqcap_G B \sqsubseteq \bot$.
9. $\mathcal{I}$ satisfies $\langle R_1 \sqsubseteq R_2, \alpha\rangle$ if $\inf_{x, y \in \Delta^{\mathcal{I}}}\{R_1{}^{\mathcal{I}}(x, y) \Rightarrow_X R_2{}^{\mathcal{I}}(x, y)\} \geq \alpha$;
10. $\mathcal{I}$ satisfies $\mathsf{trans}(R)$ if for all $x, y \in \Delta^{\mathcal{I}}$,

$$R^{\mathcal{I}}(x, y) \geq \sup_{z \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x, z) \otimes_X R^{\mathcal{I}}(z, y);$$

11. $\mathcal{I}$ satisfies $\mathsf{func}(R)$, where R is an object property, if for all $x, y, z \in \Delta^{\mathcal{I}}$ with $y \neq z$,

$$\min\big(R^{\mathcal{I}}(x, y), R^{\mathcal{I}}(x, z)\big) = 0;$$

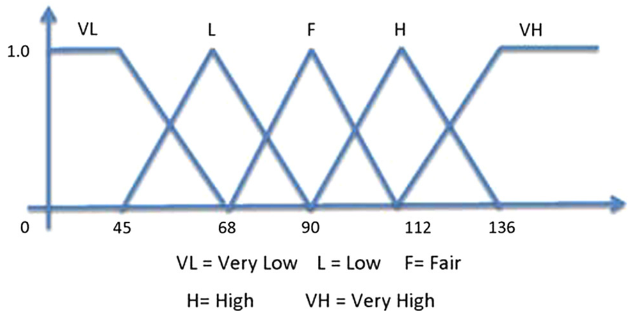
12. $\mathcal{I}$ satisfies $\mathsf{func}(T)$, where T is a data type property, if for all $x \in \Delta^{\mathcal{I}}$ and all $y, z \in \Delta^{\mathbf{D}}$ with $y \neq z$,

$$\min\big(T^{\mathcal{I}}(x, y), T^{\mathcal{I}}(x, z)\big) = 0.$$

Exceptionally, Zadeh fuzzy DLs use Zadeh implication in the semantics of GCIs, since Kleene-Dienes implication produce some counter-intuitive effects [31]. We say that $\mathcal{I}$ is a model of an ontology $\mathcal{K}$ iff $\mathcal{I}$ satisfies each axiom in $\mathcal{K}$, and that $\mathcal{K}$ *entails* an axiom α, denoted $\mathcal{K} \models \alpha$, if any model of $\mathcal{K}$ satisfies α.

It remains to define the notion of *equivalence* of concepts, GCIs, ABoxes, TBoxes, RBoxes, and KBs. Two concepts C and D are equivalent, denoted $C \equiv D$, iff for all interpretations $\mathcal{I}$, $C^{\mathcal{I}} = D^{\mathcal{I}}$. Two GCIs $C_1 \sqsubseteq C_2$ and $D_1 \sqsubseteq D_2$ are equivalent, denoted $C_1 \sqsubseteq C_2 \equiv D_1 \sqsubseteq D_2$, iff for all interpretations $\mathcal{I}$, $(C_1 \to C_2)^{\mathcal{I}} = (D_1 \to D_2)^{\mathcal{I}}$. Two TBoxes $\mathcal{T}$, $\mathcal{T}'$ (resp. ABoxes $\mathcal{A}$, $\mathcal{A}'$, RBoxes $\mathcal{R}$, $\mathcal{R}'$, KBs $\mathcal{K}$, $\mathcal{K}'$) are equivalent, denoted $\mathcal{T} \equiv \mathcal{T}'$ (resp. $\mathcal{A} \equiv \mathcal{A}'$, $\mathcal{R} \equiv \mathcal{R}'$, $\mathcal{K} \equiv \mathcal{K}$) iff $\mathcal{T}$ and $\mathcal{T}'$ (resp. $\mathcal{A}$ and $\mathcal{A}'$, $\mathcal{R}$ and $\mathcal{R}'$, $\mathcal{K}$ and $\mathcal{K}'$) entail the same set of axioms.

Example 2.2. *fuzzyDL-Learner*[5] [34] is a system that illustrates how to 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 and the graded hotel judgements of the users from Web sites (such as Trip Advisor[6]), and asks about what characterises good hotels. Then, it is possible to learn the axiom $\langle \exists hasPrice.High \sqsubseteq GoodHotel, 0.569\rangle$, where *hasPrice* is a data type property whose values are measured in euros and the price concrete domain has been automatically fuzzified as illustrated below.



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. A GCI absorption algorithm for fuzzy DLs

Given a fuzzy KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R}\rangle$, the aim of our GCI absorption algorithm is to build an equivalent fuzzy KB $\mathcal{K}' = \langle \mathcal{A}', \mathcal{T}', \mathcal{R}\rangle$ to which apply specialised inference rules. In $\mathcal{K}'$, $\mathcal{A} \subseteq \mathcal{A}'$, and $\mathcal{T}'$ is the union of two disjoint sets of axioms $\mathcal{T}_g$ and $\mathcal{T}_u$ verifying the following conditions:

1. $\mathcal{T}_g$ is a set of GCIs of the form $\langle \top \sqsubseteq C, \alpha\rangle$,
2. $\mathcal{T}_u = \mathcal{T}_{def} \cup \mathcal{T}_{inc} \cup \mathcal{T}_{dom} \cup \mathcal{T}_{rg} \cup \mathcal{T}_{disj} \cup \mathcal{T}_{syn}$ is the disjoint union of

[5] http://straccia.info/software/FuzzyDL-Learner.
[6] http://www.tripadvisor.com.

   (a) $\mathcal{T}_{def}$, which contains definitional GCIs only;
   (b) $\mathcal{T}_{inc}$, which contains primitive GCIs only;
   (c) $\mathcal{T}_{dom}$, which contains domain restrictions only;
   (d) $\mathcal{T}_{rg}$, which contains range restrictions only;
   (e) $\mathcal{T}_{disj}$, which contains disjoint restrictions only;
   (f) $\mathcal{T}_{syn}$, which contains synonym GCIs only;
3. there cannot be a concept name A that is a head of axioms in $\mathcal{T}_{def}$ and $\mathcal{T}_{inc}$; and
4. there cannot be $\mathsf{disjoint}(A, B) \in T_{disj}$, where both A, B are head of axioms in $\mathcal{T}_{def}$.[7]

If $\mathcal{T}_g = \emptyset$, the TBox is called *lazy unfoldable*. Informally, this partitioning makes it possible to apply lazy unfolding rules to axioms in $\mathcal{T}_u$ only. Intuitively, if we have in $\mathcal{T}_{inc}$ an axiom of the form $\langle A \sqsubseteq C, \alpha \rangle$ and $\sqsubseteq$ is interpreted as an r-implication, then a lazy unfolding rule for it is informally "if a is an instance of A to degree β then a is an instance of C to degree $\alpha \otimes \beta$". This is a graded variant of the lazy unfolding rule for crisp primitive inclusions $A \sqsubseteq C$ that can be seen as "if a is an instance of A then a is an instance of C". In the case $\sqsubseteq$ is interpreted as Zadeh implication, then we can consider a lazy unfolding rule of the form 'if a is an instance of A to degree β then a is also an instance of C to degree β". Analogous rules can be worked out for the other axioms occurring in $\mathcal{T}_u$ and have been implemented in *fuzzyDL*.

In the following, we are going to describe how to compute $\mathcal{T}_u$ and $\mathcal{T}_g$. Firstly, Sections 3.1–3.4 present some auxiliary transformations and simplifications. Then, Section 3.5 describes the partitioning algorithm. Note that all the transformations and simplifications do not hold for every fuzzy logic: Appendix B summarises their applicability.

3.1. Concept simplifications

In this section we will show how to simplify a concept to a simpler one. Simplifications are of the form $C_1 \mapsto C_2$, indicating that concept C_1 is replaced with C_2. The simplifications are applied recursively and the constructors $\sqcap, \sqcup$ are considered n-ary, *commutative* and *associative*. Note that the applicability of the concept simplifications depend on the particular fuzzy logic, as enumerated in Table B.6.

(CS1) $C \sqcap \bot \mapsto \bot$
(CS2) $C \sqcap \top \mapsto C$
(CS3) $C \sqcap_G C \mapsto C$
(CS4) $C \sqcup_G C \mapsto C$
(CS5) $C \sqcup \bot \mapsto C$
(CS6) $C \sqcup \top \mapsto \top$
(CS7) $\exists R.\bot \mapsto \bot$
(CS8) $\forall R.\top \mapsto \top$
(CS9) $\neg\top \mapsto \bot$
(CS10) $\neg\bot \mapsto \top$
(CS11) $\neg\neg C \mapsto C$
(CS12) $\neg(C \sqcap D) \mapsto \neg C \sqcup \neg D$
(CS13) $\neg(C \sqcup D) \mapsto \neg C \sqcap \neg D$
(CS14) $\neg\forall R.C \mapsto \exists R.\neg C$
(CS15) $\neg\exists R.C \mapsto \forall R.\neg C$
(CS16) $C \rightarrow D \mapsto \neg C \sqcup D$
(CS17) $\neg(C \rightarrow D) \mapsto C \sqcap \neg D$
(CS18) $C \sqcap \neg C \mapsto \bot$
(CS19) $C \sqcup \neg C \mapsto \top$
(CS20) $C \sqcap (C \sqcup D) \mapsto C$
(CS21) $C \sqcup (C \sqcap D) \mapsto C$

[7] This condition prevents $C \sqsubseteq D$ to be introduced via e.g., $A \doteq C$, $B \doteq \neg D$, $\mathsf{disjoint}(A, B)$.

(CS22) $C \sqcap (\neg C \sqcup D) \mapsto C \sqcap D$
(CS23) $C \sqcup (\neg C \sqcap D) \mapsto C \sqcup D$
(CS24) $\forall R.C \sqcap_G \forall R.D \mapsto \forall R.(C \sqcap_G D)$
(CS25) $\exists R.C \sqcup \exists R.D \mapsto \exists R.(C \sqcup D)$
(CS26) $(C \sqcup_G D) \to E \mapsto (C \to E) \sqcap_G (D \to E)$
(CS27) $\exists R.C \sqcap_G \exists R.\top \mapsto \exists R.C$
(CS28) $C \sqcap (D \sqcap E) \mapsto C \sqcap D \sqcap E$
(CS29) $C \sqcup (D \sqcup E) \mapsto C \sqcup D \sqcup E$
(CS30) $C \sqcap (D \sqcup_G E) \mapsto (C \sqcap D) \sqcup_G (C \sqcap E)$
(CS31) $C \sqcup (D \sqcap_G E) \mapsto (C \sqcup D) \sqcap_G (C \sqcup E)$
(CS32) $C \to \top \mapsto \top$
(CS33) $\top \to C \mapsto C$
(CS34) $\bot \to C \mapsto \top$
(CS35) $C \to \bot \mapsto \neg C$

Proposition 3.1. *Concept simplifications (CS1)–(CS35) transform a fuzzy concept into an equivalent one.*

Proof. Straightforward using the properties in [25]. □

3.2. Redundant GCIs elimination

In this section, we will show that some axioms can safely be removed, since they are trivially satisfied. The GCIs that can be eliminated are those of the following forms:

(GE1) $\langle \bot \sqsubseteq C, \alpha \rangle$
(GE2) $\langle C \sqsubseteq \top, \alpha \rangle$
(GE3) $C \doteq C$
(GE4) $\langle C \sqsubseteq C, \alpha \rangle$
(GE5) $\langle A \sqcap D \sqsubseteq A, \alpha \rangle$
(GE6) $\langle A \sqsubseteq A \sqcup D, \alpha \rangle$
(GE7) $\mathsf{domain}(R, \top)$
(GE8) $\mathsf{range}(R, \top)$

Proposition 3.2. *Axioms (GE1)–(GE8) are satisfied by any fuzzy KB.*

Proof. Trivial. □

All of the previous eliminations are applicable in any of the fuzzy logics considered.

Remark 2. It is worth to note that (GE4)–(GE6) hold in Zadeh logic because Zadeh implication has been assumed in the semantics of GCIs.

3.3. Role absorptions

In this section we will transform some axioms into domain and range restrictions. This way, the axiom is only taken into account after a new role relation is created, reducing the application space of the axioms. Firstly, we will extend the classical basic role absorption techniques, and then we will extend it to cover more cases. Table B.7 summarises the applicability of our role absorptions.

3.3.1. Basic role absorption

In the classical case, $\top \sqsubseteq \forall R.C$ and $\exists R.\top \sqsubseteq C$ correspond to $\mathsf{range}(R, C)$ and $\mathsf{domain}(R, C)$, respectively, and, thus, we may transform these GCIs into domain and range restrictions [12]. We have the following rules:

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

3.3.2. Extended role absorptions

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

(RE1) Replace every GCI $\exists R.C \sqsubseteq D \in \mathcal{T}$ with $\mathsf{domain}(R, \exists R.C \to D)$
(RE2) Replace every GCI $D \sqsubseteq \forall R.C \in \mathcal{T}$ with $\mathsf{domain}(R, \exists R.\neg C \to \neg D)$
(RE3) Replace every GCI $(E \sqcap \exists R.C) \sqsubseteq D \in \mathcal{T}$ with $\mathsf{domain}(R, \exists R.C \to (E \to D))$

Remark 3. (RE1)–(RE3) are supported by *fuzzyDL* under Zadeh semantics since the reasoner allows the combination of $\to_Z$ (concept implication using Zadeh implication in the semantics) with a Zadeh KB. Furthermore, (RE2) is supported by *fuzzyDL* under Łukasiewicz semantics since it allows the combination of $\to_Z$ with a Łukasiewicz KB. The three previous rule also work if $\to_Z$ is replaced with $\to_G$. It is easy to check that the choice of $\to_G$ and $\to_Z$ does not have an impact on the number of absorbed axioms, and it does not seem to produce significant differences in practice because fuzzyDL handle the two implications similarly.

Proposition 3.3. *Role absorptions (RB1)–(RB2) and (RE1)–(RE3) transform an axiom into an equivalent one.*

Proof. See Appendix A. □

3.4. Concept absorptions

In this section, we will rewrite some TBox axioms using either simpler axioms (using GCI transformation rules) or primitive GCIs (using primitive concept absorptions). Table B.8 summarises the applicability of these rules. Firstly, we will recap the absorptions in the classical case and then we will extend them to fuzzy DLs.

3.4.1. Concept absorptions in classical DLs

For classical DLs we have the following definitions:

GCI transformation rules

(CT1) Replace $C \sqsubseteq C_1 \sqcap_G \ldots \sqcap_G C_n$ with $\{C \sqsubseteq C_i\}$, for $i \leq n$
(CT2) Replace $C_1 \sqcup_G \ldots \sqcup_G C_n \sqsubseteq C$ with $\{C_i \sqsubseteq C\}$, for $i \leq n$
(CT3) Replace $\exists R.\{o\} \sqsubseteq C$ with $o{:}\forall R^-.C$
(CT4) Replace $\{a_1, \ldots, a_n\} \sqcap C \sqsubseteq D$ with $\{a_i{:}\neg C \sqcup D\}$, for $i \leq n$

Primitive concept absorptions

(CA0) Absorb $A \sqsubseteq C$ to $A \sqsubseteq C$[8]
(CA1) Absorb $C \sqsubseteq \neg A$ to $A \sqsubseteq \neg C$
(CA2) Absorb $C \sqsubseteq \neg A \sqcup D$ to $A \sqsubseteq \neg C \sqcup D$
(CA3) Absorb $A \sqcap D \sqsubseteq C$ to $A \sqsubseteq C \sqcup \neg D$

3.4.2. Concept absorptions in fuzzy DLs

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 $i \leq n$

[8] (CA0) and (FA0) are not recursive: as we will see, they will move a GCI from $\mathcal{T}_g$ to $\mathcal{T}_{inc}$.

(FT3) Replace $\langle \exists R.\{o\} \sqsubseteq C, \alpha \rangle$ with $\langle o{:}\forall R^{-}.C, \alpha \rangle$
(FT4) Replace $\langle \{a_1, \dots, a_n\} \sqcap C \sqsubseteq D, \alpha \rangle$ with $\{\langle a_i{:}C \rightarrow D, \alpha \rangle\}$, for $i \leq n$

Primitive concept absorptions
(FA0) Absorb $\langle A \sqsubseteq C, \alpha \rangle$ to $\langle A \sqsubseteq C, \alpha \rangle$
(FA1) Absorb $\langle C \sqsubseteq \neg A, \alpha \rangle$ to $\langle A \sqsubseteq \neg C, \alpha \rangle$
(FA2.1) Absorb $\langle C \sqsubseteq \neg A \sqcup D, \alpha \rangle$ to $\langle A \sqsubseteq \neg C \sqcup D, \alpha \rangle$
(FA2.2) Absorb $\langle C \sqsubseteq A \rightarrow D, \alpha \rangle$ to $\langle A \sqsubseteq C \rightarrow D, \alpha \rangle$
(FA3) Absorb $\langle A \sqcap D \sqsubseteq C, \alpha \rangle$ to $\langle A \sqsubseteq D \rightarrow C, \alpha \rangle$

Remark 4. It is worth to note that (FT1)–(FT2) are applicable to Łukasiewicz, Product and classical DLs because $\sqcap_G$ and $\sqcup_G$ or representable as we have seen in Remark 1. Furthermore, (FT3) is supported by *fuzzyDL* under Zadeh semantics since it allows the combination of $\forall_Z$ (universal restriction using Zadeh implication in the semantics) with a Zadeh KB.

Proposition 3.4. *Transformations (FT1)–(FT4) and (FA0)–(FA3) transform an axiom into an equivalent set of axioms.*

Proof. Straightforward. □

3.5. GCI absorption algorithm

The input of the algorithm is a fuzzy KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, and the output is an equivalent $\mathcal{K}' = \langle \mathcal{A}', \mathcal{T}', \mathcal{R} \rangle$ with $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{T}' = \{\mathcal{T}_g \cup \mathcal{T}_{def} \cup \mathcal{T}_{inc} \cup \mathcal{T}_{dom} \cup \mathcal{T}_{rg} \cup \mathcal{T}_{disj} \cup \mathcal{T}_{syn}\}$. The algorithm has 3 phases: an initial Phase A, a looping Phase B and a final Phase C.

Phase A: The following steps are applied:

1. Simplify the GCIs in $\mathcal{T}$ using the concept simplifications in Section 3.1
2. Remove redundant GCIs listed in Section 3.2
3. Initialise $\mathcal{T}_g = \mathcal{T}_{def} = \mathcal{T}_{inc} = \mathcal{T}_{dom} = \mathcal{T}_{rg} = \mathcal{T}_{disj} = \mathcal{T}_{syn} = \emptyset$
4. Remove every domain (resp. range) axiom in $\mathcal{T}$ and add it to $\mathcal{T}_{dom}$ (resp. $\mathcal{T}_{rg}$)
5. Remove every disjointness axiom in $\mathcal{T}$ and add it to $\mathcal{T}_{disj}$
6. Remove every synonym axiom in $\mathcal{T}$ and add it to $\mathcal{T}_{syn}$
7. For every 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.[9] Once none of the above steps in Phase B can be applied any more, go to Phase C.

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

GCI transformations. If some of the GCI transformation rules in Section 3.4 can be applied to an axiom $\tau \in \mathcal{T}_g$, remove τ from $\mathcal{T}_g$ and add the obtained GCIs to $\mathcal{T}_g$.

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 GCI $\tau \in \mathcal{T}_g$ such that some of the primitive concept absorptions in Section 3.4 can be applied to it producing a rewriting axiom with an atomic concept in its head that is not defined in $\mathcal{T}_{def}$, then remove τ from $\mathcal{T}_g$ and add the rewriting of τ to $\mathcal{T}_{inc}$.

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 there is no $\mathsf{disjoint}(A, B) \in \mathcal{T}_{disj}$, where B is the head of some axiom in $\mathcal{T}_{def}$, then remove τ, τ' from the sets they are in and add $A \doteq C$ to $\mathcal{T}_{def}$.

[9] In this way, it is possible, for example, to remove redundant GCIs that are introduced by the absorption rules.

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

Phase C: Replace any GCI $\langle C \sqsubseteq D, \alpha \rangle \in \mathcal{T}_g$ with an equivalent GCI $\langle \top \sqsubseteq E, \alpha \rangle$ 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 $\mathcal{T}_u = \mathcal{T}_{def} \cup \mathcal{T}_{inc} \cup \mathcal{T}_{dr} \cup \mathcal{T}_{disj} \cup \mathcal{T}_{syn}$.

The following example illustrates a difference in absorption, depending on the semantics.

Example 3.1. Consider the TBox

$$\mathcal{T} = \{A \doteq B \sqcup C, A \sqsubseteq D\}.$$

It can be verified that under classical and Zadeh logics, the absorbed TBox is given by

$$\mathcal{T}_{inc} = \{C \sqsubseteq A, B \sqsubseteq A, A \sqsubseteq D, A \sqsubseteq B \sqcup C\}$$
$$\mathcal{T}_g = \mathcal{T}_{def} = \mathcal{T}_{disj} = \mathcal{T}_{syn} = \mathcal{T}_{dr} = \emptyset.$$

Remark 5. Example 3.1 illustrates the usefulness of giving less priority to the "definition absorption" step. If this step had a highest priority, the ontology in Example 3.1 would not be absorbed completely.

Next, we will provide an example that will help to illustrate the theoretical significance of our absorption algorithm.

Example 3.2. Consider the TBox

$$\mathcal{T} = \{A \sqcap B \sqsubseteq C\}.$$

under Łukasiewicz semantics. Then the absorbed TBox is

$$\mathcal{T} = \{A \sqsubseteq B \rightarrow C\}.$$

Remark 6. Example 3.2 shows that a non-unfoldable KB may be transformed into an unfoldable one as defined in [17]. Thus, after the application of our absorption algorithm, we may transform an ontology, for which we do not know whether the satisfiability problem is decidable or not, into another form for which we do know that the satisfiability problem is decidable [17,18].

Now, as all our concept and GCI transformations are equivalence preserving, the following proposition can easily be shown.

Proposition 3.5. *Consider $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ and let be $\mathcal{K} = \langle \mathcal{A}', \mathcal{T}', \mathcal{R} \rangle$ be the KB obtained from $\mathcal{K}$ by applying the absorption algorithm. Then $\mathcal{K} \equiv \mathcal{K}'$.*

Eventually, note that we do not create any new RBox axiom or disjointness axiom. Specifically, one could have considered a disjointness absorption step such as the following one, but our algorithm applies primitive absorption instead.

Disjointness absorption. If there is $\tau \in \mathcal{T}_g$ that is of the form $\langle A \sqcap_G B \sqsubseteq \perp, \alpha \rangle$, A and B are not both head of axioms in $\mathcal{T}_{def}$, then remove τ from $\mathcal{T}_G$ and add $\mathsf{disjoint}(A, B)$ to $\mathcal{T}_{disj}$.

4. Evaluation and discussion

Our research hypothesis is that our absorption algorithm may absorb a non-negligible number of GCIs in a fuzzy ontology, resulting in a reduction in the reasoning time. To this end, we have implemented the fuzzy GCI absorption algorithm in the fuzzy ontology reasoner *fuzzyDL*, under Zadeh, Łukasiewicz, and classical logics. Currently, *fuzzyDL* supports fuzzy $\mathcal{SHIBF}_g(\mathbf{D})$ extended with many other specific fuzzy DLs constructors [24], which are not of interest in our experimentation.

Our evaluation considers a dataset of 51 ontologies, detailed in Section 4.1, and can be divided in two parts. Section 4.2 discusses the impact of our absorption algorithm in ontologies (i.e., how many ontologies can benefit from it) and Section 4.3 discusses the impact of absorption on the reasoning time.

4.1. Dataset

To evaluate our algorithm, we have selected 49 well-known classical ontologies, encoded in OWL 2 [29]. We also built 2 fuzzy ontologies, encoded in *FuzzyOWL 2* [33].[10] The classical ontologies have been downloaded from the TONES OWL ontology repository.[11]

The ontologies in our dataset are detailed in Table 3, including its identification number (column ID), its name (column ontology), and some relevant features, namely the expressivity (column expressivity) and the number of classes (column classes), primitive GCIs (column AisC), definitional GCIs (column $A \doteq C$), domain restrictions (column domain), range restrictions (column range), general concept inclusions (column GCIs), and disjoint restrictions (column disj). We can see that the features may vary significantly among the ontologies. As it is usual in practice, none of the ontologies has synonyms (for this reason a column for the case $A \doteq B$ was unnecessary) and few have GCIs not being primitive or definitional. In the following, to indicate an ontology we will just use its ID.

The two fuzzy ontologies in our dataset are ID4 and ID17. They do not contain graded axioms, but do have axioms involving fuzzy concrete domains, such as the axiom described in Example 2.1. ID17 uses also some other fuzzy DLs specific constructors not covered here [24]. In ID4 the semantics has been fixed by construction to be Zadeh semantics, while in ID17 the semantics is Łukasiewicz.

4.2. Impact of the absorption on the ontologies

Let us now detail our experiments to evaluate the applicability of our absorption algorithm in practice. At first, we run our algorithm with all the ontologies in Table 3 w.r.t. classical, Zadeh and Łukasiewicz semantics. Since none of these ontologies have graded axioms, we also designed some experiments to measure the impact of graded inclusion axioms on the absorption algorithm. With this aim, we generated automatically graded GCIs, expressed as Fuzzy OWL 2 axioms, from non-graded GCIs, and again run our algorithm w.r.t. Zadeh and Łukasiewicz semantics. Specifically, Algorithm *RandomGCIs*($\mathcal{T}, p$) was used to generate random graded GCIs in the following way. The algorithm receives a TBox $\mathcal{T}$ and a parameter $p \in [0, 1]$ indicating the probability that every inclusion axiom in $\mathcal{T}$ becomes graded.

Algorithm *RandomGCIs*($\mathcal{T}, p$)
Input: TBox $\mathcal{T}$, the probability to generate a graded GCI $p \in [0, 1]$
Output: Graded TBox $\mathcal{T}'$ with a percentage of graded inclusion axioms p.

$\mathcal{T}' \leftarrow \emptyset$
for each GCI $\tau \in \mathcal{T}$
  if τ is graded or τ is definitional
    then $\mathcal{T}' \leftarrow \mathcal{T}' \cup \{\tau\}$
    else generate a random number $\gamma \in [0, 1]$
      if $\gamma > p$
        then $\mathcal{T}' \leftarrow \mathcal{T}' \cup \{\tau\}$
        else generate a random number $\alpha \in [0, 1]$
          $\mathcal{T}' \leftarrow \mathcal{T}' \cup \{\langle \tau, \alpha \rangle\}$
return $\mathcal{T}'$

[10] http://www.straccia.info/software/FuzzyOWL.
[11] http://rpc295.cs.man.ac.uk:8080/repository.

Table 3
Ontology dataset and statistics.

ID	Ontology	Expressivity	Classes	A is C	$A \doteq C$	Domain	Range	GCIs	disj
1	Transportation	$\mathcal{ALCH}(\mathbf{D})$	444	452		81	72		317
2	Economy	$\mathcal{ALCH}(\mathbf{D})$	337	409		47	42		71
3	lubm	$\mathcal{ALEHI}(\mathbf{D})$	43	36	6	25	18		
4	FuzzyWine	$\mathcal{SHIF}(\mathbf{D})$	177	218	57	11	8	9	1
5	FBbt_XP	$\mathcal{SHI}$	7225	12 043	1028	8	6		63
6	galen-ians-full-doctored	$\mathcal{ALEHIF}$	2748	2881	699			357	
7	process	$\mathcal{SHIFB}(\mathbf{D})$	2294	3088	247	169	155		1
8	NCI	$\mathcal{ALE}$	27 652	46 800		70	70		
9	FMA	$\mathcal{ALEH}$	78 983	121 708					
10	chebi	$\mathcal{ALE}$	20 979	38 375					
11	spatial.obo	$\mathcal{ELH}$	106	116	58				
12	thesaurus	$\mathcal{ALCH}(\mathbf{D})$	65 231	83 644	10 242	103	97		171
13	biochemistry-complex	$\mathcal{ALC}$	1051	1344	151	64	65		1700
14	worm_phenotype_xp.obo	$\mathcal{EL}$	1841		1173				
15	ontology	$\mathcal{ALCHIF}$	475	567	125	6	6		1247
16	mosquito_insecticide_resistance	$\mathcal{ALE}$	524 039	658 390	46				2
17	matchmaking	$\mathcal{ALCHF}(\mathbf{D})$	107	80	19	36	6		
18	people.fd	$\mathcal{ALCI}$	59	33	20	2	4		3
19	pathway.obo	$\mathcal{ALE}$	600	696					
20	fmaOwlDlComponent_1_4_0	$\mathcal{ALCIFB}$	6487	18 477		12	71		
21	propreo	$\mathcal{SHIFB}(\mathbf{D})$	400	480	27	30	24		
22	earthrealm	$\mathcal{SHIFB}(\mathbf{D})$	2294	3088	247	169	155		1
23	so-xp.obo	$\mathcal{SHI}$	1660	1709	198				21
24	cancer_my	$\mathcal{ALCHF}(\mathbf{D})$	88	82	36	13	12		20
25	cton	$\mathcal{SHF}$	17 032	33 060	86				21 553
26	goslim	$\mathcal{AL}$	161	79	79				
27	photography	$\mathcal{SHIF}(\mathbf{D})$	188	237	45			1	242
28	gene_ontology_edit.obo	$\mathcal{SH}$	26 225	42 650					3
29	periodic-table-complex	$\mathcal{ALU}$	181	168	46				
30	amino-acid	$\mathcal{ALCF}(\mathbf{D})$	46	238	12	5	5		199
31	yowl-complex	$\mathcal{SHIF}(\mathbf{D})$	336	271	116	54	55		141
32	atom-common	$\mathcal{ALCHI}$	14	13	2	5	5		55
33	GRO	$\mathcal{ALCIF}(\mathbf{D})$	419	636	60	4	6		96
34	chemistry-complex	$\mathcal{SHIF}(\mathbf{D})$	790	1080	149	64	65		1578
35	mygrid-moby-service	$\mathcal{ALCHIF}(\mathbf{D})$	504	591	125	42	17		1247
36	time-modification	$\mathcal{ALUHIF}(\mathbf{D})$	143	226	39	65	62		19
37	EMAP.obo	$\mathcal{ALE}$	13 731	13 730					
38	teleost_taxonomy.obo	$\mathcal{AL}$	36 076	36 069					
39	heart	$\mathcal{SHI}$	75	224	30	28	29		2
40	relative-places	$\mathcal{SHIF}$	16	18	4	15	14		2
41	po	$\mathcal{SHIFB}(\mathbf{D})$	91	79	18	131	79		24
42	SIGKDD-EKAW	$\mathcal{SHIF}(\mathbf{D})$	115	134	7	51	40		74
43	legal-action	$\mathcal{ALC}$	100	175	39	53	52		19
44	pizza	$\mathcal{SHIFB}$	99	259	13	6	7		398
45	AirSystem	$\mathcal{SHIF}$	113	210	10	114	104		29
46	norm	$\mathcal{SHI}$	154	249	55	61	60		21
47	organic-compound-complex	$\mathcal{ALCHI}$	71	69	37	5	5		56
48	reaction	$\mathcal{SHIF}$	77	73	20	8	8		54
49	chemical	$\mathcal{ALCHF}(\mathbf{D})$	48	46	18	18	8		6
50	subatomic-particle-complex	$\mathcal{SHIF}(\mathbf{D})$	94	93	15	50	50		112
51	PRO	$\mathcal{SH}$	26 017	21 758	4505				681

To test the impact of graded axioms on the absorption algorithm, we considered for each of the 2 fuzzy semantics (namely Zadeh and Łukasiewicz), the 3 cases $p \in \{0.33, 0.66, 1.0\}$, indicating that one third, two thirds, and all inclusion axioms are graded, respectively. All ontologies, whether in OWL 2 format or Fuzzy OWL 2 format, where translated by a parser [33] into *fuzzyDL* syntax discarding the expressions that *fuzzyDL* is not able to process. The

Table 4
Number of non-absorbed ontologies.

c	z	l	z.33	l.33	z.66	l.66	z.100	l.100	Total
2	2	22	2	22	2	23	2	23	100
4%	4%	43%	4%	43%	4%	45%	4%	45%	22%

statistics illustrated in Table 3 refer to the *fuzzyDL* processable variant of the ontologies. All the evaluation dataset, which includes the original (fuzzy) OWL 2 ontologies, the randomly generated Fuzzy OWL 2 ontologies, their translation into *fuzzyDL*, and all the information and software to replicate the experiments, can be freely downloaded.[12]

To avoid unnecessary processing time, *fuzzyDL* checks first if the ontology is already lazy unfoldable. If this is the case, then the ontology is left as it is, otherwise the absorption algorithm is run over it. For each ontology, we run 9 tests (3 non-graded and 6 graded) and, thus, overall run 459 tests. The tests were run under a computer with Mac OS X 10.7.5, Mac Pro $2x3$ GHz Dual-Core Intel Xeon, and 9 GB Ram.

The results of the algorithm runs are reported in Tables C.9–C.17.

- The first column shows the ontology ID.
- The second column shows the logic considered in the semantics: z denotes Zadeh, l denotes Łukasiewicz and c denotes classical logic. We also use the label $f.n$, with $f \in \{z, l\}$, $n \in \{33, 66, 100\}$, to denote that the logic f has been chosen and that the TBox has been randomised with parameter $p = n/100$.
- The next seven columns show the size of each of the parts of the TBox.
- The column "LU" indicates if the lazy unfoldable check succeeded before starting the absorption algorithm.
- The column t_{abs} shows the running time (in seconds) of the absorption algorithm. The running time includes the time to read the ontology from the file. If "LU" is marked, the time only included the time to read the ontology and the lazy unfoldable check; otherwise, the execution time of the absorption algorithm is also added.

Recall that the ontologies that have completely been absorbed are identified with $\mathcal{T}_g = \emptyset$. In the following, for

$$x \in \{g, u, inc, def, syn, dom, rg, disj\}$$
$$y \in \{c, z, l, z.33, l.33, z.66, l.66, z.100, l.100\},$$

with $\mathcal{T}_x^y$ we will indicate the TBox $\mathcal{T}_x$ returned by the absorption algorithm under the case y. With $\bar{\mathcal{T}}$ we also indicate the TBox obtained from $\mathcal{T}$ by replacing all graded axioms $\langle\tau, \alpha\rangle \in \mathcal{T}$ with their non-graded variant τ. Additionally, if $\mathcal{K}$ is an ontology, with $\mathcal{K}'$ we denote the ontology obtained from $\mathcal{K}$ in which some non-graded GCIs $\tau \in \mathcal{K}$ have been replaced with a graded variant $\langle\tau, \alpha\rangle$.

A first observation is that whenever a classical ontology $\mathcal{K}$ is already lazy unfoldable (LU column flagged), then so is $\mathcal{K}'$. More generally, we can easily prove the following result.

Proposition 4.1. *If a classical ontology $\mathcal{K}$ is already lazy unfoldable before running the absorption algorithm, then so are also $\mathcal{K}$ and $\mathcal{K}'$ under any fuzzy semantics.*

Consequently, we do not further analyse those ontologies with the "LU" column flagged. For the other ones, Table 4 resumes the number of ontologies that we were not able to absorb. As we can see, the success percentage (78% overall) is non-negligible. Hence, the first part of our research hypothesis is verified.

Now we will study separately the cases of classical and Zadeh semantics, and the case of Łukasiewicz semantics. Then, we will enumerate some common non-absorbable patterns that we were able to identify.

The cases of classical and Zadeh semantics It is interesting to note that an ontology is absorbable under classical semantics if and only if it is absorbable according to Zadeh semantics. In fact, from the absorption rule conditions, the following propositions can be shown.

[12] http://straccia.info/ftp/absorption.test.zip.

Proposition 4.2. *Consider a non-graded inclusion axiom τ. Then the following are equivalent:*

1. *we can absorb τ under classical semantics;*
2. *we can absorb τ under Zadeh semantics;*
3. *we can absorb $\langle \tau, \alpha \rangle$ under Zadeh semantics.*

We get immediately then that for $y \in \{z.33, z.66, z.100\}$

$$\mathcal{T}_g^c = \mathcal{T}_g^z = \bar{\mathcal{T}}_g^y.$$

The following proposition is a more general result.

Proposition 4.3. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an ontology and let $\mathcal{K}' = \langle \mathcal{A}, \mathcal{T}', \mathcal{R} \rangle$, where $\mathcal{T}'$ is as $\mathcal{T}$ except that some GCI $\tau \in \mathcal{T}$ has been replaced with a graded variant $\langle \tau, \alpha \rangle$. Then*

$$\mathcal{T}_g^c = \mathcal{T}_g^z = \bar{\mathcal{T}}'_g.$$

Therefore, we can formulate the following proposition.

Proposition 4.4. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an ontology and let $\mathcal{K}' = \langle \mathcal{A}, \mathcal{T}', \mathcal{R} \rangle$, where $\mathcal{T}'$ is as $\mathcal{T}$ except that some non-graded GCI $\tau \in \mathcal{T}$ has been replaced with a graded variant $\langle \tau, \alpha \rangle$. Then the following are equivalent:*

1. *we can absorb $\mathcal{K}$ under classical semantics;*
2. *we can absorb $\mathcal{K}$ under Zadeh semantics;*
3. *we can absorb $\mathcal{K}'$ under Zadeh semantics.*

We were able to absorb 49 ontologies, i.e., all ontologies except 2, namely ID21 and ID49.

The case of Łukasiewicz semantics In this case, we can show the following result.

Proposition 4.5. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an ontology and assume Łukasiewicz semantics.*

If $\mathcal{T}_g^l = \emptyset$ then $\mathcal{T}_g^c = \emptyset$.

The converse does not hold, as it happens for example in ID27, where $\mathcal{T}_g^c = \emptyset = \mathcal{T}_g^l = \mathcal{T}_g^{l.33}$, but $\mathcal{T}_g^{l.66} \neq \emptyset$.

Non-absorbed GCI patterns We conclude this section by illustrating the GCI patterns that we were not able to absorb.

Pattern 1. Consider a TBox containing

$$A \doteq \forall R.C$$
$$A \sqsubseteq B$$

with B atomic. This pattern occurs in ID49 as

$$\texttt{VR\_RelatedPublishedWork} \doteq \forall \texttt{refersToPrecursor.VR\_Precursor}$$
$$\texttt{VR\_RelatedPublishedWork} \sqsubseteq \texttt{NerveAgentRelatedPublishedWork}.$$

In any semantics setting, our algorithm is not able to absorb the pattern. Specifically, the algorithm generates

$$\{A \sqsubseteq \forall R.C, A \sqsubseteq B\} \subseteq \mathcal{T}_{inc}$$
$$\top \sqsubseteq \forall R.C \rightarrow A \in \mathcal{T}_g.$$

Pattern 2. Consider a TBox containing

$$\langle \exists R.A \sqsubseteq \exists S.B, \alpha \rangle$$

with $\alpha < 1$. This pattern occurs in ID27, cases $l.66$ and $l.100$ (in the $l.33$ case the axiom was not graded). Specifically, in ID27.l.66 the pattern occurs as

$$\langle \exists \texttt{increases.LightIntensity} \sqsubseteq \exists \texttt{increases.ExposureLevel}, 0.46 \rangle.$$

This pattern happens in Łukasiewicz logic with graded axioms. While for $\alpha = 1$ our algorithm absorbs it using rule (RE1), the rule does not apply for the $\alpha < 1$ case.

It remains to see whether one may work out some kind of graded domain and range axioms of the form $\mathsf{domain}(R, C, \alpha)$ and $\mathsf{range}(R, C, \alpha)$, with semantics $\langle \exists R.\top \sqsubseteq C, \alpha \rangle$ and $\langle \top \sqsubseteq \forall R.C, \alpha \rangle$, respectively, and in which cases they would allow to absorb GCIs of the above form.

Pattern 3. Consider a TBox containing

$$A \doteq B_1 \sqcup B_2$$
$$A \sqsubseteq B_3$$

with B_i atomic. This pattern occurs in the remaining non-absorbable ontologies, namely ID13, ID24, ID25, ID29, ID30, ID31, ID32, ID34, ID36, ID39, ID40, ID41, ID43, ID44, ID45, ID46, ID47, ID48, ID50, and ID51.

For instance, in ID24, the pattern occurs as

$$\texttt{CarbonGroup} \doteq \texttt{Alkyl} \sqcup \texttt{Aryl}$$
$$\texttt{CarbonGroup} \sqsubseteq \texttt{OrganicGroup}$$

While for classical and Zadeh logics our algorithm absorbs the patterns as (see also Example 3.1)

$$\{A \sqsubseteq B_1 \sqcup B_2, A \sqsubseteq B_3, B_1 \sqsubseteq A, B_2 \sqsubseteq A\} \subseteq \mathcal{T}_{inc},$$

this is no longer possible under Łukasiewicz logic. The same problem happens for instance in Product logic.

4.3. Impact of absorption on reasoning

We also conducted a preliminary evaluation on the impact of the absorption algorithm on reasoning time. Specifically, for each ontology in Table 3, we manually selected 2 subsumption problems: one for which the subsumption holds, and another one for which the subsumption does not hold. For each ontology, we submitted the 2 subsumption problems for the 9 cases $f \in \{c, l, z\}$, and $f.n$, with $f \in \{l, z\}$, $n \in \{33, 66, 100\}$. We set a timeout limit of 5 minutes.

The results are reported in the column t_{subs} of Tables C.9–C.17. The value refers to the average time of the two subsumption tests, where *TO* means that at least one problem exceeded the timeout limit. t_{subs} measures only the reasoning time after excluding the absorption time t_{abs}.

We also run the subsumption tests by switching off the absorption algorithm, so that all GCIs have been put into $\mathcal{T}_g$, i.e., all GCIs have been transformed into the form $\langle \top \sqsubseteq C, n \rangle$. We used the same timeout limit.

The complete results are reported in the t_{noAbs} column of Tables C.9–C.17. Similarly as for t_{subs}, t_{noAbs} measures only the reasoning time by excluding the ontology reading time: the time measurement starts exactly at the same moment as for t_{subs} in order to make these two measurements comparable.

Table 5 summarises the results. We differentiate between the ontologies that are lazy unfoldable since the beginning (those with LU flag), absorbable ontologies in classical and Zadeh, absorbable ontologies in classical, Zadeh, and Łukasiewicz, and non-absorbable ontologies in any of the three cases. For each of these cases, we indicate whether there is a timeout always (regardless of whether the ontology is absorbed or not), there is a timeout only if there is no absorption, or there is never a timeout.

As we can see in Tables C.9–C.17, despite that *fuzzyDL* has not yet been optimised neither for instance checking problems nor for subsumption problems, the subsumption problems, when solved, have been solved in no more than 2 seconds (most of the times, in a fraction of second). The only exceptions are ID30 (for the case $f = l$) and ID44.

Table 5
Summary of the results.

	Timeout always	No timeout if absorbed	No timeout ever
Lazy unfoldable	1 (ID20)	14 (ID1, ID2, ID7, ID8, ID9, ID10, ID12, ID14, ID17, ID19, ID22, ID28, ID37, ID38)	1 (ID26)
Absorbable c, l, z	5 (ID5, ID6, ID15, ID33, ID35)	7 (ID3, ID4, ID11, ID16, ID18, ID23, ID42)	0
Absorbable c, z	10 (ID13, ID27, ID31, ID34, ID36, ID39, ID43, ID45, ID46, ID47)	7 (ID24, ID25, ID30, ID41, ID44, ID48, ID51)	4 (ID29, ID32, ID40, ID50)
Not absorbable	2 (ID21, ID49)	0	0

These cases, together with the unresolved cases, indicate that we need further to develop optimised subsumption and instance checking algorithms.

As expected, without the absorption *fuzzyDL* failed to complete most of the tests. Therefore, the benefit of the absorption algorithm seems to be non-negligible on reasoning time.[13]

Remark 7. There are 5 ontologies for which the subsumption tests succeeded within the imposed timeout limit, namely ID26, ID29, ID32, ID40, and ID50, even without running the absorption algorithm. However, let us point out that in these cases, the constructed *completion-forest* [1,20] is relatively small, which motivates the low running time. Indeed, the following observations regarding the $\exists R.C$ constructor, which is one of the main sources of complexity in DLs, can help to understand this behaviour:

- in ID26, ID29, and ID32 the constructor $\exists R.C$ does not occur;
- in ID50 the constructor $\exists R.C$ appears once;
- ID40 has some $\exists R.C$ constructors, but no $\exists R.C$ is involved in the execution of the subsumption test.

As a final consideration, it is useful to consider what one could expect from other ontologies not included in our dataset. Since our ontologies have different features, we believe that our results seem significant enough. However, the application of our absorption procedure to other ontologies may still give rise to other absorbable patterns that we have not yet identified. However, the design of our algorithm makes it easy to plug such patterns into our absorption algorithm.

5. Conclusions

In this paper, we have presented the first absorption algorithm for fuzzy DLs. From a theoretical point of view, our absorption algorithm may transform ontologies for which we do not know in advance whether a decision problem is decidable or not, into another form for which we do know that the decision problem at hand is decidable: this happens for example in Example 3.2.

To understand the effectiveness of our absorption algorithm, we have implemented our algorithm into the *fuzzyDL* reasoner and evaluated it over 51 ontologies. Our results are very encouraging and show that this optimisation may dramatically decreases the inference time. We were also able to identify some patterns that our algorithm is not able to absorb yet.

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

[13] Of course, a deeper investigation is necessary here.

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 [1,10–12,14–16].

Classification. While we already have implemented in *fuzzyDL* all fuzzy lazy unfolding rules related to absorbed TBoxes and preliminary subsumption tests perform very fast, a non-trivial optimised fuzzy classification algorithm in the style of [35] 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 investigate are optimising 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 optimising instance retrieval, i.e., determining the set $ans(\mathcal{K}, C) = \{\langle a, \alpha \rangle \mid \alpha = bed(\mathcal{K}, a{:}C)\}$ [8,9,36–39]. In that direction, *fuzzyDL* already implements a fuzzy variant of anywhere blocking [40].

Acknowledgements

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

Appendix A. Selection of proofs

In this appendix we include some proofs of the results shown in the paper. We will show that role absorptions (RE1) and (RE3) are correct for Gödel DLs, and that (RE2) is correct for Łukasiewicz and Zadeh DLs extended with Zadeh implication concept constructor (a combination supported by *fuzzyDL*).

(RE1) In Gödel DLs, $C \sqsubseteq D$ is the same as $\top \sqsubseteq C \to D$. So, we can rewrite $\exists R.C \sqsubseteq D \in \mathcal{T}$ as $\top \sqsubseteq \exists R.C \to D$ and $\exists R.\top \sqsubseteq \exists R.C \to D$ as $\top \sqsubseteq \exists R.\top \to (\exists R.C \to D)$. The latter corresponds to $\mathsf{domain}(R, \exists R.C \to D)$. We have to show that $\top \sqsubseteq \exists R.C \to D$ and $\top \sqsubseteq \exists R.\top \to (\exists R.C \to D)$ are equivalent.

$\Rightarrow$) By assumption, for any v, $(\exists R.C \to D)(v) = 1$. But then obviously $(\exists R.\top \to (\exists R.C \to D))(v) = 1$ for any r-implication $\to$.

$\Leftarrow$) By assumption, for any v, $(\exists R.\top \to (\exists R.C \to D))(v) = 1$ and thus, $(\exists R.\top)(v) \leq (\exists R.C \to D)(v)$. Now, for some w,

$$(\exists R.C)(v) = R(v, w) \otimes C(w) \leq (\exists R.\top)(v) \leq (\exists R.C \to D)(v),$$

i.e.,

$$(\exists R.C)(v) \leq (\exists R.C \to D)(v), \tag{A.1}$$

which is of the form

$$x \leq x \to y.$$

Suppose that $y < x$. Then $y < x \leq x \to y = y$, which is absurd. Thus, $x \leq x \to y$ implies $x \leq y$. Applied to Eq. (A.1), we get

$$(\exists R.C)(v) \leq D(v),$$

that is, $(\exists R.C \to D)(v) = 1$, which concludes.

(RE2) In Łukasiewicz and Zadeh fuzzy DLs, $C \sqsubseteq D$ is equivalent to $\neg D \sqsubseteq \neg C$. Hence, $D \sqsubseteq \forall R.C$ is equivalent to $\neg \forall R.C \sqsubseteq \neg D$. Using inter-definability of universal and existential restrictions, the latter is equivalent to $\exists R.\neg C \sqsubseteq \neg D$. Now, we can apply extended role absorption (RE1) and replace this GCI with $\mathsf{domain}(R, \exists R.\neg C \to_G \neg D)$.

(RE3) Similarly as in (FA3), $(E \sqcap \exists R.C) \sqsubseteq D$ is equivalent to $\exists R.C \sqsubseteq E \to D$ to which we apply then rule (RE1).

Table B.6
Supported concept simplifications.

ID	classical	*fuzzyDL*	Any	Ł	G	*Π*	Z
(CS1)	✓	✓	✓	✓	✓	✓	✓
(CS2)	✓	✓	✓	✓	✓	✓	✓
(CS3)	✓	✓	✓	✓	✓	✓	✓
(CS4)	✓	✓	✓	✓	✓	✓	✓
(CS5)	✓	✓	✓	✓	✓	✓	✓
(CS6)	✓	✓	✓	✓	✓	✓	✓
(CS7)	✓	✓	✓	✓	✓	✓	✓
(CS8)	✓	✓	✓	✓	✓	✓	✓
(CS9)	✓	✓	✓	✓	✓	✓	✓
(CS10)	✓	✓	✓	✓	✓	✓	✓
(CS11)	✓	✓		✓			✓
(CS12)	✓	✓	✓	✓	✓	✓	✓
(CS13)	✓	✓	✓	✓	✓	✓	✓
(CS14)	✓	✓		✓			✓
(CS15)	✓	✓		✓			✓
(CS16)	✓	✓		✓			✓
(CS17)	✓	✓		✓			✓
(CS18)	✓	✓		✓	✓	✓	
(CS19)	✓	✓		✓			
(CS20)	✓	✓			✓		✓
(CS21)	✓	✓			✓		✓
(CS22)	✓	✓			✓		
(CS23)	✓	✓			✓		
(CS24)	✓	✓	✓	✓	✓	✓	✓
(CS25)	✓	✓			✓		✓
(CS26)	✓	✓	✓	✓	✓	✓	✓
(CS27)	✓	✓	✓	✓	✓	✓	✓
(CS28)	✓	✓	✓	✓	✓	✓	✓
(CS29)	✓	✓	✓	✓	✓	✓	✓
(CS30)	✓	✓	✓	✓	✓	✓	✓
(CS31)	✓	✓	✓	✓	✓	✓	✓
(CS32)	✓	✓	✓	✓	✓	✓	✓
(CS33)	✓	✓	✓	✓	✓	✓	✓
(CS34)	✓	✓	✓	✓	✓	✓	✓
(CS35)	✓	✓	✓	✓	✓	✓	

Table B.7
Supported role absorptions.

ID	Classical	*fuzzyDL*	Any	Ł	G	*Π*	Z
(RB1)	✓	✓	✓	✓	✓	✓	✓
(RB2)	✓	✓	✓	✓	✓	✓	✓
(RE1)	✓	✓			✓		
(RE2)	✓	✓					
(RE3)	✓	✓			✓		

Appendix B. Applicability of our results

This appendix summarises for which fuzzy logics apply our results. Apart from the classical case, and the four main fuzzy DLs (Ł, G, Π, and Z), we consider any t-norm based DL (denoted *any*, with $\rightarrow$ interpreted as the residuum of the t-norm), and the language supported by the *fuzzyDL* reasoner (denoted *fuzzyDL*). *fuzzyDL* supports arbitrary combinations of operators from Łukasiewicz and Zadeh logics, but also Zadeh and Gödel implications as concept constructors. Table B.6 sums up the concept simplifications, Table B.7 the role absorptions, and Table B.8 compiles the GCIs simplifications and concept absorptions.

Table B.8
Supported concept absorptions.

ID	Classical	*fuzzyDL*	Any	Ł	G	*Π*	Z
(CT1)	✓	✓					
(CT2)	✓	✓					
(CT3)	✓	✓					
(CT4)	✓	✓					
(CA0)	✓	✓					
(CA1)	✓	✓					
(CA2)	✓	✓					
(CA3)	✓	✓					
(FT1)		✓	✓	✓	✓	✓	✓
(FT2)		✓	✓	✓	✓	✓	✓
(FT3)		✓	✓	✓	✓	✓	
(FT4)		✓	✓	✓	✓	✓	✓
(FA0)		✓	✓	✓	✓	✓	✓
(FA1)		✓		✓			✓
(FA2.1)		✓		✓			
(FA2.2)		✓	✓	✓	✓	✓	
(FA3)		✓	✓	✓	✓	✓	

Table C.9
Results of the absorption algorithm: ID1–ID3.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
1	c	452			81	72		317	✓	0.023	0.013	TO
	z	452			81	72		317	✓	0.024	0.013	TO
	l	452			81	72		317	✓	0.021	0.013	TO
	z.33	452			81	72		317	✓	0.021	0.013	TO
	l.33	452			81	72		317	✓	0.023	0.012	TO
	z.66	452			81	72		317	✓	0.025	0.013	TO
	l.66	452			81	72		317	✓	0.026	0.013	TO
	z.100	452			81	72		317	✓	0.024	0.013	TO
	l.100	452			81	72		317	✓	0.025	0.012	TO
2	c	409			47	42		71	✓	0.106	0.016	TO
	z	409			47	42		71	✓	0.099	0.016	TO
	l	409			47	42		71	✓	0.086	0.017	TO
	z.33	409			47	42		71	✓	0.098	0.015	TO
	l.33	409			47	42		71	✓	0.085	0.018	TO
	z.66	409			47	42		71	✓	0.098	0.015	TO
	l.66	409			47	42		71	✓	0.089	0.018	TO
	z.100	409			47	42		71	✓	0.097	0.016	TO
	l.100	409			47	42		71	✓	0.084	0.017	TO
3	c	54			25	18				1.944	0.013	TO
	z	54			25	18				2.012	0.014	TO
	l	48			25	18				1.664	0.015	TO
	z.33	54			25	18				2.486	0.015	TO
	l.33	48			25	18				1.743	0.014	TO
	z.66	54			25	18				2.219	0.015	TO
	l.66	48			25	18				1.688	0.016	TO
	z.100	54			25	18				2.188	0.015	TO
	l.100	48			25	18				1.772	0.016	TO

Appendix C. Detailed results of our experiments

This section contains the complete results of the absorption algorithm runs, detailed in Tables C.9–C.17.

Table C.10
Results of the absorption algorithm: ID4–ID9.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
4	c	394			18	8		1		0.11	0.196	TO
	z	394			18	8		1		0.112	0.200	TO
	l	394			18	8		1		0.109	0.194	TO
	z.33	394			18	8		1		0.114	0.196	TO
	l.33	394			18	8		1		0.109	0.198	TO
	z.66	394			18	8		1		0.114	0.200	TO
	l.66	394			18	8		1		0.112	0.199	TO
	z.100	394			18	8		1		0.111	0.198	TO
	l.100	394			18	8		1		0.106	0.198	TO
5	c	15 175			8	6		63		2.021	TO	TO
	z	15 175			8	6		63		1.664	TO	TO
	l	14 099			8	6		63		1.913	TO	TO
	z.33	15 175			8	6		63		1.122	TO	TO
	l.33	14 099			8	6		63		1.028	TO	TO
	z.66	15 175			8	6		63		1.181	TO	TO
	l.66	14 099			8	6		63		1.31	TO	TO
	z.100	15 175			8	6		63		1.128	TO	TO
	l.100	14 099			8	6		63		1.25	TO	TO
6	c	5507		18						1.768	TO	TO
	z	5507		18						1.729	TO	TO
	l	4600		18						1.863	TO	TO
	z.33	5507		18						1.624	TO	TO
	l.33	4600		18						1.852	TO	TO
	z.66	5507		18						1.627	TO	TO
	l.66	4600		18						1.801	TO	TO
	z.100	5507		18						1.581	TO	TO
	l.100	4600		18						1.825	TO	TO
7	c	3084	12	235	169	155		1	✓	0.297	1.241	TO
	z	3084	12	235	169	155		1	✓	0.307	1.216	TO
	l	3084	12	235	169	155		1	✓	0.301	1.296	TO
	z.33	3084	12	235	169	155		1	✓	0.289	1.192	TO
	l.33	3084	12	235	169	155		1	✓	0.253	1.130	TO
	z.66	3084	12	235	169	155		1	x	0.848	1.061	TO
	l.66	3084	12	235	169	155		1	✓	0.762	0.972	TO
	z.100	3084	12	235	169	155		1	✓	0.862	1.513	TO
	l.100	3084	12	235	169	155		1	✓	0.785	1.186	TO
8	c	46 800			70	70			✓	0.378	0.184	TO
	z	46 800			70	70			✓	0.408	0.163	TO
	l	46 800			70	70			✓	0.424	0.181	TO
	z.33	46 800			70	70			✓	0.774	0.135	TO
	l.33	46 800			70	70			✓	0.678	0.19	TO
	z.66	46 800			70	70			✓	0.794	0.149	TO
	l.66	46 800			70	70			✓	0.746	0.173	TO
	z.100	46 800			70	70			✓	0.471	0.143	TO
	l.100	46 800			70	70			✓	0.473	0.198	TO
9	c	121 708							✓	0.627	0.734	TO
	z	121 708							✓	0.617	0.659	TO
	l	121 708							✓	0.656	0.809	TO
	z.33	121 708							✓	0.645	0.673	TO
	l.33	121 708							✓	0.626	0.723	TO
	z.66	121 708							✓	0.608	0.827	TO
	l.66	121 708							✓	0.575	0.807	TO
	z.100	121 708							✓	0.681	0.82	TO
	l.100	121 708							✓	0.666	0.763	TO

Table C.11
Results of the absorption algorithm: ID10–ID15.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
10	c	38 375							✓	0.491	0.161	TO
	z	38 375							✓	0.53	0.163	TO
	l	38 375							✓	0.496	0.163	TO
	z.33	38 375							✓	0.66	0.176	TO
	l.33	38 375							✓	0.744	0.189	TO
	z.66	38 375							✓	0.762	0.193	TO
	l.66	38 375							✓	0.606	0.176	TO
	z.100	38 375							✓	0.729	0.193	TO
	l.100	38 375							✓	0.732	0.207	TO
11	c	240								0.074	0.183	TO
	z	240								0.071	0.192	TO
	l	232								0.063	0.180	TO
	z.33	240								0.07	0.187	TO
	l.33	232								0.063	0.183	TO
	z.66	240								0.07	0.202	TO
	l.66	232								0.059	0.198	TO
	z.100	240								0.071	0.192	TO
	l.100	232								0.064	0.185	TO
12	c	83 644	10 242		103	97		171	✓	0.613	0.374	TO
	z	83 644	10 242		103	97		171	✓	0.634	0.353	TO
	l	83 644	10 242		103	97		171	✓	0.606	0.383	TO
	z.33	83 644	10 242		103	97		171	✓	0.599	0.363	TO
	l.33	83 644	10 242		103	97		171	✓	0.623	0.355	TO
	z.66	83 644	10 242		103	97		171	✓	0.602	0.365	TO
	l.66	83 644	10 242		103	97		171	✓	0.681	0.343	TO
	z.100	83 644	10 242		103	97		171	✓	0.594	0.355	TO
	l.100	83 644	10 242		103	97		171	✓	0.581	0.384	TO
13	c	1886		9	76	65		1700		0.252	TO	TO
	z	1886		9	76	65		1700		0.243	TO	TO
	l	1550	12	9	76	65	54	1700		0.222	TO	TO
	z.33	1886		9	76	65		1700		0.256	TO	TO
	l.33	1550	12	9	76	65	54	1700		0.233	TO	TO
	z.66	1886		9	76	65		1700		0.241	TO	TO
	l.66	1550	12	9	76	65	54	1700		0.234	TO	TO
	z.100	1886		9	76	65		1700		0.248	TO	TO
	l.100	1550	12	9	76	65	54	1700		0.243	TO	TO
14	c		1173						✓	0.013	0.025	TO
	z		1173						✓	0.013	0.025	TO
	l		1173						✓	0.012	0.024	TO
	z.33		1173						✓	0.014	0.03	TO
	l.33		1173						✓	0.016	0.025	TO
	z.66		1173						✓	0.013	0.026	TO
	l.66		1173						✓	0.012	0.024	TO
	z.100		1173						✓	0.014	0.025	TO
	l.100		1173						✓	0.012	0.044	TO
15	c	726			131	6		1247		0.163	TO	TO
	z	726			131	6		1247		0.147	TO	TO
	l	692			131	6		1247		0.12	TO	TO
	z.33	726			131	6		1247		0.146	TO	TO
	l.33	692			131	6		1247		0.133	TO	TO
	z.66	726			131	6		1247		0.151	TO	TO
	l.66	692			131	6		1247		0.134	TO	TO
	z.100	726			131	6		1247		0.144	TO	TO
	l.100	692			131	6		1247		0.146	TO	TO

Table C.12
Results of the absorption algorithm: ID16–ID21.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
16	c	658 440						2		228.547	1.401	TO
	z	658 440						2		196.418	1.643	TO
	l	658 482						2		104.403	1.552	TO
	z.33	658 440						2		104.927	1.636	TO
	l.33	658 482						2		85.294	1.626	TO
	z.66	658 440						2		113.289	1.589	TO
	l.66	658 482						2		113.912	1.358	TO
	z.100	658 440						2		140.782	1.62	TO
	l.100	658 482						2		128.486	1.675	TO
17	c	80	19		36	6			✓	0.004	0.016	TO
	z	80	19		36	6			✓	0.004	0.022	TO
	l	80	19		36	6			✓	0.004	0.016	TO
	z.33	80	19		36	6			✓	0.004	0.016	TO
	l.33	80	19		36	6			✓	0.004	0.017	TO
	z.66	80	19		36	6			✓	0.003	0.019	TO
	l.66	80	19		36	6			✓	0.005	0.016	TO
	z.100	80	19		36	6			✓	0.005	0.017	TO
	l.100	80	19		36	6			✓	0.005	0.018	TO
18	c	94	2		2	4		3		0.038	0.109	TO
	z	94	2		2	4		3		0.033	0.119	TO
	l	71	2		2	4		3		0.038	0.180	TO
	z.33	94	2		2	4		3		0.036	0.122	TO
	l.33	71	2		2	4		3		0.033	0.186	TO
	z.66	94	2		2	4		3		0.042	0.121	TO
	l.66	71	2		2	4		3		0.03	0.189	TO
	z.100	94	2		2	4		3		0.042	0.119	TO
	l.100	71	2		2	4		3		0.029	0.179	TO
19	c	696							✓	0.026	0.024	TO
	z	696							✓	0.025	0.020	TO
	l	696							✓	0.028	0.021	TO
	z.33	696							✓	0.023	0.021	TO
	l.33	696							✓	0.033	0.021	TO
	z.66	696							✓	0.023	0.021	TO
	l.66	696							✓	0.023	0.023	TO
	z.100	696							✓	0.024	0.02	TO
	l.100	696							✓	0.026	0.02	TO
20	c	18 477			12	71			✓	0.845	TO	TO
	z	18 477			12	71			✓	0.849	TO	TO
	l	18 477			12	71			✓	0.859	TO	TO
	z.33	18 477			12	71			✓	0.829	TO	TO
	l.33	18 477			12	71			✓	0.868	TO	TO
	z.66	18 477			12	71			✓	0.847	TO	TO
	l.66	18 477			12	71			✓	0.909	TO	TO
	z.100	18 477			12	71			✓	0.85	TO	TO
	l.100	18 477			12	71			✓	0.892	TO	TO
21	c	556			34	24	2			0.129	TO	TO
	z	556			34	24	2			0.127	TO	TO
	l	520	7		34	24	2			0.118	TO	TO
	z.33	556			34	24	2			0.138	TO	TO
	l.33	520	7		34	24	2			0.116	TO	TO
	z.66	556			34	24	2			0.126	TO	TO
	l.66	520	7		34	24	2			0.123	TO	TO
	z.100	556			34	24	2			0.135	TO	TO
	l.100	520	7		34	24	2			0.124	TO	TO

Table C.13
Results of the absorption algorithm: ID22–ID27.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
22	c	3084	12	235	169	155		1	✓	0.281	1.404	TO
	z	3084	12	235	169	155		1	✓	0.278	1.418	TO
	l	3084	12	235	169	155		1	✓	0.28	1.304	TO
	z.33	3084	12	235	169	155		1	✓	0.284	1.379	TO
	l.33	3084	12	235	169	155		1	✓	0.275	1.196	TO
	z.66	3084	12	235	169	155		1	✓	0.754	1.080	TO
	l.66	3084	12	235	169	155		1	✓	0.702	0.965	TO
	z.100	3084	12	235	169	155		1	✓	0.753	2.070	TO
	l.100	3084	12	235	169	155		1	✓	0.698	1.382	TO
23	c	2321						21		0.309	0.055	TO
	z	2321						21		0.294	0.056	TO
	l	2105						21		0.265	0.063	TO
	z.33	2321						21		0.304	0.057	TO
	l.33	2105						21		0.265	0.063	TO
	z.66	2321						21		0.295	0.055	TO
	l.66	2105						21		0.271	0.063	TO
	z.100	2321						21		0.295	0.055	TO
	l.100	2105						21		0.278	0.064	TO
24	c	167		1	14	12		20		0.052	0.457	TO
	z	167		1	14	12		20		0.053	0.463	TO
	l	152			14	12	1	20		0.048	1.508	TO
	z.33	167		1	14	12		20		0.052	0.434	TO
	l.33	152			14	12	1	20		0.049	1.549	TO
	z.66	167		1	14	12		20		0.052	0.449	TO
	l.66	152			14	12	1	20		0.05	1.548	TO
	z.100	167		1	14	12		20		0.054	0.442	TO
	l.100	152			14	12	1	20		0.05	1.483	TO
25	c	33 145			83			21 553		1.015	0.124	TO
	z	33 145			83			21 553		0.998	0.109	TO
	l	33 145			83		3	21 553		0.989	0.171	TO
	z.33	33 145			83			21 553		1.224	0.109	TO
	l.33	33 145			83		3	21 553		1.212	0.191	TO
	z.66	33 145			83			21 553		1.163	0.119	TO
	l.66	33 145			83		3	21 553		1.204	0.192	TO
	z.100	33 145			83			21 553		1.014	0.108	TO
	l.100	33 145			83		3	21 553		1.057	0.179	TO
26	c	79		79					✓	0.005	0.009	0.082
	z	79		79					✓	0.004	0.01	0.062
	l	79		79					✓	0.004	0.009	0.075
	z.33	79		79					✓	0.004	0.009	0.058
	l.33	79		79					✓	0.004	0.014	0.067
	z.66	79		79					✓	0.005	0.009	0.057
	l.66	79		79					✓	0.004	0.009	0.058
	z.100	79		79					✓	0.004	0.009	0.057
	l.100	79		79					✓	0.005	0.01	0.058
27	c	375			1			242		0.082	TO	TO
	z	375			1			242		0.079	TO	TO
	l	327			1			242		0.068	TO	TO
	z.33	379			1			242		0.081	TO	TO
	l.33	331			1			242		0.07	TO	TO
	z.66	379			1			242		0.081	TO	TO
	l.66	331					1	242		0.075	TO	TO
	z.100	379			1			242		0.081	TO	TO
	l.100	331					1	242		0.082	TO	TO

Table C.14
Results of the absorption algorithm: ID28–ID33.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
28	c	42 650						3	✓	0.372	0.057	TO
	z	42 650						3	✓	0.378	0.058	TO
	l	42 650						3	✓	0.4	0.057	TO
	z.33	42 650						3	✓	0.419	0.061	TO
	l.33	42 650						3	✓	0.506	0.056	TO
	z.66	42 650						3	✓	0.372	0.058	TO
	l.66	42 650						3	✓	0.386	0.061	TO
	z.100	42 650						3	✓	0.389	0.058	TO
	l.100	42 650						3	✓	0.392	0.064	TO
29	c	493		9						0.091	0.057	0.226
	z	493		9						0.084	0.067	0.23
	l	205	11	9			26			0.062	0.077	0.169
	z.33	493		9						0.079	0.069	0.241
	l.33	205	11	9			26			0.068	0.074	0.181
	z.66	493		9						0.083	0.067	0.239
	l.66	205	11	9			26			0.064	0.074	0.173
	z.100	493		9						0.08	0.066	0.236
	l.100	205	11	9			26			0.066	0.084	0.181
30	c	284			5	5		199		0.059	0.050	TO
	z	284			5	5		199		0.06	0.051	TO
	l	256			5	5	6	199		0.051	7.901	TO
	z.33	284			5	5		199		0.061	0.05	TO
	l.33	256			5	5	6	199		0.053	7.876	TO
	z.66	284			5	5		199		0.06	0.052	TO
	l.66	256			5	5	6	199		0.053	7.970	TO
	z.100	284			5	5		199		0.065	0.050	TO
	l.100	256			5	5	6	199		0.06	8.037	TO
31	c	333		79	54	55		141		0.118	TO	TO
	z	333		79	54	55		141		0.103	TO	TO
	l	324	1	79	54	55	20	141		0.151	TO	TO
	z.33	333		79	54	55		141		0.1	TO	TO
	l.33	324	1	79	54	55	20	141		0.148	TO	TO
	z.66	333		79	54	55		141		0.109	TO	TO
	l.66	324	1	79	54	55	20	141		0.15	TO	TO
	z.100	333		79	54	55		141		0.097	TO	TO
	l.100	324	1	79	54	55	20	141		0.133	TO	TO
32	c	24			5	5		55		0.008	0.027	0.094
	z	24			5	5		55		0.007	0.027	0.103
	l	15			5	5	2	55		0.008	0.03	0.111
	z.33	24			5	5		55		0.007	0.03	0.1
	l.33	15			5	5	2	55		0.006	0.031	0.104
	z.66	24			5	5		55		0.007	0.027	0.108
	l.66	15			5	5	2	55		0.006	0.031	0.101
	z.100	24			5	5		55		0.007	0.028	0.104
	l.100	15			5	5	2	55		0.007	0.031	0.104
33	c	877			4	6		96		0.183	TO	TO
	z	877			4	6		96		0.164	TO	TO
	l	755	1		4	6		96		0.144	TO	TO
	z.33	877			4	6		96		0.161	TO	TO
	l.33	755	1		4	6		96		0.15	TO	TO
	z.66	877			4	6		96		0.169	TO	TO
	l.66	755	1		4	6		96		0.156	TO	TO
	z.100	877			4	6		96		0.17	TO	TO
	l.100	755	1		4	6		96		0.17	TO	TO

Table C.15
Results of the absorption algorithm: ID34–ID39.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
34	c	1601		9	76	65		1578		0.237	TO	TO
	z	1601		9	76	65		1578		0.228	TO	TO
	l	1284	12	9	76	65	52	1578		0.207	TO	TO
	z.33	1601		9	76	65		1578		0.22	TO	TO
	l.33	1284	12	9	76	65	52	1578		0.21	TO	TO
	z.66	1601		9	76	65		1578		0.231	TO	TO
	l.66	1284	12	9	76	65	52	1578		0.215	TO	TO
	z.100	1601		9	76	65		1578		0.223	TO	TO
	l.100	1284	12	9	76	65	52	1578		0.218	TO	TO
35	c	750			167	17		1247		0.158	TO	TO
	z	750			167	17		1247		0.146	TO	TO
	l	716			167	17		1247		0.124	TO	TO
	z.33	750			167	17		1247		0.15	TO	TO
	l.33	716			167	17		1247		0.125	TO	TO
	z.66	750			167	17		1247		0.142	TO	TO
	l.66	716			167	17		1247		0.144	TO	TO
	z.100	750			167	17		1247		0.15	TO	TO
	l.100	716			167	17		1247		0.146	TO	TO
36	c	263		4	97	62		19		0.076	TO	TO
	z	263		4	97	62		19		0.079	TO	TO
	l	261		4	97	62	3	19		0.065	TO	TO
	z.33	263		4	97	62		19		0.076	TO	TO
	l.33	261		4	97	62	3	19		0.069	TO	TO
	z.66	263		4	97	62		19		0.064	TO	TO
	l.66	261		4	97	62	3	19		0.066	TO	TO
	z.100	263		4	97	62		19		0.067	TO	TO
	l.100	261		4	97	62	3	19		0.062	TO	TO
37	c	13 730							✓	0.56	0.046	TO
	z	13 730							✓	0.705	0.048	TO
	l	13 730							✓	0.711	0.051	TO
	z.33	13 730							✓	0.574	0.049	TO
	l.33	13 730							✓	0.614	0.047	TO
	z.66	13 730							✓	0.54	0.045	TO
	l.66	13 730							✓	0.654	0.049	TO
	z.100	13 730							✓	0.559	0.048	TO
	l.100	13 730							✓	0.701	0.049	TO
38	c	36 069							✓	0.435	0.059	TO
	z	36 069							✓	0.466	0.058	TO
	l	36 069							✓	0.388	0.057	TO
	z.33	36 069							✓	0.308	0.057	TO
	l.33	36 069							✓	0.383	0.057	TO
	z.33	36 069							✓	0.307	0.06	TO
	l.66	36 069							✓	0.321	0.061	TO
	z.100	36 069							✓	0.334	0.06	TO
	l.100	36 069							✓	0.35	0.058	TO
39	c	305			39	29		2		0.075	TO	TO
	z	305			39	29		2		0.071	TO	TO
	l	264			36	29	11	2		0.062	TO	TO
	z.33	305			39	29		2		0.069	TO	TO
	l.33	264			36	29	11	2		0.062	TO	TO
	z.66	305			39	29		2		0.072	TO	TO
	l.66	264			36	29	11	2		0.074	TO	TO
	z.100	305			39	29		2		0.073	TO	TO
	l.100	264			36	29	11	2		0.068	TO	TO

Table C.16
Results of the absorption algorithm: ID40–ID45.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
40	c	22			18	14		2		0.008	0.051	1.067
	z	22			18	14		2		0.009	0.054	1.141
	l	22			18	14	1	2		0.009	0.282	1.062
	z.33	22			18	14		2		0.008	0.05	1.084
	l.33	22			18	14	1	2		0.008	0.286	1.095
	z.66	22			18	14		2		0.008	0.052	1.136
	l.66	22			18	14	1	2		0.008	0.293	1.1
	z.100	22			18	14		2		0.008	0.052	1.065
	l.100	22			18	14	1	2		0.009	0.291	1.053
41	c	104	2	2	140	81		24		0.034	0.026	TO
	z	104	2	2	140	81		24		0.035	0.028	TO
	l	98	3	2	140	81	5	24		0.035	0.347	TO
	z.33	104	2	2	140	81		24		0.033	0.028	TO
	l.33	98	3	2	140	81	5	24		0.038	0.352	TO
	z.66	104	2	2	140	81		24		0.033	0.028	TO
	l.66	98	3	2	140	81	5	24		0.039	0.352	TO
	z.100	104	2	2	140	81		24		0.035	0.028	TO
	l.100	98	3	2	140	81	5	24		0.037	0.344	TO
42	c	132	2	3	56	40		74		0.037	1.27	TO
	z	132	2	3	56	40		74		0.035	1.29	TO
	l	132	2	3	56	40		74		0.036	1.179	TO
	z.33	132	2	3	56	40		74		0.038	1.315	TO
	l.33	136	2	1	56	40		74		0.035	1.210	TO
	z.66	132	2	3	56	40		74		0.035	1.286	TO
	l.66	138	2		56	40		74		0.038	1.229	TO
	z.100	132	2	3	56	40		74		0.036	1.275	TO
	l.100	138	2		56	40		74		0.039	1.207	TO
43	c	212		4	85	52		19		0.061	TO	TO
	z	212		4	85	52		19		0.058	TO	TO
	l	210		4	85	52	3	19		0.048	TO	TO
	z.33	212		4	85	52		19		0.062	TO	TO
	l.33	210		4	85	52	3	19		0.05	TO	TO
	z.66	212		4	85	52		19		0.058	TO	TO
	l.66	210		4	85	52	3	19		0.052	TO	TO
	z.100	212		4	85	52		19		0.055	TO	TO
	l.100	210		4	85	52	3	19		0.052	TO	TO
44	c	296			6	7		398		0.07	74.584	TO
	z	296			6	7		398		0.064	215.947	TO
	l	284			6	7	1	398		0.055	15.478	TO
	z.33	296			6	7		398		0.065	227.776	TO
	l.33	284			6	7	1	398		0.058	15.607	TO
	z.66	296			6	7		398		0.069	225.109	TO
	l.66	284			6	7	1	398		0.06	15.595	TO
	z.100	296			6	7		398		0.066	225.164	TO
	l.100	284			6	7	1	398		0.064	16.188	TO
45	c	227			115	104		29		0.059	TO	TO
	z	227			115	104		29		0.06	TO	TO
	l	223	1		115	104	5	29		0.055	TO	TO
	z.33	227			115	104		29		0.056	TO	TO
	l.33	223	1		115	104	5	29		0.051	TO	TO
	z.66	227			115	104		29		0.054	TO	TO
	l.66	223	1		115	104	5	29		0.052	TO	TO
	z.100	227			115	104		29		0.054	TO	TO
	l.100	223	1		115	104	5	29		0.055	TO	TO

Table C.17
Results of the absorption algorithm: ID46–ID51.

ID	Logic	$\mathcal{T}_{inc}$	$\mathcal{T}_{def}$	$\mathcal{T}_{syn}$	$\mathcal{T}_{dom}$	$\mathcal{T}_{rg}$	$\mathcal{T}_{g}$	$\mathcal{T}_{disj}$	LU	t_{abs}	t_{subs}	t_{noAbs}
46	c	309		5	104	60		21		0.081	TO	TO
	z	309		5	104	60		21		0.074	TO	TO
	l	303		5	104	60	3	21		0.067	TO	TO
	z.33	309		5	104	60		21		0.074	TO	TO
	l.33	303		5	104	60	3	21		0.07	TO	TO
	z.66	309		5	104	60		21		0.076	TO	TO
	l.66	303		5	104	60	3	21		0.069	TO	TO
	z.100	309		5	104	60		21		0.08	TO	TO
	l.100	303		5	104	60	3	21		0.07	TO	TO
47	c	164			5	5		56		0.06	TO	TO
	z	164			5	5		56		0.053	TO	TO
	l	140			5	5	3	56		0.045	TO	TO
	z.33	164			5	5		56		0.049	TO	TO
	l.33	140			5	5	3	56		0.045	TO	TO
	z.66	164			5	5		56		0.052	TO	TO
	l.66	140			5	5	3	56		0.047	TO	TO
	z.100	164			5	5		56		0.051	TO	TO
	l.100	140			5	5	3	56		0.047	TO	TO
48	c	96			15	8		54		0.036	0.0375	TO
	z	96			15	8		54		0.036	0.041	TO
	l	94	1		13	8	13	54		0.049	TO	TO
	z.33	96			15	8		54		0.036	0.04	TO
	l.33	94	1		13	8	13	54		0.049	TO	TO
	z.66	96			15	8		54		0.035	0.0405	TO
	l.66	94	1		13	8	13	54		0.048	TO	TO
	z.100	96			15	8		54		0.036	0.04	TO
	l.100	94	1		13	8	13	54		0.052	TO	TO
49	c	64			35	8	1	6		0.029	TO	TO
	z	64			35	8	1	6		0.027	TO	TO
	l	64			35	8	1	6		0.027	TO	TO
	z.33	64			35	8	1	6		0.026	TO	TO
	l.33	64			35	8	1	6		0.027	TO	TO
	z.66	64			35	8	1	6		0.026	TO	TO
	l.66	64			35	8	1	6		0.029	TO	TO
	z.100	64			35	8	1	6		0.029	TO	TO
	l.100	64			35	8	1	6		0.029	TO	TO
50	c	111			50	50		112		0.036	0.028	0.718
	z	111			50	50		112		0.037	0.032	0.761
	l	109	1		50	50	13	112		0.034	0.144	0.730
	z.33	111			50	50		112		0.036	0.031	0.728
	l.33	109	1		50	50	13	112		0.036	0.145	0.741
	z.66	111			50	50		112		0.035	0.03	0.716
	l.66	109	1		50	50	13	112		0.036	0.146	0.753
	z.100	111			50	50		112		0.036	0.03	0.723
	l.100	109	1		50	50	13	112		0.037	0.15	0.773
51	c	35 988						681		1.311	0.045	TO
	z	35 988						681		1.284	0.045	TO
	l	30 765					3	681		2.67	TO	TO
	z.33	35 988						681		1.42	0.043	TO
	l.33	30 765					3	681		4.367	TO	TO
	z.66	35 988						681		1.36	0.048	TO
	l.66	30 765					3	681		4.208	TO	TO
	z.100	35 988						681		1.375	0.047	TO
	l.100	30 765					3	681		4.258	TO	TO

References

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P.F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, 2003.
[2] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Sci. Am. 284 (5) (2001) 34–43.
[3] B. Cuenca-Grau, I. Horrocks, B. Motik, B. Parsia, P.F. Patel-Schneider, U. Sattler, OWL 2: the next step for OWL, J. Web Semant. 6 (4) (2008) 309–322.
[4] I. Horrocks, O. Kutz, U. Sattler, The even more irresistible $\mathcal{SROIQ}$, in: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning, KR 2006, AAAI Press, 2006, pp. 57–67.
[5] J. Yen, Generalizing term subsumption languages to fuzzy logic, in: Proceedings of the 12th International Joint Conference on Artificial Intelligence, IJCAI 1991, 1991, pp. 472–477.
[6] T. Lukasiewicz, U. Straccia, Managing uncertainty and vagueness in description logics for the semantic web, J. Web Semant. 6 (4) (2008) 291–308.
[7] U. Straccia, Foundations of Fuzzy Logic and Semantic Web Languages, CRC Studies in Informatics Series, Chapman & Hall, 2013.
[8] V. Haarslev, H.-I. Pai, N. Shiri, Optimizing tableau reasoning in $\mathcal{ALC}$ extended with uncertainty, in: Proceedings of the 2007 International Workshop on Description Logics, DL 2007, 2007.
[9] N. Simou, T.P. Mailis, G. Stoilos, G.B. Stamou, Optimization techniques for fuzzy description logics, in: Proceedings of the 23rd International Workshop on Description Logics, in: CEUR Electronic Workshop Proceedings, vol. 573, 2010.
[10] I. Horrocks, S. Tobies, Reasoning with axioms: theory and practice, in: Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning, KR 2000, Morgan Kaufman, 2000, pp. 285–296.
[11] A.K. Hudek, G.E. Weddell, Binary absorption in tableaux-based reasoning for description logics, in: Proceedings of the 2006 International Workshop on Description Logics, DL 2006, in: CEUR Workshop Proceedings, vol. 189, CEUR-WS.org, 2006.
[12] D. Tsarkov, I. Horrocks, Efficient reasoning with range and domain constraints, in: Proceedings of the 2004 Description Logic Workshop, DL 2004, 2004, pp. 41–50.
[13] E. Sirin, B.C. Grau, B. Parsia, From wine to water: optimizing description logic reasoning for nominals, in: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning, KR 2006, AAAI Press, 2006, pp. 90–99.
[14] D. Tsarkov, I. Horrocks, P.F. Patel-Schneider, Optimizing terminological reasoning for expressive description logics, J. Autom. Reason. 39 (3) (2007) 277–316.
[15] J. Wu, V. Haarslev, Planning of axiom absorption, in: Proceedings of the 21st International Workshop on Description Logics, DL 2008, in: CEUR Workshop Proceedings, vol. 353, CEUR-WS.org, 2008.
[16] M. Zuo, V. Haarslev, High performance absorption algorithms for terminological reasoning, in: Proceedings of the 2006 International Workshop on Description Logics, DL 2006, in: CEUR Workshop Proceedings, vol. 189, CEUR-WS.org, 2006.
[17] F. Bobillo, F. Bou, U. Straccia, On the failure of the finite model property in some fuzzy description logics, Fuzzy Sets Syst. 172 (1) (2011) 1–12.
[18] M. Cerami, U. Straccia, On the (un)decidability of fuzzy description logics under Łukasiewicz t-norm, Inf. Sci. 227 (1) (2013) 1–21.
[19] F. Baader, S. Borgwardt, R. Peñaloza, On the decidability status of fuzzy $\mathcal{ALC}$ with general concept inclusions, J. Philos. Log. (2014), http://dx.doi.org/10.1007/s10992-014-9329-3, in press.
[20] F. Bobillo, U. Straccia, Fuzzy description logics with general t-norms and datatypes, Fuzzy Sets Syst. 160 (23) (2009) 3382–3402.
[21] S. Borgwardt, F. Distel, R. Peñaloza, The limits of decidability in fuzzy description logics with general concept inclusions, Artif. Intell., http://dx.doi.org/10.1016/j.artint.2014.09.001, in press.
[22] F. Bobillo, M. Delgado, J. Gómez-Romero, U. Straccia, Joining Gödel and Zadeh fuzzy logics in fuzzy description logics, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 20 (4) (2012) 475–508.
[23] M. Cerami, À. García-Cerdañ, F. Esteva, On finitely-valued fuzzy description logics, Int. J. Approx. Reason. (2014), http://dx.doi.org/10.1016/j.ijar.2013.09.021, in press.
[24] F. Bobillo, U. Straccia, fuzzyDL: an expressive fuzzy description logic reasoner, in: 2008 International Conference on Fuzzy Systems, FUZZ-IEEE 2008, IEEE Computer Society, 2008, pp. 923–930.
[25] P. Hájek, Metamathematics of Fuzzy Logic, Kluwer, 1998.
[26] G.J. Klir, B. Yuan, Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice-Hall, Inc., 1995.
[27] P.S. Mostert, A.L. Shields, On the structure of semigroups on a compact manifold with boundary, Ann. Math. 65 (1) (1957) 117–143.
[28] L.A. Zadeh, Fuzzy sets, Inf. Control 8 (3) (1965) 338–353.
[29] OWL 2 Web Ontology Language Document Overview, http://www.w3.org/TR/2009/REC-owl2-overview-20091027, 2009.
[30] U. Straccia, A fuzzy description logic for the semantic web, in: E. Sanchez (Ed.), Fuzzy Logic and the Semantic Web, Capturing Intelligence, Ch. 4, Elsevier, 2006, pp. 73–90.
[31] F. Bobillo, M. Delgado, J. Gómez-Romero, Crisp representations and reasoning for fuzzy ontologies, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 17 (4) (2009) 501–530.
[32] U. Straccia, Description logics with fuzzy concrete domains, in: Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, UAI 2005, AUAI Press, 2005, pp. 559–567.
[33] F. Bobillo, U. Straccia, Fuzzy ontology representation using OWL 2, Int. J. Approx. Reason. 52 (7) (2011) 1073–1094.
[34] F.A. Lisi, U. Straccia, A logic-based computational method for the automated induction of fuzzy ontology axioms, Fundam. Inform. 124 (4) (2013) 503–519.
[35] B. Glimm, I. Horrocks, B. Motik, R. Shearer, G. Stoilos, A novel approach to ontology classification, J. Web Semant. 14 (2012) 84–101.
[36] V. Haarslev, R. Möller, On the scalability of description logic instance retrieval, J. Autom. Reason. 41 (2) (2008) 99–142.

[37] I. Horrocks, L. Li, D. Turi, S. Bechhofer, The instance store: DL reasoning with large numbers of individuals, in: Proceedings of the 2004 Description Logic Workshop, DL 2004, 2004, pp. 31–40.

[38] J. Wu, A.K. Hudek, D. Toman, G.E. Weddell, Absorption for ABoxes, in: Proceedings of the 2012 International Workshop on Description Logics, DL 2012, in: CEUR Workshop Proceedings, vol. 846, CEUR-WS.org, 2012.

[39] J. Wu, A.K. Hudek, D. Toman, 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, pp. 648–652.

[40] B. Motik, R. Shearer, I. Horrocks, Optimized reasoning in description logics using hypertableaux, in: Proceedings of the 21st International Conference on Automated Deduction, CADE-21, Springer-Verlag, 2007, pp. 67–83.