Expert Systems with Applications 39 (2012) 258–272

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

DeLorean: A reasoner for fuzzy OWL 2 ☆

Fernando Bobillo[a,*], Miguel Delgado[b], Juan Gómez-Romero[c]

[a] Department of Computer Science and Systems Engineering, University of Zaragoza, Spain
[b] Department of Computer Science and Artificial Intelligence, University of Granada, Spain
[c] Applied Artificial Intelligence Group, University Carlos III, Madrid, Spain

ARTICLE INFO

Keywords:
Fuzzy rough description logics
Fuzzy rough ontologies
Fuzzy description logic reasoner
Fuzzy rough ontology reasoner

ABSTRACT

Today, there is a growing interest in the development of knowledge representations able to deal with uncertainty, which is a very common requirement in real world applications. Despite the undisputed success of ontologies, classical ontologies are not suitable to deal with uncertainty and, consequently, several extensions with fuzzy logic and rough logic, among other formalisms, have been proposed.

In this article we describe DeLorean 2, the first ontology reasoner that supports fuzzy extensions of the standard languages OWL and OWL 2. In a strict sense, DeLorean is not a reasoner but a translator from fuzzy rough ontology languages ($GZ\ \mathcal{SROIQ}(\mathbf{D})$) into classical ontology languages ($\mathcal{SROIQ}(\mathbf{D})$). This allows using classical (widely available) Description Logic inference engines to reason with the representation resulting from the transformation. We describe the main features of the application: evolution, functionality, architecture, graphical interface, input language, and implementation details.

1. Introduction

The use of ontologies as formalisms for knowledge representation in many different application domains has grown significantly lately. Ontologies have been successfully used as part of expert and multiagent systems, as well as a core element in the Semantic Web, which proposes to extend the current web to give information a well-defined meaning (Berners-Lee, Hendler, & Lassila, 2001). An ontology is defined as an explicit and formal specification of a shared conceptualization (Gruber, 1993), which means that ontologies represent the concepts and the relationships in a domain promoting interrelation with other models and automatic processing. Ontologies allow adding semantics to data, making knowledge maintenance, information integration, and reuse of components easier. The language OWL 2 (Cuenca-Grau et al., 2008) has very recently become a W3C Recommendation for ontology representation. OWL 2 is an extension of the previous standard language OWL (W3C, 2004).

Description Logics (DLs for short) (Baader, Calvanese, McGuinness, Nardi, & Patel-Schneider, 2003) are a family of logics for representing structured knowledge. Each logic is denoted by using a string of capital letters which identify the constructors of the logic and therefore its complexity. DLs have proved to be very useful as ontology languages. For instance, the languages OWL 2 and OWL are almost equivalent to $\mathcal{SROIQ}(\mathbf{D})$ (Horrocks, Kutz, & Sattler, 2006) and $\mathcal{SHOIN}(\mathbf{D})$ (Horrocks & Patel-Schneider, 2004), respectively.

Nevertheless, it has been widely pointed out that classical ontologies are not appropriate to deal with imprecise and vague knowledge, which is inherent to several real world domains (Sanchez, 2006). Since fuzzy set theory and fuzzy logic are suitable formalisms to handle these types of knowledge, several fuzzy extensions of DLs can be found in the literature (Lukasiewicz & Straccia, 2008), yielding fuzzy ontologies. Fuzzy ontologies have proved to be useful in several applications, such as information retrieval (Calegari & Sanchez, 2008), image interpretation (Hudelot, Atif, & Bloch, 2008), and the Semantic Web and the Internet (Costa, Laskey, & Lukasiewicz, 2008; Sanchez, 2006), among many others.

Rough sets theory offers a qualitative approach to model vagueness (Pawlak, 1982). Fuzzy logic and rough logic are complementary formalisms and hence it is natural to combine them by means of fuzzy rough sets. Fuzzy rough extensions of ontologies have also been presented (Bobillo & Straccia, 2009).

In this paper we describe DeLorean (DEscription LOgic REasoner with vAgueNess),[1] a reasoner that supports fuzzy rough extensions of the languages OWL and OWL 2, also implementing some interesting optimization techniques. As stated by some authors, we strongly believe that "*the broad acceptance of the forthcoming*

☆ This paper is a revised and extended version of "DeLorean: A Reasoner for Fuzzy OWL 1.1", in the Proceedings of the 4th International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2008).

* Corresponding author.
E-mail addresses: fbobillo@unizar.es (F. Bobillo), mdelgado@ugr.es (M. Delgado), jgromero@inf.uc3m.es (J. Gómez-Romero).

[1] http://www.webdiis.unizar.es/fbobillo/delorean.

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.07.016

OWL 2 ontology language will largely depend on the availability of editors, reasoners and numerous other tools that support the use of OWL 2 from a high-level/application perspective" (Horridge, Bechhofer, & Noppens, 2007). This is a significant contribution in this direction, since DeLorean is the first tool that supports fuzzy OWL 2.

In a strict sense, DeLorean is not a reasoner but a *translator* from a fuzzy rough ontology language into a classical ontology language (the standard language OWL or OWL 2, depending on the expressivity of the original ontology). Then, a classical DL reasoner is employed to reason with the resulting ontology. Nevertheless, due to this ability of combining the reduction procedure with the classical DL reasoning, we will refer to DeLorean as a reasoner.

This paper is organized as follows. Section 2 recalls some preliminaries on fuzzy rough set theory and fuzzy logic. Section 3 describes the fuzzy rough DL $GZ\ \mathcal{SROIQ}(\mathbf{D})$, which is equivalent to the language supported by DeLorean. Then, Section 4 contains the main part, a detailed description of the architecture and the development of the application. Next, Section 5 reviews some related work. Finally, Section 6 sets out some conclusions and ideas for future work.

2. Fuzzy rough sets and fuzzy logic

Fuzzy set theory and fuzzy logic were proposed by Zadeh (1965) to manage imprecise and vague knowledge. While in classical set theory elements either belong to a set or not, in fuzzy set theory elements can belong to a set to some degree. More formally, 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 any $x \in X$ to a value in the interval of real numbers between 0 and 1. As in the classical case, 0 means no-membership and 1 full membership, but now a value between 0 and 1 represents the extent to which x can be considered as an element of X.

All classical set operations are extended to fuzzy sets. The intersection, union, complement and implication set operations are performed by some functions (a t-norm, a t-conorm, a negation, and an implication, respectively). Table 1 shows the most important fuzzy logics, which are set according to the definition of these operations: Zadeh, Łukasiewicz, Gödel, and Product. The implication of Zadeh fuzzy logic is called Kleene-Dienes implication.

It is also common to restrict to *finite chains of degrees of truth*, instead of the real interval $[0,1]$. For our purposes all finite chains with the same number of elements are equivalent, so we will deal with the simplest finite chain of $p+1$ elements: $\mathcal{N} = \{0 = \gamma_0 < \gamma_1 < \cdots < \gamma_p = 1\}$, where $p \geqslant 1$ (Mayor & Torrens, 1993). Such an $\mathcal{N}$ can be understood as a set of linguistic terms or labels. For example, {`false`, `closeToFalse`, `neutral`, `closeToTrue`, `true`}. From a practical point of view, it is very often sufficient to use a small p.

For every $\alpha \in \mathcal{N}$, the *α-cut* of a fuzzy set A is defined as the (crisp) set such that its elements belong to A with degree at least α, i.e. $\{x \mid \mu_A(x) \geqslant \alpha\}$. Similarly, the *strict α-cut* is defined as $\{x \mid \mu_A(x) > \alpha\}$.

T-norms, t-conorms, negations and implications can be restricted to finite chains (Mas, Monserrat, & Torrens, 2004; Mayor & Torrens, 1993). Table 2 shows the most important fuzzy logics under finite chains: Zadeh, Łukasiewicz, and Gödel. Clearly, a product-based fuzzy logic cannot be defined over a finite chain since, in general, $\gamma_i \otimes \gamma_j \notin \mathcal{N}$.

Fuzzy logic uses fuzzy set theory to perform approximate reasoning. For the scope of this paper, *fuzzy statements* have the form $\phi \geqslant \alpha$ or $\phi \leqslant \beta$, where $\alpha \in \mathcal{N} \setminus \{\gamma_0\}, \beta \in \mathcal{N} \setminus \{\gamma_p\}$, and ϕ is a statement. Fuzzy statement represents that the degree of truth of ϕ is *at least* α (resp. *at most* β). For example, ripeTomato $\geqslant$ `closeToTrue` says that it is almost true that a tomato is ripe.

Relations can also be extended to the fuzzy case. A (binary) *fuzzy relation R* over two countable sets X and Y is a function $R : X \times Y \to \mathcal{N}$. Several properties of the relations (such as reflexive, irreflexive, symmetric, asymmetric, transitive, or disjointness can) and operations (inverse, composition) can trivially be extended to the finite case.

The notion of rough set was introduced by Pawlak (1982). The key idea in rough set theory is the approximation of a vague concept when there is only incomplete information about the concept. More precisely, there are only some examples of elements that belong to the concept, and an indiscernibility equivalence (reflexive, symmetric, and transitive) or similarity (reflexive and symmetric) relation between elements of the domain. In this case, a vague concept is approximated by means of a pair of concepts: a sub-concept or lower approximation, and a super-concept or upper approximation. The lower approximation describes the sets of elements that definitely belong to the vague set. The upper approximation describes the sets of elements that possibly belong to the vague set. A *rough set* is then defined as a pair of concepts: a lower approximation and an upper approximation of a vague concept.

A very natural extension to rough sets is to consider a fuzzy similarity relation instead of an indiscernibility relation, which gives raise to *fuzzy rough sets* (Cock, Cornelis, & Kerre, 2007; Radzikowska & Kerre, 2002).

While in rough sets one element of the domain can only belong to one equivalence class, in fuzzy rough sets one element can belong to several fuzzy similarity classes (with different degrees of truth). Thus, the notion of tight and loose approximation naturally appear (Cock et al., 2007): a tight approximation considers all fuzzy similarity classes, whereas a lower approximation considers the best one among the similarity classes.

Given a fuzzy similarity relation s, a t-norm $\otimes$ and an implication function $\Rightarrow$, the lower approximation ($s{\downarrow}A$), the upper approximation ($s{\uparrow}A$), the tight lower approximation ($s{\downarrow\downarrow}A$), the loose lower approximation ($s{\uparrow\downarrow}A$), the tight upper approximation ($s{\downarrow\uparrow}A$), and the loose upper approximation ($s{\uparrow\uparrow}A$) of a fuzzy subset A of X are defined by the following membership functions:

$$(s \downarrow A)(x) = \inf_{y \in X}\{s(x,y) \Rightarrow A(y)\}$$

$$(s \uparrow A)(x) = \sup_{y \in X}\{s(x,y) \otimes A(y)\}$$

$$(s{\downarrow\downarrow}A)(x) = \inf_{z \in X}\left\{s(x,z) \Rightarrow \inf_{y \in X}\{s(z,y) \Rightarrow A(y)\}\right\}$$

$$(s{\uparrow\downarrow}A)(x) = \sup_{z \in X}\left\{s(x,z) \otimes \inf_{y \in X}\{s(z,y) \Rightarrow A(y)\}\right\}$$

$$(s{\downarrow\uparrow}A)(x) = \inf_{z \in X}\left\{s(x,z) \Rightarrow \sup_{y \in X}\{s(z,y) \otimes A(y)\}\right\}$$

$$(s{\uparrow\uparrow}A)(x) = \sup_{z \in X}\left\{s(x,z) \otimes \sup_{y \in X}\{s(z,y) \otimes A(y)\}\right\}$$

Table 1
Popular fuzzy logics over $[0,1]$.

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

Table 2
Popular fuzzy logics over a finite chain.

Family	$\gamma_i \otimes \gamma_j$	$\gamma_i \oplus \gamma_j$	$\ominus\gamma_i$	$\gamma_i \Rightarrow \gamma_j$
Zadeh	$\min\{\gamma_i, \gamma_j\}$	$\max\{\gamma_i, \gamma_j\}$	γ_{p-i}	$\max\{\gamma_{p-i}, \gamma_j\}$
Gödel	$\min\{\gamma_i, \gamma_j\}$	$\max\{\gamma_i, \gamma_j\}$	$\begin{cases} \gamma_p, & \gamma_i = 0 \\ \gamma_0, & \gamma_i > 0 \end{cases}$	$\begin{cases} \gamma_p & \gamma_i \leqslant \gamma_j \\ \gamma_j, & \gamma_i > \gamma_j \end{cases}$
Łukasiewicz	$\gamma_{\max\{i+j-p,0\}}$	$\gamma_{\min\{i+j,p\}}$	γ_{p-i}	$\gamma_{\min\{p-i+j,p\}}$

3. Fuzzy rough $\mathcal{SROIQ}(\mathbf{D})$

In this section we define a fuzzy rough extension of $\mathcal{SROIQ}(\mathbf{D})$, namely *GZ* $\mathcal{SROIQ}(\mathbf{D})$. *GZ* $\mathcal{SROIQ}(\mathbf{D})$ correspond to a fuzzy extension of the DL $\mathcal{SROIQ}(\mathbf{D})$, equivalent to OWL 2, under a semantics given by Zadeh (denoted by the *Z*) and Gödel (denoted by the *G*) fuzzy logics, restricted to a finite chain of degrees of truth. In fact, concepts are fuzzy rough sets of individuals, roles are fuzzy binary relations. Axioms are extended to the fuzzy case, and some of them hold to a degree.

This logic was firstly presented in Bobillo, Delgado, and Gómez-Romero (2009) (*Z* $\mathcal{SROIQ}(\mathbf{D})$, and Bobillo, Delgado, Gómez-Romero, & Straccia (2009) (*G* $\mathcal{SROIQ}(\mathbf{D})$, and extended with fuzzy rough concepts in Bobillo & Straccia (2009); Bobillo & Straccia (submitted). In Bobillo, Delgado, Gómez-Romero, & Straccia (Submitted), an unified and comprehensive view on *GZ* $\mathcal{SROIQ}(\mathbf{D})$ was provided.

In the remainder of this section, we will assume the reader to be familiar with non-fuzzy DLs (Baader et al., 2003).

3.1. Fuzzy concrete domains

A *fuzzy concrete domain* (Straccia, 2005) $\mathbf{D}$ is a pair $\langle \Delta_{\mathbf{D}}, \Phi_{\mathbf{D}} \rangle$, where $\Delta_{\mathbf{D}}$ is a concrete interpretation domain, and $\Phi_{\mathbf{D}}$ is a set of fuzzy concrete predicates $\mathbf{d}$ with an arity n and an interpretation $\mathbf{d}_{\mathbf{D}}$: $\Delta_{\mathbf{D}}{}^n \to [0,1]$, which is an n-ary fuzzy relation over $\Delta_{\mathbf{D}}$.

As fuzzy concrete predicates we allow a discretized version of the trapezoidal membership function (Fig. 1 (a)). The trapezoidal membership function can be used to represent other popular functions, such as the triangular, the left-shoulder, and the right-shoulder functions (Straccia, 2005).

3.2. Fuzzy modifiers

A *fuzzy modifier mod* is a function $f_{mod} : \mathcal{N} \to \mathcal{N}$ which applies to a fuzzy set to change its membership function. A typical example is the fuzzy modifier *very*. We will allow modifiers defined in terms of the triangular membership function (Fig. 1 (b)).

Table 3
Syntax of concepts, roles, and axioms in *GZ* $\mathcal{SROIQ}$.

Concept constructor	Syntax
Top concept	$\top$
Bottom concept	$\perp$
Atomic concept	A
Conjunction	$C \sqcap D$
Disjunction	$C \sqcup D$
Gödel negation	$\neg_G C$
Zadeh negation	$\neg_Z C$
Gödel universal restriction	$\forall_G R.C$
Zadeh universal restriction	$\forall_Z R.C$
Existential restriction	$\exists R.C$
Concrete Gödel universal restriction	$\forall_G T.\mathbf{d}$
Concrete Zadeh universal restriction	$\forall_Z T.\mathbf{d}$
Concrete existential restriction	$\exists T.\mathbf{d}$
Fuzzy nominal	$\{\alpha/a\}$
At-least restriction	$(\geqslant m\ S.C)$
Gödel at-most restriction	$(\leqslant_G n\ S.C)$
Zadeh at-most restriction	$(\leqslant_Z n\ S.C)$
Concrete at-least restriction	$(\geqslant m\ T.\mathbf{d})$
Concrete Gödel at-most restriction	$(\leqslant_G n\ T.\mathbf{d})$
Concrete Zadeh at-most restriction	$(\leqslant_Z n\ T.\mathbf{d})$
Local reflexivity	$\exists S.\mathtt{Self}$
Cut concept	$[C \geqslant \alpha]$
Modified concept	$mod(C)$
Tight lower approximation	$(s_i{\downarrow}{\downarrow}C)$
Loose lower approximation	$(s_i{\uparrow}{\downarrow}C)$
Lower approximation	$(s_i{\downarrow}C)$
Tight upper approximation	$(s_i{\downarrow}{\uparrow}C)$
Loose upper approximation	$(s_i{\uparrow}{\uparrow}C)$
Upper approximation	$(s_i{\uparrow}C)$
Role constructor	**Syntax**
Atomic abstract role	R_A
Concrete abstract role	T
Universal role	U
Inverse role	R^-
Axiom	**Syntax**
Concept assertion	$\langle a : C \bowtie \gamma \rangle$
Role assertion	$\langle (a,b) : R \bowtie \gamma \rangle$
Gödel negated role assertion	$\langle (a,b) : \neg_G R \bowtie \gamma \rangle$
Zadeh negated role assertion	$\langle (a,b) : \neg_Z R \bowtie \gamma \rangle$
Inequality assertion	$a \neq b$
Equality assertion	$a = b$
Gödel General Concept Inclusion (GCI)	$\langle C \sqsubseteq_G D \rhd \gamma \rangle$
Zadeh General Concept Inclusion (GCI)	$\langle C \sqsubseteq_Z D \rhd \gamma \rangle$
Gödel Role Inclusion Axiom (RIA)	$\langle R_1 \ldots R_m \sqsubseteq_G R \rhd \gamma \rangle$
Zadeh Role Inclusion Axiom (RIA)	$\langle R_1 \ldots R_m \sqsubseteq_Z R \rhd \gamma \rangle$
Gödel concrete RIA	$\langle T_1 \sqsubseteq_G T_2 \rhd \gamma \rangle$
Zadeh concrete RIA	$\langle T_1 \sqsubseteq_Z T_2 \rhd \gamma \rangle$
Transitive role axiom	$\mathtt{trans}(R)$
Disjoint role axiom	$\mathtt{dis}(S_1, S_2)$
Disjoint concrete role axiom	$\mathtt{dis}(T_1, T_2)$
Reflexive role axiom	$\mathtt{ref}(R)$
Irreflexive role axiom	$\mathtt{irr}(S)$
Symmetric role axiom	$\mathtt{sym}(R)$
Asymmetric role axiom	$\mathtt{asy}(S)$

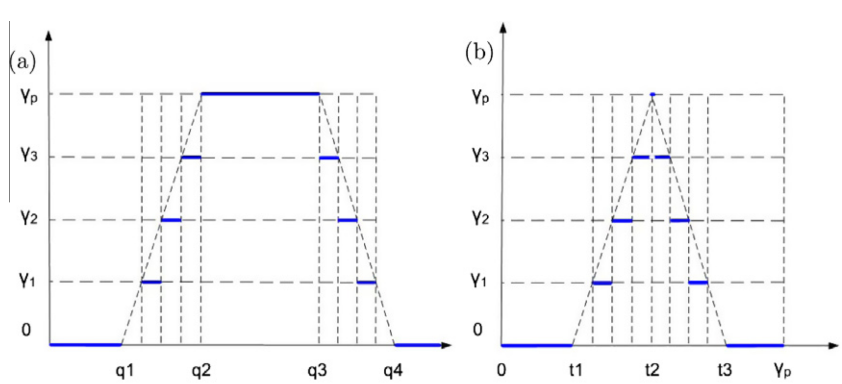

Fig. 1. (a) Trapezoidal membership function; (b) Triangular fuzzy modifier.

Table 4
Semantics of concepts, roles, and axioms in $GZ\ \mathcal{SROIQ}$.

Concept C	Semantics
$(\top)^{\mathcal{I}}(x)$	γ_p
$(\bot)^{\mathcal{I}}(x)$	γ_0
$(A)^{\mathcal{I}}(x)$	$A^{\mathcal{I}}(x)$
$(C \sqcap D)^{\mathcal{I}}(x)$	$C^{\mathcal{I}}(x) \otimes D^{\mathcal{I}}(x)$
$(C \sqcup D)^{\mathcal{I}}(x)$	$C^{\mathcal{I}}(x) \oplus D^{\mathcal{I}}(x)$
$(\neg_G C)^{\mathcal{I}}(x)$	$\ominus_G C^{\mathcal{I}}(x)$
$(\neg_Z C)^{\mathcal{I}}(x)$	$\ominus_Z C^{\mathcal{I}}(x)$
$(\forall_G R.C)^{\mathcal{I}}(x)$	$\inf_{y \in \Delta^{\mathcal{I}}} \left\{ R^{\mathcal{I}}(x,y) \Rightarrow_G C^{\mathcal{I}}(y) \right\}$
$(\forall_Z R.C)^{\mathcal{I}}(x)$	$\inf_{y \in \Delta^{\mathcal{I}}} \left\{ R^{\mathcal{I}}(x,y) \Rightarrow_Z C^{\mathcal{I}}(y) \right\}$
$(\exists R.C)^{\mathcal{I}}(x)$	$\sup_{y \in \Delta^{\mathcal{I}}} \left\{ R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y) \right\}$
$(\forall_G T.\mathtt{d})^{\mathcal{I}}(x)$	$\inf_{v \in \Delta_{\mathbf{D}}} \{ T^{\mathcal{I}}(x,v) \Rightarrow_G \mathtt{d}_{\mathbf{D}}(v) \}$
$(\forall_Z T.\mathtt{d})^{\mathcal{I}}(x)$	$\inf_{v \in \Delta_{\mathbf{D}}} \{ T^{\mathcal{I}}(x,v) \Rightarrow_Z \mathtt{d}_{\mathbf{D}}(v) \}$
$(\exists T.\mathtt{d})^{\mathcal{I}}(x)$	$\sup_{v \in \Delta_{\mathbf{D}}} \{ T^{\mathcal{I}}(x,v) \otimes \mathtt{d}_{\mathbf{D}}(v) \}$
$(\{\alpha/a\})^{\mathcal{I}}(x)$	α if $x = a^{\mathcal{I}}$, γ_0 otherwise
$(\geqslant m\ S.C)^{\mathcal{I}}(x)$	$\sup_{y_1,\ldots,y_m \in \Delta^{\mathcal{I}}} \left[\left(\min_{i=1}^{m} \left\{ S^{\mathcal{I}}(x,y_i) \otimes C^{\mathcal{I}}(y_i) \right\} \right) \otimes \left(\otimes_{j<k} \{ y_j \neq y_k \} \right) \right]$
$(\leqslant_G n\ S.C)^{\mathcal{I}}(x)$	$\inf_{y_1,\ldots,y_{n+1} \in \Delta^{\mathcal{I}}} \left[\left(\min_{i=1}^{n+1} \left\{ S^{\mathcal{I}}(x,y_i) \otimes C^{\mathcal{I}}(y_i) \right\} \right) \Rightarrow_G \left(\oplus_{j<k} \{ y_j = y_k \} \right) \right]$
$((\leqslant_Z n\ S.C))^{\mathcal{I}}(x)$	$\inf_{y_1,\ldots,y_{n+1} \in \Delta^{\mathcal{I}}} \left[\left(\min_{i=1}^{n+1} \left\{ S^{\mathcal{I}}(x,y_i) \otimes C^{\mathcal{I}}(y_i) \right\} \right) \Rightarrow_Z \left(\oplus_{j<k} \{ y_j = y_k \} \right) \right]$
$(\geqslant m\ T.\mathtt{d})^{\mathcal{I}}(x)$	$\sup_{v_1,\ldots,v_m \in \Delta_{\mathbf{D}}} \left[\left(\min_{i=1}^{m} \{ T^{\mathcal{I}}(x,v_i) \otimes \mathtt{d}_{\mathbf{D}}(v_i) \} \right) \otimes \left(\otimes_{j<k} \{ y_j \neq y_k \} \right) \right]$
$(\leqslant_G n\ T.\mathtt{d})^{\mathcal{I}}(x)$	$\inf_{v_1,\ldots,v_{n+1} \in \Delta_{\mathbf{D}}} \left[\left(\min_{i=1}^{n+1} \{ T^{\mathcal{I}}(x,v_i) \otimes \mathtt{d}_{\mathbf{D}}(v_i) \} \right) \Rightarrow_G \left(\oplus_{j<k} \{ y_j = y_k \} \right) \right]$
$((\leqslant_Z n\ T.\mathtt{d}))^{\mathcal{I}}(x)$	$\inf_{v_1,\ldots,v_{n+1} \in \Delta_{\mathbf{D}}} \left[\left(\min_{i=1}^{n+1} \{ T^{\mathcal{I}}(x,v_i) \otimes \mathtt{d}_{\mathbf{D}}(v_i) \} \right) \Rightarrow_Z \left(\oplus_{j<k} \{ y_j = y_k \} \right) \right]$
$(\exists S.\mathtt{Self})^{\mathcal{I}}(x)$	$S^{\mathcal{I}}(x,x)$
$([C \geqslant \alpha])^{\mathcal{I}}(x)$	γ_p if $C^{\mathcal{I}}(x) \geqslant \alpha$, γ_0 otherwise
$(mod(C))^{\mathcal{I}}(x)$	$f_{mod}(C^{\mathcal{I}}(x))$
$(s_i \downarrow\downarrow C)^{\mathcal{I}}(x)$	$\inf_{z \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(x,z) \Rightarrow \inf_{y \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(z,y) \Rightarrow C^{\mathcal{I}}(y) \right\} \right\}$
$(s_i \uparrow\downarrow C)^{\mathcal{I}}(x)$	$\sup_{z \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(x,z) \otimes \inf_{y \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(z,y) \Rightarrow C^{\mathcal{I}}(y) \right\} \right\}$
$(s_i \downarrow C)^{\mathcal{I}}(x)$	$\inf_{y \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y) \right\}$
$(s_i \downarrow\uparrow C)^{\mathcal{I}}(x)$	$\inf_{z \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(x,z) \Rightarrow \sup_{y \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(z,y) \otimes C^{\mathcal{I}}(y) \right\} \right\}$
$(s_i \uparrow\uparrow C)^{\mathcal{I}}(x)$	$\sup_{z \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(x,z) \otimes \sup_{y \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(z,y) \otimes C^{\mathcal{I}}(y) \right\} \right\}$
$(s_i \uparrow C)^{\mathcal{I}}(x)$	$\sup_{y \in \Delta^{\mathcal{I}}} \left\{ s_i^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y) \right\}$
Role R	**Semantics**
$(R_A)^{\mathcal{I}}(x,y)$	$R_A^{\mathcal{I}}(x,y)$
$(T)^{\mathcal{I}}(x,y)$	$T^{\mathcal{I}}(x,y)$
$(U)^{\mathcal{I}}(x,y)$	γ_p
$(R^-)^{\mathcal{I}}(x,y)$	$R^{\mathcal{I}}(y,x)$
Axiom τ	**Semantics**
$\langle a{:}C \bowtie \gamma \rangle$	$C^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie \gamma$
$\langle (a,b){:}R \rhd \gamma \rangle$	$R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \rhd \gamma$
$\langle (a,b){:}\neg_G R \rhd \gamma \rangle$	$\ominus_G R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \rhd \gamma$
$\langle (a,b){:}\neg_Z R \rhd \gamma \rangle$	$\ominus_Z R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \rhd \gamma$
$\langle (a,v){:}T \rhd \gamma \rangle$	$T^{\mathcal{I}}(a^{\mathcal{I}}, v_{\mathbf{D}}) \rhd \gamma$
$\langle (a,v){:}\neg_G T \rhd \gamma \rangle$	$\ominus_G T^{\mathcal{I}}(a^{\mathcal{I}}, v_{\mathbf{D}}) \rhd \gamma$
$\langle (a,v){:}\neg_Z T \rhd \gamma \rangle$	$\ominus_Z T^{\mathcal{I}}(a^{\mathcal{I}}, v_{\mathbf{D}}) \rhd \gamma$
$a \neq b$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$
$a = b$	$a^{\mathcal{I}} = b^{\mathcal{I}}$
$\langle C \sqsubseteq_G D \rhd \gamma \rangle$	$\left(\inf_{x \in \Delta^{\mathcal{I}}} C^{\mathcal{I}}(x) \Rightarrow_G D^{\mathcal{I}}(x) \right) \rhd \gamma$
$\langle C \sqsubseteq_Z D \rhd \gamma \rangle$	$\left(\inf_{x \in \Delta^{\mathcal{I}}} C^{\mathcal{I}}(x) \Rightarrow_Z D^{\mathcal{I}}(x) \right) \rhd \gamma$
$\langle R_1 \ldots R_m \sqsubseteq_G R \rhd \gamma \rangle$	$\left(\inf_{x_1,x_{m+1} \in \Delta^{\mathcal{I}}} \sup_{x_2 \ldots x_m \in \Delta^{\mathcal{I}}} (R_1^{\mathcal{I}}(x_1,x_2) \otimes \cdots \otimes R_n^{\mathcal{I}}(x_m,x_{m+1})) \Rightarrow_G R^{\mathcal{I}}(x_1,x_{m+1}) \right) \rhd \gamma$
$\langle R_1 \ldots R_m \sqsubseteq_Z R \rhd \gamma \rangle$	$\left(\inf_{x_1,x_{m+1} \in \Delta^{\mathcal{I}}} \sup_{x_2 \ldots x_m \in \Delta^{\mathcal{I}}} (R_1^{\mathcal{I}}(x_1,x_2) \otimes \cdots \otimes R_n^{\mathcal{I}}(x_m,x_{m+1})) \Rightarrow_Z R^{\mathcal{I}}(x_1,x_{m+1}) \right) \rhd \gamma$
$\langle T_1 \sqsubseteq_G T_2 \rhd \gamma \rangle$	$\inf_{x \in \Delta^{\mathcal{I}}, v \in \Delta_{\mathbf{D}}} T_1^{\mathcal{I}}(x,v) \Rightarrow_G T_2^{\mathcal{I}}(x,v) \rhd \gamma$
$\langle T_1 \sqsubseteq_Z T_2 \rhd \gamma \rangle$	$\inf_{x \in \Delta^{\mathcal{I}}, v \in \Delta_{\mathbf{D}}} T_1^{\mathcal{I}}(x,v) \Rightarrow_Z T_2^{\mathcal{I}}(x,v) \rhd \gamma$
$\mathtt{trans}(R)$	$\forall x,y \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(x,y) \geqslant \sup_{z \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,z) \otimes R^{\mathcal{I}}(z,y)$
$\mathtt{dis}(S_1,S_2)$	$\forall x,y \in \Delta^{\mathcal{I}}, S_1^{\mathcal{I}}(x,y) = \gamma_0$ or $S_2^{\mathcal{I}}(x,y) = \gamma_0$
$\mathtt{dis}(T_1,T_2)$	$\forall x \in \Delta^{\mathcal{I}}, v \in \Delta_{\mathbf{D}}, T_1^{\mathcal{I}}(x,v) = \gamma_0$ or $T_2^{\mathcal{I}}(x,v) = \gamma_0$
$\mathtt{ref}(R)$	$\forall x \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(x,x) = \gamma_p$
$\mathtt{irr}(S)$	$\forall x \in \Delta^{\mathcal{I}}, S^{\mathcal{I}}(x,x) = \gamma_0$
$\mathtt{sym}(R)$	$\forall x,y \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(x,y) = R^{\mathcal{I}}(y,x)$
$\mathtt{asy}(S)$	$\forall x,y \in \Delta^{\mathcal{I}}$, if $S^{\mathcal{I}}(x,y) > \gamma_0$ then $S^{\mathcal{I}}(y,x) = \gamma_0$

3.3. Notation

Firstly, let us introduce some notation that will be used in the paper. C, D are (possibly complex) concepts, A is an atomic concept, R is a (possibly complex) role, R_A is an atomic role, S is a simple role (see the definition below), T is a concrete fuzzy role, a, b are abstract individuals, v is a concrete individual, $\mathbf{d}$ are fuzzy concrete predicates, n, m are natural numbers ($n \geqslant 0, m > 0$), s_i is a fuzzy similarity relation (reflexive and symmetric), *mod* is a fuzzy modifier, $\bowtie \in \{\geqslant, <, \leqslant, >\}$, $\triangleright \in \{\geqslant, >\}$, $\triangleleft \in \{\leqslant, <\}$. We will also use $\equiv$ to denote semantical equivalence.

3.4. Syntax

Fuzzy $\mathcal{SROIQ}$ assumes three alphabets of symbols, for concepts, roles and individuals. The syntax of fuzzy concepts and roles is shown in Table 3. In the following, we will use the generic term "fuzzy concepts" to denote also fuzzy rough concepts.

Remark 3.1. Note that, in contrast to to the classical case, there are two types of negations, universal restrictions and at-most restrictions, one corresponding to Gödel fuzzy logic and another one corresponding to Zadeh fuzzy logic (denoted with the subscripts $_G$ and $_Z$, respectively). However, there is only one type of conjunction, disjunction, existential or at-least restrictions because the semantics in both logics coincide.

Another differences with respect to the classical case are fuzzy nominals, modified concepts, cut concepts, fuzzy rough concepts, and concrete roles with fuzzy datatypes. Note that we consider singleton fuzzy nominals, since a complex fuzzy nominal $\{\alpha_1/o_1, \alpha_2/o_2, \ldots, \alpha_n/o_n\}$ can be represented as $\{\alpha_1/o_1\} \sqcup \{\alpha_2/o_2\} \sqcup \cdots \sqcup \{\alpha_n/o_n\}$.

A *Fuzzy Knowledge Base* (KB) contains a finite set of axioms organized in three parts: a fuzzy ABox $\mathcal{A}$ (axioms about individuals), a fuzzy TBox $\mathcal{T}$ (axioms about concepts) and a fuzzy RBox $\mathcal{R}$ (axioms about roles). A *fuzzy axiom* is an axiom that has a truth degree in $\mathcal{N}$. The axioms that are allowed in our logic are also shown in Table 3.

Remark 3.2. Note again the difference with respect to the classical case as there are two types of negated role assertions, fuzzy GCIs and fuzzy RIAs.

Simple roles are inductively defined as follows:

- R_A is simple if it does not occur on the right side of a fuzzy RIA.
- R^- is simple if R is.
- If R occurs on the right side of a fuzzy RIA, R is simple if the axiom is of the form $\langle S \sqsubseteq R \triangleright \gamma \rangle$, for a simple role S.

As in the non-fuzzy case, there are some restrictions in the use of roles, in order to guarantee the decidability of the logic (Horrocks et al., 2006):

- Concept constructors ($\geqslant m\ S.C$), ($\leqslant_G n\ S.C$), ($\leqslant_Z\ n\ S.C$), $\exists S.\mathtt{Self}$ require simple roles.
- RBox axioms $\mathtt{dis}(S_1, S_2)$, $\mathtt{irr}(S)$, $\mathtt{asy}(S)$ require simple roles.
- RBox axioms cannot contain the universal role U.
- Finally, given a regular order $\prec$, every RIA should be $\prec$-regular (Horrocks et al., 2006). Intuitively, regularity prevents a role hierarchy from containing cyclic dependencies.

3.5. Semantics

A fuzzy interpretation $\mathcal{I}$ with respect to a fuzzy concrete domain $\mathbf{D}$ is a pair $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a non-empty set $\Delta^{\mathcal{I}}$ (the interpretation domain) disjoint with $\Delta_{\mathbf{D}}$ and a fuzzy interpretation function $\cdot^{\mathcal{I}}$ mapping:

- Every abstract individual a onto an element $a^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$.
- Every concrete individual v onto an element $v_{\mathbf{D}}$ of $\Delta_{\mathbf{D}}$.
- Every fuzzy rough concept C onto a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to \mathcal{N}$.
- Every fuzzy abstract role R onto a function $R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to \mathcal{N}$.
- Every fuzzy concrete role T onto a function $T^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta_{\mathbf{D}} \to \mathcal{N}$.
- An n-ary fuzzy *concrete predicate* $\mathbf{d}$ onto the fuzzy relation $\mathbf{d}_{\mathbf{D}} : \Delta_{\mathbf{D}}^n \to \mathcal{N}$.
- A fuzzy *modifier mod* onto a function $f_{mod} : \mathcal{N} \to \mathcal{N}$.

The fuzzy interpretation function is extended to fuzzy *complex concepts, roles* and *axioms* as shown in Table 4.

We use $\otimes$ for denoting Gödel (maximum) t-norm, $\oplus$ for Gödel (minimum) t-conorm, $\ominus_G$ for Gödel negation, $\ominus_Z$ for Łukasiewicz negation, $\Rightarrow_G$ for Gödel implication, and $\Rightarrow_Z$ for Kleene-Dienes implication.

$C^{\mathcal{I}}$ denotes the membership function of the fuzzy concept C with respect to the fuzzy interpretation $\mathcal{I}$. $C^{\mathcal{I}}(x)$ gives us the degree of being x an element of the fuzzy concept C under $\mathcal{I}$. Similarly, $R^{\mathcal{I}}$ denotes the membership function of the fuzzy role R with respect to $\mathcal{I}$. $R^{\mathcal{I}}(x, y)$ gives us the degree of being (x,y) an element of the fuzzy role R.

Note an important difference with the previous work in fuzzy DLs, which maps every concept C onto a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0, 1]$, and every role R onto a function $R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0, 1]$. The current semantics is more natural, and allows a representation involving ontologies of lower size (Bobillo et al., Submitted).

Assuming a semantics in [0,1], a fuzzy KB $\{\langle a : C > 0.5\rangle, \langle a : C < 0.75 \rangle\}$ is satisfiable, by taking any $C^{\mathcal{I}}(a) \in (0.5, 0.75)$. But now, given $\mathcal{N} = \{0, 0.25, 0.5, 0.75, 1\}$, the same KB is not satisfiable, since $C^{\mathcal{I}}(a) \in (0.5, 0.75)$ is in contradiction with the fact that $C^{\mathcal{I}}(a) \in \mathcal{N}$.

3.6. Definable axioms

As in the non-fuzzy case, fuzzy GCIs can be used to express some interesting axioms.

- *Concept/role equivalence*. $C_1, C_2 \ldots C_m$ are equivalent, denoted as $C_1 \equiv C_2 \equiv \cdots \equiv C_m$, can be expressed with the axioms $\langle C_i \sqsubseteq C_j \geqslant 1\rangle$, $\langle C_j \sqsubseteq C_i \geqslant 1\rangle$ with $1 \leqslant i, j \leqslant m$, $i \neq j$. The case of roles is exactly the same.
- *Disjointness of concepts*. The fact that $C_1 \ldots C_m$ are disjoint can be expressed as $\langle C_i \sqcap C_j \sqsubseteq \bot \geqslant 1\rangle$.
- *Disjoint union of concepts*. C_1 can be defined as a disjoint union of concepts $C_2 \ldots C_m$ with the axioms $C_1 \equiv C_2\, C_m$ and disjoint $(C_2, \ldots, C_m)$.
- *Domain* of a role. The fact that concept C is the domain of a role R can be expressed as $\langle \top \sqsubseteq \forall R^-.C \geqslant 1\rangle$ or $\langle \exists R.\top \sqsubseteq C \geqslant 1\rangle$.
- *Range* of a role. The fact that concept C is the range of a role R can be expressed as $\langle \top \sqsubseteq \forall R.C \geqslant 1\rangle$.
- *Functionality* of a role. The fact that a role R is functional can be expressed as $\langle \top \sqsubseteq (\leqslant 1\ R.\top) \geqslant 1\rangle$.

3.7. Reasoning tasks

In fuzzy rough DLs, there are many reasoning tasks. Usually, they can be reduced to fuzzy KB satisfiability (Straccia, 2001).

- *Fuzzy KB satisfiability*. A fuzzy interpretation $\mathcal{I}$ *satisfies* (is a model of) a fuzzy KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R}\rangle$ iff it satisfies each element in $\mathcal{A}$, $\mathcal{T}$ and $\mathcal{R}$.

- *Concept satisfiability*. C is α-satisfiable w.r.t. a fuzzy KB $\mathcal{K}$ iff $\mathcal{K} \cup \{\langle a : C \geqslant \alpha \rangle\}$ is satisfiable, where a is a new individual, not appearing in $\mathcal{K}$.
- *Entailment*: A fuzzy concept assertion $a : C \bowtie \alpha$ is entailed by a fuzzy KB $\mathcal{K}$ (denoted $\mathcal{K} \models \langle a : C \bowtie \alpha \rangle$) iff $\mathcal{K} \cup \{\langle a : C \neg \bowtie \alpha \rangle\}$ is unsatisfiable. The case for fuzzy role assertions is similar. The negation $\neg\bowtie$ of an operator $\bowtie$ is defined as follows: $\neg \geqslant = <$, $\neg > = \leqslant$, $\neg \leqslant = >$, $\neg < = \geqslant$.
- *Concept subsumption*: D subsumes C (denoted $C \sqsubseteq D$) w.r.t. a fuzzy KB $\mathcal{K}$ iff, for every $\alpha \in \mathcal{N}$, $\mathcal{K} \cup \{a : C \geqslant \alpha\} \cup \{a : D < \alpha\}$ is unsatisfiable, where a is a new individual.
- *Best degree bound* (BDB). The BDB of a concept or role assertion τ is defined as the $\sup\{\alpha : \mathcal{K} \models \langle \tau \geqslant \alpha \rangle\}$. It can be computed as a binary search in $\mathcal{N}$, performing at most $\log |\mathcal{N}|$ entailment tests (Straccia, 2001).
- *Maximal concept satisfiability degree*. The maximal satisfiability degree of a fuzzy concept C w.r.t. a fuzzy KB $\mathcal{K}$ is defined as the $\sup\{\beta : \mathcal{K} \models \langle a : C > \beta \rangle\}$, where a is a new individual. It can be computed as a binary search in $\mathcal{N}$, performing at most $\log |\mathcal{N}|$ entailment tests.

3.8. Reasoning in FUZZYDL

Aside from the development of customized inference procedures, which is a difficult task, reasoning in $Z\ \mathcal{SROIQ}(\mathbf{D})$ (Bobillo et al., 2009) and in $G\ \mathcal{SROIQ}(\mathbf{D})$ (Bobillo et al., 2009) is possible by providing a reduction to an equivalent non-fuzzy ontology, in such a way that current non-fuzzy DL reasoners can then be used. These algorithms can be combined, as shown in Bobillo et al. (Submitted). The integration of fuzzy rough sets and fuzzy DL is seamless, as the rough set component can mapped into the fuzzy DL component, as shown in Bobillo & Straccia (2009); Bobillo & Straccia (submitted).

An interesting property from a practical point of view of the non-fuzzy representations is that, under a certain condition, the reduction of an ontology can be reused when adding new axioms and only the reduction of the new axioms has to be included. This condition is that the new axiom does not introduce new atomic concepts, atomic roles, or degrees of truth.

Restricting the set of degrees of truth in the language is necessary in Gödel logic for theoretical reasons (Bobillo et al., 2009), but it is also convenient for practical reasons. As we have already mentioned, it allows reusing the non-fuzzy representations. Furthermore, the complexity of the resulting non-fuzzy KBs is reduced (linear with the size of the input ontology), and the computation of some reasoning tasks that require a binary search becomes more feasible.

3.9. A motivating example

We conclude this section, with an example illustrating the expressivity of fuzzy ontologies as well as the usefulness of the reasoning tasks to derive new information about the explicitly represented knowledge.

A known issue in health-care support is that consensus in the used vocabulary is required to achieve understanding among different physicians and systems. Medical taxonomies are an effort in this direction, as they provide a well-defined catalogue of codes to label diseases univocally. Two examples are ICD[2] (for general medicine) and DSM-IV (American Psychiatric Association, 1994) (for mental disorders), which identify prototypical clinical medical profiles with a name and a code. Medical taxonomies have been developed to be non-fuzzy, so they can be transcribed almost directly to OWL.

However, vagueness could be introduced at different levels of the taxonomy so that richer semantics would be represented:

- In order to associate diagnostic codes to patient electronic records, fuzzy assertions would be be useful, allowing the knowledge base to contain statements such as "Patient001's Serotonin Level is quite low" or "Patient001's disease is likely to be a Substance-Induced Anxiety Disorder". This can be represented using the following fuzzy concept assertions:

$$\langle \mathsf{patient001} : \exists \mathsf{hasSerotoninLevel.LowLevel} \geqslant \mathtt{closeToTrue} \rangle, \langle \mathsf{patient001} : \exists \mathsf{hasDisease.SubstanceInducedAnxietyDisorder} \geqslant \mathtt{closeToTrue} \rangle,$$

- In the current version of DSM-IV, "Substance-Induced Anxiety Disorder" is defined as a subclass of "Substance-Related Disorder". A fuzzy GCI could refine this definition by stating that a "Substance-Induced Anxiety Disorder can be partially considered a Substance-Related Disorder", as well as an "Anxiety Disorder".

$$\langle \mathsf{SubstanceInducedAnxietyDisorder} \sqsubseteq_G \mathsf{SubstanceRelatedDisorder} \geqslant \mathtt{closeToTrue} \rangle$$
$$\langle \mathsf{SubstanceInducedAnxietyDisorder} \sqsubseteq_G \mathsf{AnxietyDisorder} \geqslant \mathtt{true} \rangle$$

We assume, for instance, a set of degrees of truth {`false`, `closeToFalse`, `neutral`, `closeToTrue`, `true`}.

Now, we can use a fuzzy DL reason to obtain new knowledge. For instance, it holds that $BDB(\mathsf{patient001}{:}\exists\mathsf{hasDisease.\ AnxietyDisorder}) = \mathtt{closeToTrue}$, even if this information was not implicitly represented in the original KB.

Also, the concept $\mathsf{SubstanceInducedAnxietyDisorder} \sqcap \neg\ \mathsf{SubstanceRelatedDisorder}$ is `neutral`-satisfiable, which does not happen in the non-fuzzy case.

4. DeLorean fuzzy DL reasoner

This section describes in detail our prototype implementation, called DeLorean (DEscription LOgic REasoner with vAgueNess),[3] the first reasoner that supports fuzzy extensions of the languages OWL and OWL 2.

In a strict sense, DeLorean is not a reasoner but a translator from fuzzy (rough) ontologies to standard ontologies. Given a fuzzy rough ontology (in a $GZ\ \mathcal{SROIQ}(\mathbb{D})$-based language, e.g. the one presented in Section 4.3), it obtains an equivalent non-fuzzy representation (in OWL or OWL2, depending on the expressivity of the original ontology). A classical DL reasoner can be used to reason with the resulting ontology. Since DeLorean transparently performs the reduction procedure and the reasoning with an underlying DL reasoner, we refer to it simply as a reasoner.

This section is organized as follows. Section 4.1 describes the architecture, features and different versions of the reasoner. Section 4.2 presents the graphical interface and the functionality of the reasoner. The syntax of the fuzzy language and the functions of the Application Programming Interface (API) are analyzed in Sections 4.3 and 4.4 respectively. Finally, Sections 4.5 describes some of the implemented optimizations.

[2] http://www.who.int/classifications/icd/en.

[3] http://www.webdiis.unizar.es/fbobillo/delorean.

4.1. Architecture, features and versions

4.1.1. DeLorean 1.0

In 2007 we developed this first version by using the Java programming language, the JavaCC parser generator, the Jena API, and the DIG 1.1. interface.

JavaCC[4] (Java Compiler Compiler) was used to read the inputs. It is an open-source parser generator for the Java programming language. Given a formal specification in EBNF (Extended Backus-Naur Form) notation of a grammar, it produces as output the Java source code of the parser.

Jena (McBride, 2002)[5] is a very popular Semantic Web API. It is an open-source framework which includes an RDF API, a SPARQL query engine and an OWL API. Jena does not allow directly to reason with OWL ontologies, but it can use an external reasoner by means of the DIG interface.

DIG (Description logic Implementation Group) (Bechhofer, Möller, & Crowther, 2003) was used to communicate with classical DL reasoners. It is a common interface to access DL reasoners, avoiding the need to know the particularities of the representation languages of all of them. The latest version is 1.1 and it supports OWL DL without datatypes ($\mathcal{SHOIQ}$). There was a project to develop a newer version, DIG 2.0,[6] supporting $\mathcal{SROIQ}$ (Turhan et al., 2006), but it never came out.

The use of Java makes DeLorean platform independent. Moreover, DeLorean can take advantage of any DL reasoner as long as it supports DIG interface. However, DIG interface 1.1 does not support full $\mathcal{SROIQ}$, so the logic supported by this first version of DeLorean was restricted to $Z\ \mathcal{SHOIN}$ (which is reduced to OWL DL). From a historical point of view, this version was the first reasoner that supported a fuzzy extension of the OWL DL language (Bobillo, Delgado, & Gómez-Romero, 2007).

4.1.2. DeLorean 1.1

With the aim of augmenting the expressivity of the logic, we implemented a second version which used OWL API 2 instead of Jena API.

OWL API 2 is an open-source API for manipulating OWL 2 ontologies (Horridge et al., 2007). It extends the previous OWL API (or WonderWeb API) (Bechhofer, Lord, & Volz, 2003), which only supports OWL ontologies. Applications can use some integrated DL reasoners such as Pellet (Sirin, Parsia, Cuenca-Grau, Kalyanpur, & Katz, 2007) and FaCT++ (Tsarkov & Horrocks, 2006), but the API also supports integration with DIG-compliant DL reasoners. The API does not support the universal role.

In this version, DeLorean supported the fuzzy DLs $Z\ \mathcal{SROIQ}(\mathbb{D})$ and $G\ \mathcal{SROIQ}(\mathbb{D})$. The only limitation is that the universal role cannot be used, since OWL API 2 did not allow it. From a historical point of view, DeLorean was the first reasoner that supports a fuzzy extension of OWL 2.

Since DIG interface did not allow the full expressivity of OWL 2, our solution was to integrate directly DeLorean with a concrete DL reasoner, *Pellet* (Sirin et al., 2007),[7] which can be directly used from OWL API 2. This way, the user was free to choose to use either a generic DL reasoner (restricting the expressivity to $\mathcal{SHOIQ}$) or *Pellet* (with no expressivity limitations).

4.1.3. DeLorean 2

The current version presents many differences with the previous one: more expressivity of the language (it supports the fuzzy rough DL $GZ\ \mathcal{SROIQ}(\mathbb{D})$), more functionality (it supports several reasoning tasks and not only KB consistency), and use of the modern SW technologies, since we have replaced OWL API 2 with OWL API 3.

OWL API 3 is the latest version of OWL APIs (Horridge & Bechhofer, 2009).[8] It has an important impact on client code. One of the most important differences is that the API replaces DIG support with OWLlink support. OWL API 3 is becoming a de facto standard and many SW tools already support it, such as the ontology editor *Protégé* 4.1[9] (Noy et al., 2001), or the DL reasoners *Pellet* 2.1, *Fact++* 1.3 (Tsarkov & Horrocks, 2006, Horridge, Tsarkov, & Redmond, 2006)[10] and *HermiT* 1.2.2 (Shearer, Motik, & Horrocks, 2008).[11]

OWLlink[12] is an extensible protocol for communication among OWL2-aware systems (Liebig, Luther, & Noppens, 2009). OWLlink is based on the DIG 2 proposal and aims to replace the outdated DIG 1.1 protocol. At the current date, the only DL reasoner that supports OWLlink is *RacerPro* (Haarslev & Möller, 2001),[13] but this situation is expected to change soon.

Because OWLlink is not widely supported yet, we have also integrated directly DeLorean with *Pellet* and *HermiT* reasoners. Hence, the user is free to choose either one of these reasoners or a generic one via OWLlink protocol.

4.1.4. Architecture

Fig. 2 illustrates the architecture of the system:

- The *Parser* reads a fuzzy rough ontology contained in an input physical URI and translates it into an internal representation of a fuzzy KB. Interestingly, we can use any language to encode the fuzzy rough ontology, as long as this module can understand the representation. Moreover, we could have several parsers, each of them being responsible of the translation of a different fuzzy rough ontology language.
- The *Reduction* module implements the reduction of the fuzzy rough ontology into an equivalent non-fuzzy ontology. In particular, it builds an OWL API 3 model with an equivalent non-fuzzy ontology, which can be exported to an OWL 2 file. The implementation also takes into account the optimizations described in Section 4.5.
- The *Inference* module communicates with a non-fuzzy reasoner (either one of the integrated reasoners or a reasoner via OWLlink protocol) in order to perform different reasoning tasks, as described in the next section.
- A simple *User interface* manages inputs and outputs. More details are given in the next section.

4.2. Using DeLorean

4.2.1. User interface

A snapshot of the user interface is shown in Fig. 3. It is structured in four parts:

Input. Here, the user can specify the input fuzzy ontology, using the syntax explained in Section 4.3, and the DL reasoner that will be used in the reasoning. The possible choices are *HermiT*, *Pellet*, and a OWLlink-complaint reasoner. Once a fuzzy ontology is loaded, the reasoner will check that every degree of truth that appears in it belongs to the set specified in the section on the right.

[4] https://javacc.dev.java.net.
[5] http://www.jena.sourceforge.net.
[6] http://www.dig.cs.manchester.ac.uk.
[7] http://www.clarkparsia.com/pellet.
[8] http://www.owlapi.sourceforge.net.
[9] http://www.protege.stanford.edu.
[10] http://www.owl.man.ac.uk/factplusplus.
[11] http://www.hermit-reasoner.com.
[12] http://www.owllink.org.
[13] http://www.racer-systems.com/products/racerpro/index.phtml.

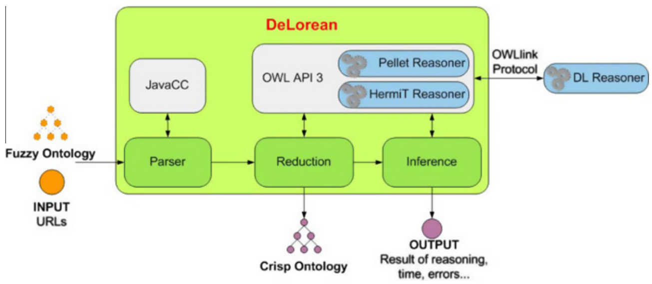

Fig. 2. Architecture of DeLorean reasoner.

Degrees of truth. The user can specify here the set of degrees of truth that will be considered. 0 (false) and 1 (true) are mandatory. Other degrees can be added, ordered (by moving them up or down), and removed. For the user convenience, it is possible to directly specify a number of degrees of truth, and they will be generated directly.

Reasoning. This part is used to perform the different reasoning tasks that DeLorean supports. A more detailed description of the different reasoning tasks is given below.

Output. Here, output messages are displayed. Some information about the reasoning is shown here, such as the time taken, or the result of the reasoning task asked.

4.2.2. Reasoning tasks

The panel *Reasoning* of the user interface is divided into five tabs, each of them dedicated to a specific reasoning task (we recall the reader that these reasoning tasks have been defined in Section 3.7).

Crisp representation. The main reasoning task is the computation of the equivalent *crisp representation* of the fuzzy ontology, which is actually necessary for the other reasoning tasks. In this tab we can export the resulting non-fuzzy ontology into a new OWL 2 file. This tab can be seen in Fig. 3.

Satisfiability. In this tab (see Fig. 4 (a)), the user can perform three tasks: *fuzzy KB consistency*, *fuzzy concept satisfiability* and the computation of the *maximum degree of satisfiability of a fuzzy concept*. In the two latter cases, the interface makes it possible to specify the name of the fuzzy concept for which the satisfiability test will be computed. Note that the interface expects the name of a fuzzy concept, and not a concept expression.

Entailment. In this tab (see Fig. 4 (b)), the user can compute, given the current fuzzy ontology, the *entailment* of a fuzzy concept assertion or a fuzzy role assertion. Firstly, the user has to specify the type of the assertion, and then the corresponding parameters. For fuzzy concept assertions, the parameters are: name of the individual, name of the fuzzy concept, inequality sign, and degree of truth. For fuzzy role assertions, the parameters are: name of the subject individual, name of the role, name of the object individual, inequality sign, and degree of truth for fuzzy role assertions.

Subsumption. In this tab (see Fig. 4 (c)), it is possible to compute *fuzzy concept subsumption*. It is firstly necessary to specify the names of both the subsumed fuzzy concept and the subsumer fuzzy concept.

BDB. Finally, in the fifth tab (see Fig. 4 (d)), the user can compute the BDB of a fuzzy concept assertion or a fuzzy role assertion. As in the case of entailment, previously the user has to specify the type of the assertion and the corresponding parameters. For fuzzy concept assertions, the parameters are: name of the individual, and name of the fuzzy concept. For fuzzy role assertions, the parameters are: name of the subject individual, name of the role, and name of the object individual.

4.3. Language to encode the ontologies

The input fuzzy rough ontology can be submitted to DeLorean in two ways. The first option is to load a file storing the fuzzy ontology encoded in a proper language. The second option is to create a new ontology programmatically by using the reasoner API. In this section we shall focus on the first option, describing the syntax of the language supported by DeLorean. The second option will be studied in Section 4.4.

Let us remark that we support alternative languages to encode the fuzzy rough ontology. It is only necessary to develop a new parser translating the new language into the internal representation of the reasoner.

The language we have defined assumes three alphabets of symbols, for individuals, concepts (fuzzy sets of individuals), and roles (fuzzy binary relation between individuals). We use the notation

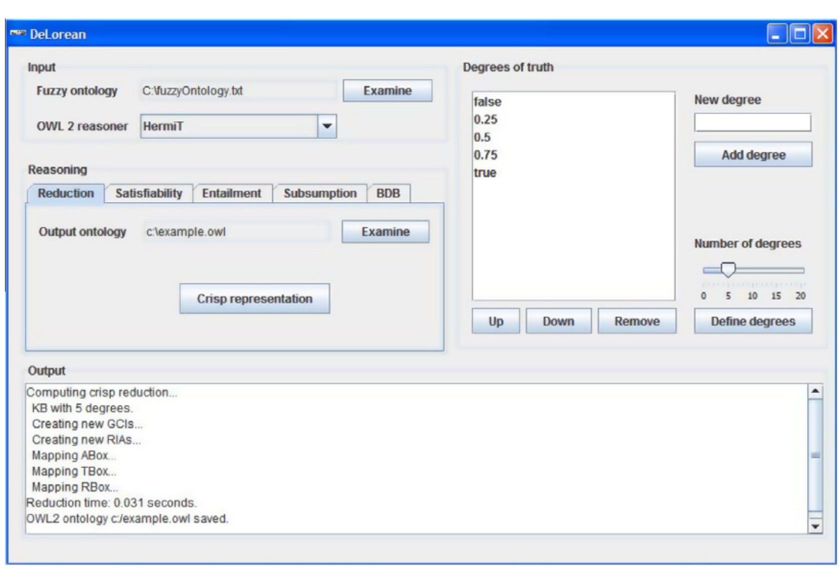

Fig. 3. User interface of DeLorean reasoner.

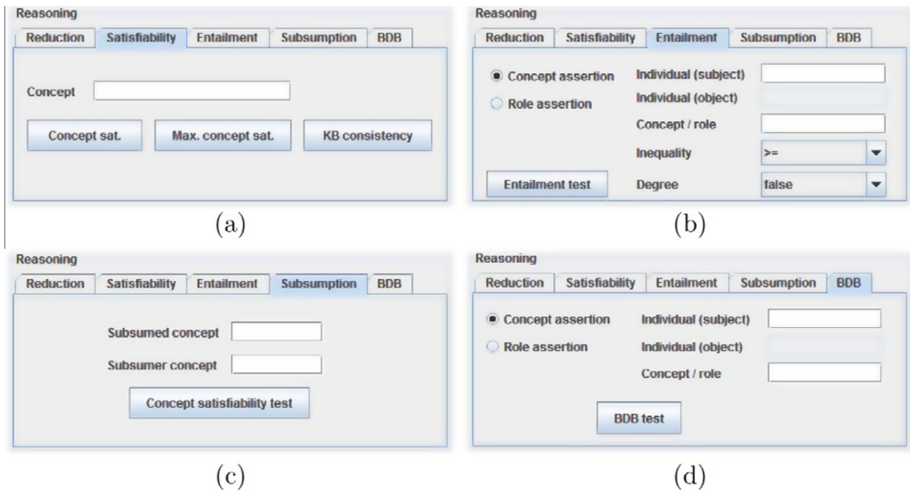

Fig. 4. (a) Concept satisfiability and KB consistency; (b) Concept and role entailment; (c) Concept subsumption; (d) BDB.

introduced in Section 3). The syntax of axioms and fuzzy concepts draws inspiration from the Knowledge Representation System Specification (Patel-Schneider & Swartout, 1993).

4.3.1. Concepts

Concepts are defined according to the syntax defined in Table 5. Note the similarity with the fuzzy DL syntax. Some syntactic sugar has been added: the extension of *n*-ary conjunctions and disjunctions, and the exactly cardinality constructors (a conjunction of at-least and at-most restrictions). In the case of the concrete concepts, the only fuzzy datatype supported is the trapezoidal functions, which can be defined as shown in Table 5.

4.3.2. Roles

Roles are either atomic or the universal role, as shown in Table 5. Inverse roles are defined using axioms, as we will see below.

4.3.3. Axioms

The axioms of a fuzzy KB are defined according to the syntax defined in Table 5. Note that we allow some useful syntactic sugar axioms (role domain, role range, role functionality, concept disjointness), as well as statements imposing that concepts or roles should be interpreted as crisp.

4.3.4. Restrictions

Note that, in fuzzy nominal concepts, if α is omitted then 1 is assumed. Also, whenever $\bowtie\gamma$ or $\triangleright\gamma$ is omitted in axioms, $\geqslant 1$ is assumed.

In cardinality restrictions, if C or $\mathbf{d}$ are omitted, $\top$ is assumed. In RIAs involving concrete roles, $n = 1$ (there cannot be a role chain in the left side).

There are some restrictions in the degrees. If $\bowtie$ is $\geqslant$ or $<$, then γ cannot be γ_0; whereas if $\bowtie$ is $>$ or $\leqslant$, then γ cannot be γ_p.

4.3.5. Importing OWL 2 ontologies

In order to make the representation of fuzzy KBs easier, we have also developed a parser (OWL2Parser class) that allows importing OWL 2 ontologies. These (non-fuzzy) ontologies are saved as a text file that the user can edit and extend, for example by adding membership degrees to the fuzzy axioms or specifying a particular fuzzy operator (Zadeh or Gödel family) for some complex concept.

4.4. Fuzzy ontology management and reasoning with the DELOREAN API

As we have seen, DeLorean can be used as a stand-alone application. Additionally, DeLorean services can be used from other programs by means of the DeLorean API. The DeLorean API is a Java library that allows fuzzy ontology management and reasoning. Briefly, the DeLorean API provides support for fuzzy ontology creation, fuzzy axiom definition, and fuzzy reasoning. These basics of programming with the DeLorean API are depicted in Fig. 5. The figure shows a code snippet implementing the three main features of the API:

- Fuzzy ontology creation.
- Population of the fuzzy ontology with axioms.
- Preparation and execution of reasoning tasks.

The remaining part of this section explains in detail which functionalities are provided. More information can be found in the Javadoc documentation provided with the reasoner.

4.4.1. Fuzzy ontology creation

The first step to use the reasoner is to create a new fuzzy ontology. This new ontology can be empty or, alternatively, we can load an existing ontology file with the syntax specified in Section 4.3. Reading and parsing an ontology file is performed by the Parser class, which is a factory to obtain a KnowledgeBase instance. Interestingly enough, the programmer can use a custom implementation of the Parser read ontology method.

4.4.2. Individuals, roles, and concept creation

The KnowledgeBase class provides methods to retrieve individuals, atomic roles, and atomic concepts previously defined in the fuzzy ontology. In general, these retrieval methods returns the entity with the specified name, creating it in case there does not already exist an entity with such a name. The retrieval methods return instances of the classes Individual, Role, and Concept, respectively.

Complex concepts can be obtained by using the different methods of the Concept class (see Table 6). All these methods return a Concept instance.

When the semantics of the concept depends on the choice of the fuzzy operator, it is required to use the appropriate method. This is the case for all, complement, and qatmost. Gödel semantics are noted by prepending the goedel identifier to the method name. Note also that some concept constructors can be applied to

Table 5
Syntax for fuzzy concepts and axioms in DeLorean.

DeLorean concept syntax	fuzzy DL syntax
A	A
top	$\top$
bottom	$\bot$
(and $C_1 \ldots C_m$)	$C_1 \sqcap \cdots \sqcap C_m$
(or $C_1 \ldots C_m$)	$C_1 \sqcup \cdots \sqcup C_m$
(g-not C)	$\neg_Z C$
(z-not C)	$\neg_G C$
(some R C)	$\exists R.C$
(some R d)	$\exists R.\mathbf{d}$
(g-all R C)	$\forall_G R.C$
(g-all R d)	$\forall_G R.\mathbf{d}$
(z-all R C)	$\forall_Z R.C$
(z-all R d)	$\forall_Z R.\mathbf{d}$
(one-of a [α])	$\{\alpha/a\}$
(self S)	$\exists S.\mathtt{Self}$
(cut C α)	$[C \geqslant \alpha]$
(at-least M S [C])	$(\geqslant m\ S.C)$
(at-least M S [d])	$(\geqslant m\ S.\mathbf{D})$
(g-at-most N S [C])	$(\leqslant_G m\ S.C)$
(g-at-most N S [d])	$(\leqslant_G m\ S.\mathbf{D})$
(z-at-most N S [C])	$(\leqslant_Z m\ S.C)$
(z-at-most N S [d])	$(\leqslant_Z m\ S.\mathbf{D})$
(exactly M S [C])	$(\geqslant m\ S.C) \sqcap (\leqslant_Z m\ S.C)$
(exactly M S [d])	$(\geqslant m\ S.\mathbf{D}) \sqcap (\leqslant_Z m\ S.\mathbf{D})$
(triangular t1 t2 t3 C)	$f_{tri(t1,t2,t3)}(C)$
(upper s C)	$(s_i{\uparrow}C)$
(tight-upper s C)	$(s_i{\downarrow\uparrow}C)$
(loose upper s C)	$(s_i{\uparrow\uparrow}C)$
(lower s C)	$(s_i{\downarrow}C)$
(tight-lower s C)	$(s_i{\downarrow\downarrow}C)$
(loose lower s C)	$(s_i{\uparrow\downarrow}C)$
DeLorean role syntax	**fuzzy DL syntax**
R	R
topRole	U
DeLorean datatype syntax	**fuzzy DL syntax**
(trapezoidal q1 q2 q3 q4)	$\mathbf{d} = trap(q_1,q_2,q_3,q_4)$
DeLorean axiom syntax	**fuzzy DL syntax**
(instance a C $\bowtie \gamma$)	$\langle a{:}C \bowtie \gamma\ \rangle$
(related a b $R \rhd \gamma$)	$\langle (a,b): R \bowtie \gamma \rangle$
(g-not-related a b $R \rhd \gamma$)	$\langle\ (a,b){:}\neg_G R \bowtie \gamma \rangle$
(z-not-related a b $R \rhd \gamma$)	$\langle\ (a,b){:}\neg_Z R \bowtie \gamma \rangle$
(same-as a b)	$a = b$
(different-to a b)	$a \neq b$
(g-implies-concept C_1 $C_2 \rhd \gamma$)	$\langle C_1\ \sqsubseteq_G C_2 \rhd \gamma \rangle$
(kd-implies-concept C_1 $C_2 \rhd \gamma$)	$\langle C_1\ \sqsubseteq_Z C_2 \rhd \gamma \rangle$
(equivalent-concepts $C_1 \ldots C_m$)	$\langle C_1 \equiv \cdots \equiv C_m \rangle$
(disjoint-concepts $C_1 \ldots C_m$)	disjoint $(C_1 \ldots C_m)$
(disjoint-union-concept $C_1 \ldots C_m$)	$\{C_1 \equiv C_2 \sqcup \cdots \sqcup C_m$, disjoint $(C_2 \ldots C_m)\}$
(crisp-concept C)	$C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to \{\gamma_0, \gamma_p\}$
(g-implies-role R_1 $R_2 \ldots R_m$ $R \rhd \gamma$)	$\langle R_1 \ldots R_m \sqsubseteq_G R \rhd \gamma \rangle$
(kd-implies-role R_1 $R_2 \ldots R_m$ $R \rhd \gamma$)	$\langle R_1 \ldots R_m \sqsubseteq_Z R \rhd \gamma\ \rangle$
(equivalent-roles R_1 R_2)	$R_1 \equiv R_2$
(inverse R_1 R_2)	$R_1 \equiv R_2^-$
(domain R C)	$\langle \exists R.\top \sqsubseteq C \geqslant\ 1 \rangle$
(g-range R C)	$\langle \top \sqsubseteq_G \forall_G R.C \geqslant 1 \rangle$
(z-range R C)	$\langle \top \sqsubseteq_G \forall_Z R.C \geqslant 1 \rangle$
(functional R)	$\langle \top \sqsubseteq (\leqslant_Z 1\ R.\top) \rangle$
(inverse-functional R)	$\langle \top \sqsubseteq (\leqslant_Z 1\ R^-.\top) \rangle$
(transitive R)	$\mathtt{trans}(R)$
(symmetric R)	$\mathtt{sym}(R)$
(asymmetric S)	$\mathtt{asy}(S)$
(reflexive R)	$\mathtt{ref}(R)$
(irreflexive S)	$\mathtt{irr}(S)$
(disjoint-roles $S_1 \ldots S_m$)	disjoint $(S_1 \ldots S_m)$
(crisp-role R)	$R^{\mathcal{I}} : \Delta^{\mathcal{I}} \to \{\gamma_0, \gamma_p\}$

abstract and concrete roles (all, goedelAll, qatleast, qatmost, some). Therefore, we provide two different methods in each case, requiring respectively a Concept and a TrapezoidalNumber as parameter. The class TrapezoidalNumber manages trapezoidal membership functions, and requires the four values (q_1, q_2, q_3, q_4) defining the function.

4.4.3. Adding axioms to the fuzzy ontology

The KnowledgeBase class provides several methods to add axioms to the fuzzy ontology (see Table 7). Individuals, roles, and concepts involved in the axioms are created as already explained.

The names of the methods are self-descriptive, and there are only a couple of remarks to notice. Firstly, the DeLorean API does not offer different methods for creating axioms involving concrete and abstract roles. The reasoner is responsible for maintaining apart a list of concrete roles and a list of abstract roles and checking that they are disjoint. Secondly, as it occurs in the case of complex concept constructors, whenever the semantics of an axiom depends on the choice of the fuzzy operator, the API offers different methods to create it. This is the case for GCIs, RIAs, role range axioms, and negated role assertions.

Inequalities, in turn, are defined as values of the enumerated type Inequality. Degrees of truth are managed with the class Degree, which represents a linguistic label. The degrees of truth allowed for a fuzzy ontology must be previously set with the setDegrees method. This method restricts the degrees of truth than can be used in subsequent axioms.

4.4.4. Performing reasoning tasks

To perform a reasoning task with a fuzzy ontology, the API provides de DeLorean class. To create an instance of DeLorean, we only have to specify the fuzzy ontology (a KnowledgeBase instance) and the namespace of this ontology. Next, it is necessary to specify the DL reasoner that will be used. As we have seen, there are three possible choices of DL reasoner, which can be indicated with the static constants defined in the Owl2Reasoner class.

After having created the fuzzy ontology, and specified the set of degrees of truth and the DL reasoner, it is possible to perform reasoning tasks. Before running more complex reasoning tasks, it is necessary to compute the equivalent non-fuzzy representation of the fuzzy ontology. The reduction can be performed with the normal procedure or with an optimized procedure (see Section 4.5). After the reduction, several methods for computing different reasoning tasks can be called: ontology consistence, entailment, BDB, concept satisfiability, maximum concept satisfiability degree, and subsumption (see Table 8).

4.5. Optimizations

In this section we overview the main optimizations that the reasoner implements. These optimizations have proved to be very useful in reducing the size of the equivalent non-fuzzy ontology. A more detailed exposition of the benefits of the optimizations is not possible without providing full details of the reasoning algorithm, which is out of the scope of this paper. For more technical details and proofs, as well as a careful evaluation of their impact, both at a theoretical and a practical level, we refer the reader to Bobillo, Delgado, & Gómez-Romero (2008), Bobillo et al. (2009).

4.5.1. Minimizing the number of final concepts and axioms

Algorithms to obtain non-fuzzy representations for fuzzy DLs are based on the creation of new concepts representing α-cuts of the fuzzy concepts. In particular, for every fuzzy atomic concept A and every $\gamma \in \mathcal{N}$, four new concepts are created: $A_{\geqslant\gamma}$, $A_{>\gamma}$, $A_{\leqslant\gamma}$, $A_{<\gamma}$. These new concepts require some new axioms to maintain their semantics; for instance, $A_{>\gamma} \sqsubseteq A_{\geqslant\gamma}$. The case of roles is similar.

DeLorean uses a more optimized representation. For every fuzzy atomic concept A and every $\gamma \in \mathcal{N}$, DeLorean creates at-most only one (non-fuzzy) concept. Consequently, the number of new axioms is considerably reduced.

```
import edu.es.ugr.arai.delorean.*;
import java.util.Vector;
import java.util.zip.DataFormatException;

public class UsingAPI {

    public static void main(String args[])
    {
        // Create KB
        KnowledgeBase kb = new KnowledgeBase();

        // Select degrees of truth
        Vector<Degree> deg = new Vector<Degree>();
        deg.add (new Degree ("false"));
        deg.add (new Degree ("neutral"));
        deg.add (new Degree ("true"));
        kb.setDegrees(new DegreeSet (deg));

        // Add axioms to the KB
        Individual i = kb.getIndividual ("a");
        Concept c = new Concept ("C");
        kb.addConceptAssertion (i, c, Inequality.GREATER_EQUAL,
                                deg.get(1));

        // Setup reasoner and reasoning
        DeLorean r = new DeLorean (kb,"http://arai.ugr.es/ont.owl");
        r.setReasoner (Owl2Reasoner.PELLET_REASONER);

        try {
            // Compute reduction
            r.reduceKB();

            // Perform reasoning tasks
            System.out.println ("Consistent? " +
                                r.isOntologyConsistent() );
        }
        catch (DataFormatException e)
        {
            System.out.println ("Exception " + e);
        }
    }
}
```

Fig. 5. Example of use of DeLorean API.

4.5.2. Allowing crisp concepts and roles

According to the previous paragraph, for a fuzzy KB with n_A fuzzy atomic concepts, at most $n_A \cdot |\mathcal{N}|$ concepts are created, in addition to the axioms to preserve the semantics. Fortunately, in real applications not all concepts and roles are fuzzy.

DeLorean allows crisp concept and role declarations. If a concept (resp. role) is declared as crisp, we just need one concept (resp. role) to represent it and no new axioms. This optimization requires some manual intervention, since the ontology developer has to identify the crisp concepts and roles.

4.5.3. Ignoring superfluous elements

Theoretical algorithms to obtain a non-fuzzy equivalence of a fuzzy ontology are designed to promote reusing of the final ontology. However, for some reasoning tasks some of them are superfluous; for instance when checking the satisfiability of a fuzzy ontology.

Clearly, if some additional axioms are added the ontology, the reduction may be different and previous superfluous elements may not be superfluous any more. DeLorean ignores superfluous elements whenever possible, and guarantees that a new reduction is computed only when it is absolutely necessary.

4.5.4. Optimizing common cases

In some particular and frequent cases, the reduction of the axioms can be optimized w.r.t. the direct representation; for example, in range role axioms, domain role axioms, functional role axioms, irreflexive role axioms, or disjoint concept axioms.

4.5.5. Reducing reasoning tasks

Finally, we would like to highlight that DeLorean reuses previously computed reductions when performing a new reasoning task.

Let $\mathtt{crisp}(\mathcal{K})$ denote the non-fuzzy representation of the fuzzy KB $\mathcal{K}$, let $\mathtt{crisp}(\tau)$ denote the non-fuzzy representation of the fuzzy $\mathcal{SROIQ}(\mathbb{D})$ axiom τ, and let $\mathtt{crisp}(C, > \gamma)$ denote the strict γ-cut of C.

Consider for instance the fuzzy entailment problem. According to Section 3.7, $\mathcal{K} \models \langle a : C \bowtie \gamma \rangle$ iff $\mathcal{K}' = \mathcal{K} \cup \{\langle a : C \neg \bowtie \gamma \rangle\}$ is unsatisfiable. However, instead of checking the (un) satisfiability of $\mathcal{K}'$ (which does not make it possible to reuse the reduction of $\mathcal{K}$), we check if $\mathtt{crisp}(\mathcal{K}) \models \mathtt{crisp}(\langle a : C \bowtie \gamma \rangle)$.

In particular, DeLorean computes the supported reasoning tasks as follows:

- *Concept satisfiability*: C is satisfiable w.r.t. C iff $\mathtt{crisp}(C, > \gamma_0)$ is satisfiable w.r.t. $\mathtt{crisp}(\mathcal{K})$.
- *Entailment*: $\mathcal{K} \models \langle a : C \bowtie \gamma \rangle$ iff $\mathtt{crisp}(\mathcal{K}) \models \mathtt{crisp}(\langle a : C \bowtie \gamma \rangle)$. The case for fuzzy role assertions is similar.
- *Concept subsumption*: $\mathcal{K} \models \{\langle C \sqsubseteq_G D \geqslant 1 \rangle\}$ iff $\mathtt{crisp}(\mathcal{K}) \models \mathtt{crisp}(\langle C \sqsubseteq_G D \geqslant 1 \rangle)$,

Table 6
Complex concept construction methods.

Method	Parameters			
all	Role r	Concept c		
all	Role r	Trapezoidal d		
and	Concept c1	Concept c2		
at least	int card	Role r		
atmost	int card	Role r		
bottomConcept				
complement	Concept c			
cut	Concept c	Degree x		
goedelAll	Role r	Concept c		
goedelAll	Role r	TrapezoidalNumber t		
goedelAtmost	int card	Role r		
goedelComplement	Concept c			
goedelQatmost	int card	Role r	Concept c	
goedelQatmost	int card	Role r	TrapezoidalNumber t	
looseLowerApprox	Role r	Concept c		
looseUpperApprox	Role r	Concept c		
lowerApprox	Role r	Concept c		
nominal	Individual ind	Degree d		
or	Concept c1	Concept c2		
qatleast	int card	Role r	Concept c	
qatleast	int card	Role r	TrapezoidalNumber t	
qatmost	int card	Role r	Concept c	
qatmost	int card	Role r	TrapezoidalNumber t	
self	Role r			
some	Role r	Concept c		
some	Role r	TrapezoidalNumber t		
tightLowerApprox	Role r	Concept c		
tightUpperApprox	Role r	Concept c		
topConcept				
triangular	double a	double b	double c	Concept conc
upperApprox	Role r	Concept c		

Table 7
Axiom creation methods.

Method	Parameters				
addAsymmetricRole	Role role				
addConceptAssertion	Individual a	Concept c	Inequality i	Degree d	
addConceptEquivalence	ArrayList < Concept > a				
addCrispConcept	String conceptName				
addCrispRole	String roleName				
addDisjointConcepts	ArrayList < Concept > a				
addDisjointRoles	ArrayList < Role > a				
addDisjointUnionConcept	ArrayList < Concept > a				
addEquality	String ind1	String ind2			
addFunctionalRole	Role role				
addGoedelGCI	Concept conc1	Concept c	Inequality i	Degree d	
addGoedelNegatedRoleAss	Individual a	Role role	Individual b	Inequality i	Degree d
addGoedelRIA	ArrayList < Role > roleC	Role roleP	Inequality i	Degree d	
addGoedelRoleRange	Role role	Concept c			
addIndividual	String indName	Individual ind			
addInequality	String ind1	String ind2			
addInverseRole	Role roleName	Role invName			
addInverseRoleFunctional	Role role				
addIrreflexiveRole	Role role				
addKDGCI	Concept conc1	Concept conc2	Inequality i	Degree d	
addKDRIA	ArrayList < Role > roleC	Role roleP	Inequality i	Degree d	
addNegatedRoleAssertion	Individual a	Role role	Individual b	Inequality i	Degree d
addReflexiveRole	Role role				
addRoleAssertion	Individual a	Role role	Individual b	Inequality i	Degree d
addRoleDomain	Role role	Concept c			
addRoleEquivalence	ArrayList < Role > a				
addRoleRange	Role role	Concept c			
addSymmetricRole	Role role				
addTransitiveRole	Role role				

- *BDB*. Given a fuzzy $\mathcal{K}$ and $a{:}C$, DeLorean computes a binary search of the maximal $\gamma \in \mathcal{N}$ such that $\texttt{crisp}(\mathcal{K}) \models \texttt{crisp}(\langle a : C \geqslant \gamma \rangle)$. The case for fuzzy role assertions is similar.
- *Maximal concept satisfiability degree*. Given a fuzzy $\mathcal{K}$ and C, DeLorean computes a binary search of the maximal $\gamma \in \mathcal{N}$ such that $\texttt{crisp}(C, > \gamma)$ is satisfiable w.r.t. $\texttt{crisp}(\mathcal{K})$.

Table 8
Reasoning methods.

Type		Parameters				
boolean	entails	Individual a	Concept c	Inequality i	Degree d	
boolean	entails	Individual a	Role role	Individual b	Inequality i	Degree d
Degree	getBdb	Individual a	Concept c			
Degree	getBdb	Individual a	Role r	Individual b		
Degree	getMaxSatDegree	Concept c				
boolean	isConceptSatisfiable	Concept c				
boolean	isKBConsistent					
void	optimizedReduceKB					
void	reduceKB					
boolean	subsumes	Concept sub	Concept sup			
void	write	String file				

5. Related work

Since the first work of J. Yen in 1991 Yen (1991), an important number of fuzzy extensions to DLs can be found in the literature (Lukasiewicz & Straccia, 2008). In this section we will overview other existing fuzzy DL reasoners (in chronological order).

fuzzyDL. This may be considered the first fuzzy DL reasoner (Bobillo & Straccia, 2008). It extends fuzzy rough $\mathcal{SHIF}(\mathtt{D})$ with concept modifiers (using linear hedges and triangular functions), explicit definitions of fuzzy concepts (by means of triangular, trapezoidal, left-shoulder and right-shoulder functions), concrete features or datatypes (which can have a value with is an integer, a real or a string), some concept constructors (weighted concepts, weighted sum concepts, and threshold concepts). From a reasoning point of view, it supports both Zadeh and Łukasiewicz logics, and is able to compute several queries, ranging from typical reasoning tasks (such as entailment, BDB, concept satisfiability and subsumption problems) to variable optimization and defuzzifications. Reasoning is based on a mixture of a tableau and a MILP optimization problem. Another interesting feature is that the degrees of the fuzzy axioms may not only be numerical constants, but also variables, thus being able to deal with unknown degrees of truth. Developed in Java, it is publicly available.[14]

Fire. It implements a tableau algorithm for fuzzy $\mathcal{SHIN}$ under Zadeh logic (Stoilos, Simou, Stamou, & Kollias, 2006). It supports several reasoning tasks (consistency, entailment, Best Degree Bound, concept satisfiability, subsumption problems, and classification). An interesting feature is its graphical interface, although users need to deal directly with the syntax of the language for the representation of the fuzzy KB. Moreover, it can serialize ontologies in fuzzy $\mathcal{SHIF}$ into RDF triples, and is integrated with classical RDF storing systems, which provide persistent storing and querying over large-scale fuzzy information. Developed in Java, it is publicly available.[15]

GURDL. It supports an extension of $\mathcal{ALC}$ with an abstract and more general notion of uncertainty (Haarslev, Pai, & Shiri, 2009). The reasoning algorithm is also based on a mixture of tableau rules and the resolution of a set of inequations. Moreover, it implements some interesting optimization techniques (lexical normalization, concept simplification, partitions based on individual connectivity, caching) and studies the applicability of some techniques used in the non-fuzzy case. It is not publicly available.

GERDS. GEneralised Resolution Deductive System is a prototype implementation of a resolution algorithm for fuzzy $\mathcal{ALC}$ extended with some role constructors (role negation, top role and bottom role) under Łukasiewicz logic (Habiballa, 2007). It is publicly available.[16]

YADLR. It is a Prolog implementation of a combination of a resolution-based algorithm with linear programming for reasoning with a multi-valued DL under Łukasiewicz logic (Konstantopoulos & Apostolikas, 2007). An interesting feature is that it allows variables as degrees of truth. It is not publicly available.

KAON2. It is a DL reasoner which has been extended with the reduction of fuzzy DLs to classical DLs proposed in Straccia (2004), as mentioned in Agarwal and Hitzler (2005). An empirical study of the scalability of reasoning with fuzzy ontologies using this reasoner has been performed in Cimiano et al. (2008). The reasoner is publicly available,[17] but the fuzzy extension is not.

ONTOSEARCH2. It is the first scalable query engine for fuzzy ontologies (Pan, Thomas, & Sleeman, 2006; Thomas, Pan, & Sleeman, 2007). It implements an instance retrieval algorithm from a KB in fuzzy DL-Lite, allowing queries to be defined using a fuzzy extension of SPARQL. It is publicly available.[18]

DLMedia. It is an ontology-based multimedia information retrieval system combining logic-based retrieval with multimedia feature-based similarity retrieval (Straccia & Visco, 2008). An ontology layer may be used to define the application domain, using DLR-Lite with fuzzy concrete domains expressing similarity relations between keywords. It is publicly available.[19]

FRESG. The most recent fuzzy DL reasoner implements a tableau algorithm for fuzzy $\mathcal{ALC}(\mathtt{D})$ under Zadeh logic, with customized fuzzy data types and customized fuzzy data type predicates. It supports several reasoning tasks (consistency, entailment, concept satisfiability, retrieval, and realization). It has a simple graphical interface, but it is not publicly available (Wang, Ma, & Yin, 2009).

5.1. Discussion

fuzzyDL supports a very expressive logic, fuzzy rough $\mathcal{SROIQ}(\mathtt{D})$, that no other reasoner is able to support. Hence, *fuzzyDL* is the only that currently supports fuzzy OWL 2. Also, no other reasoner supports Gödel logic, a semantics relying on a finite chain of degrees of truth, and (with the exception of *fuzzyDL*) fuzzy rough concepts. However, most of the other reasoners also provide features that *fuzzyDL* is not able to support.

- *fuzzyDL* supports Łukasiewicz logic, new concept constructors, new fuzzy datatypes, new reasoning tasks, variables as degrees of truth, and new reasoning tasks.

[14] http://www.straccia.info/software/fuzzyDL/fuzzyDL.html.
[15] http://www.image.ece.ntua.gr/nsimou/FiRE.
[16] http://www1.osu.cz/home/habibal/page8.html.
[17] http://www.kaon2.semanticweb.org.
[18] http://www.dipper.csd.abdn.ac.uk/OntoSearch.
[19] http://faure.isti.cnr.it/straccia/software/DL-Media/DL-Media.html.

- *Fire* implements a classification algorithm and persistent storing support.
- *GURDL* supports a more general representation of uncertainty.
- *GERDS* supports additional role constructors.
- *YADLR* supports variables as degrees of truth.
- *DLMedia* and *ONTOSEARCH2* provide scalable reasoning.
- *FRESG* supports customized datatypes and new reasoning tasks.

6. Conclusions and future work

This paper has presented the fuzzy rough ontology reasoner DeLorean, the first one that supports fuzzy extensions of the languages OWL and OWL 2. DeLorean integrates translation and reasoning tasks. Given a fuzzy rough ontology in our fuzzy extension of OWL or OWL 2, DeLorean computes its non-fuzzy representation in OWL or OWL 2, respectively. Then, it uses a classical DL reasoner to perform the reasoning.

We describe the technologies used in the development, some implementations details (in particular the optimization techniques) and its functionality, with some unique features with respect to other fuzzy ontology reasoners. It supports a combination of Zadeh and Gödel fuzzy logics, allowing the ontology developer to take the best of them. It also supports a finite chain of degrees of truth, which is very useful in practice, since expert knowledge is usually expressed using linguistic terms, and since numerical interpretations of these labels can be avoided. Also, it is possible to manage vagueness in two different but complementary ways, extending the fuzzy ontology language with fuzzy rough sets.

The main direction for future work is to perform a detailed benchmark of the reasoner. On the one hand, we will compare fuzzy reasoning with DeLorean against classical reasoning. We expect to demonstrate that the overload added by the fuzzy representation is compensated for the increase of the expressiveness. On the other hand, we will compare DeLorean with other fuzzy DL reasoners.

In the former case, we will consider a real-world fuzzy ontology. As far as we know, there does not exist any significant fuzzy KB; the only one that we are aware of is a fuzzy extension of LUBM (Pan, Stamou, Stoilos, Thomas, & Taylor, 2008), but it is a non expressive ontology (in fuzzy DL Lite). We expect that fuzzy versions of more realistic ontologies will be tractable under our approach.

The latter case is complicated because different reasoners support different features and expressivities, and have different input formats. In that regard, it would be interesting to design a common fuzzy OWLlink interface.

Additionally, we plan to design and implement a tableau algorithm for a fuzzy Gödel DL, because nowadays we are not aware of any other reasoning algorithm that directly supports this kind of logics –apart from the reduction implemented by DeLorean.

We also plan to develop a graphical interface (such as a Protégé plug-in) to assist user in the development of fuzzy ontologies.

Acknowledgement

The authors have been partially funded by the Spanish Ministry of Science and Technology under project TIN2009–14538-C02–01. J. Gómez-Romero also acknowledges funding by the Comunidad de Madrid under project CAM CONTEXTS S2009/TIC-1485.

References

Agarwal, S., & Hitzler, P. (2005). sMart – A semantic matchmaking portal for electronic markets. In *Proceedings of the 7th international IEEE conference on E-commerce technology (CEC'05)*. IEEE Computer Society.

American Psychiatric Association. (1994). Diagnostic and statistical manual of mental disorders: DSM-IV-TR.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., & Patel-Schneider, P. F. (2003). *The description logic handbook: theory, implementation, and applications*. Cambridge University Press.

Bechhofer, S., Lord, P., Volz, R. (2003). Cooking the Semantic Web with the OWL API. In *Proceedings of the 2nd international semantic web conference (ISWC 2003)*.

Bechhofer, S., Möller, R., Crowther, P. (2003). The DIG Description Logic interface: DIG/1.1. In *Proceedings of the 16th international workshop on description logics (DL 2003)*.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American, 284*(5), 34–43.

Bobillo, F., Delgado, M., Gómez-Romero, J. (2007). Optimizing the crisp representation of the fuzzy Description Logic $\mathcal{SROIQ}$. In *Proceedings of the 3rd ISWC workshop on uncertainty reasoning for the semantic web (URSW 2007), CEUR workshop proceedings (Vol. 327)*.

Bobillo, F., Delgado, M., Gómez-Romero, J. (2008). DeLorean: A reasoner for fuzzy OWL 1.1. In *Proceedings of the 4th international workshop on uncertainty reasoning for the semantic web (URSW 2008), CEUR workshop proceedings (Vol. 423)*.

Bobillo, F., Delgado, M., & Gómez-Romero, J. (2009). Crisp representations and reasoning for fuzzy ontologies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 17*(4), 501–530.

Bobillo, F., Delgado, M., Gómez-Romero, J., Straccia, U. (Submitted). Joining Gödel and Zadeh fuzzy logics in fuzzy Description Logics.

Bobillo, F., Delgado, M., Gómez-Romero, J., & Straccia, U. (2009). Fuzzy Description Logics under Gödel semantics. *International Journal of Approximate Reasoning, 50*(3), 494–514.

Bobillo, F., Straccia, U. (submitted). Generalized fuzzy rough Description Logics.

Bobillo, F., & Straccia, U. (2008). fuzzyDL: An expressive fuzzy description logic reasoner. In *Proceedings of the 17th IEEE international conference on fuzzy systems (FUZZ-IEEE 2008)* (pp. 923–930). IEEE Computer Society.

Bobillo, F., & Straccia, U. (2009). Supporting fuzzy rough sets in fuzzy Description Logics. In *Proceedings of the 9th European conference on symbolic and quantitative approaches to reasoning with uncertainty (ECSQARU 2009). Lecture Notes in Computer Science* (Vol. 5590, pp. 676–687). Springer.

Calegari, S., & Sanchez, E. (2008). Object-fuzzy concept network: An enrichment of ontologies in semantic information retrieval. *Journal of the American Society for Information Science and Technology, 59*(13), 2171–2185.

Cimiano, P., Haase, P., Ji, Q., Mailis, T., Stamou, G.B., Stoilos, G., Tran, T., Tzouvaras, V. (2008). Reasoning with large A-Boxes in fuzzy Description Logics using DL reasoners: An experimental valuation. In *Proceedings of the 1st workshop on advancing reasoning on the web: scalability and commonsense (ARea2008), CEUR workshop proceedings (Vol. 350)*.

Costa, P.C.G., Laskey, K.B., Lukasiewicz, T. (2008). Uncertainty representation and reasoning in the Semantic Web. In J. Cardoso and M.D. Lytras (Eds.), *Semantic Web Engineering in the Knowledge Society*.

Cuenca-Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P. F., & Sattler, U. (2008). OWL 2: The next step for OWL. *Journal of Web Semantics, 6*(4), 309–322.

Cock, Martine De, Cornelis, Chris, & Kerre, Etienne E. (2007). Fuzzy rough sets: The forgotten step. *IEEE Transactions on Fuzzy Systems, 15*(1), 121–130.

Gruber, T. R. (1993). A translation approach to portable ontologies. *Knowledge Acquisition, 5*(2), 199–220.

Haarslev, V., & Möller, R. (2001). Racer system description. In *Proceedings of international joint conference on automated reasoning (IJCAR-01). Lecture Notes in Computer Science* (Vol. 2083, pp. 701–705). Springer-Verlag.

Haarslev, V., Pai, H.-I., & Shiri, N. (2009). A formal framework for Description Logics with uncertainty. *International Journal of Approximate Reasoning, 50*(9), 1399–1415.

Habiballa, H. (2007). solution strategies for fuzzy Description Logic. In *Proceedings of the 5th Conference of the European Society for fuzzy logic and technology (EUSFLAT 2007) (Vol. 2, pp. 27–36)*.

Horridge, M., Bechhofer, S. (2009). The OWL API: A Java API for working with OWL 2 ontologies. In *Proceedings of the 6th international workshop on OWL: experiences and directions (OWLED 2009), CEUR workshop proceedings (Vol. 529)*.

Horridge, M., Bechhofer, S., Noppens, O. (2007). Igniting the OWL 1.1 touch paper: The OWL API. In *Proceedings of the 3rd international workshop on OWL: experiences and directions (OWLED 2007), CEUR workshop proceedings (Vol. 258)*.

Horridge, M., Tsarkov, D., Redmond, T. (2006). Supporting early adoption of OWL 1.1 with Protege-OWL and FaCT++. In *Proceedings of the 2nd international workshop on OWL: experience and directions (OWLED 2006), CEUR workshop proceedings (Vol. 216)*.

Horrocks, I., Kutz, O., Sattler, U. (2006). The even more irresistible $\mathcal{SROIQ}$. In Proceedings of the 10th international conference of knowledge representation and reasoning (KR 2006) (pp. 452–457).

Horrocks, I., & Patel-Schneider, P. F. (2004). Reducing OWL entailment to Description Logic satisfiability. *Journal of Web Semantics, 1*(4), 345–357.

Hudelot, C., Atif, J., & Bloch, I. (2008). Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems, 159*(15), 1929–1951.

Konstantopoulos, S., & Apostolikas, G. (2007). Fuzzy-DL reasoning over unknown fuzzy degrees. In *Proceedings of the 3rd international workshop on semantic web and web semantics (SWWS 07), Part II. Lecture Notes in Computer Science* (Vol. 4806, pp. 1312–1318). Springer-Verlag.

Liebig, T., Luther, M., Noppens, O. (2009). The OWLlink protocol. In *Proceedings of the 6th international workshop on OWL: experiences and directions (OWLED 2009), CEUR workshop proceedings (Vol. 529)*.

Lukasiewicz, T., & Straccia, U. (2008). Managing uncertainty and vagueness in Description Logics for the semantic web. *Journal of Web Semantics, 6*(4), 291–308.

Mas, M., Monserrat, M., & Torrens, J. (2004). S-implications and r-implications on a finite chain. *Kybernetika, 40*(1), 3–20.

Mayor, G., & Torrens, J. (1993). On a class of operators for expert systems. *International Journal of Intelligent Systems, 8*(7), 771–778.

McBride, B. (2002). Jena: A semantic web toolkit. *IEEE Internet Computing, 6*(6), 55–59.

Noy, N. F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R. W., & Musen, M. A. (2001). Creating semantic web contents with Protégé-2000. *IEEE Intelligent Systems, 16*(2), 60–71.

Pan, J.Z., Stamou, G., Stoilos, G., Thomas, E., Taylor, S. (2008). Scalable querying service over fuzzy ontologies. In *Proceedings of the 17th international world wide web conference (WWW2008)*.

Pan, J.Z., Thomas, E., Sleeman, D. (2006). ONTOSEARCH2: Searching and querying web ontologies. In *Proceeedings of the IADIS international conference WWW/Internet 2006*.

Patel-Schneider, P.F., Swartout, B. (1993). Description-Logic knowledge representation system specification from the KRSS group of the ARPA knowledge sharing effort. Technical report, DARPA Knowledge Representation System Specification (KRSS) Group of the Knowledge Sharing Initiative.

Pawlak, Z. (1982). Rough sets. *International Journal of Computer and Information Sciences, 11*(5), 341–356.

Radzikowska, Anna Maria, & Kerre, Etienne E. (2002). A comparative study of fuzzy rough sets. *Fuzzy Sets and Systems, 126*(2), 137–155.

Sanchez, E. (Ed.). (2006). *Fuzzy logic and the semantic web. Capturing Intelligence* (Vol. 1). Elsevier Science.

Shearer, R., Motik, B., Horrocks, I. (2008). HermiT: A highly-efficient OWL reasoner. In *Proceedings of the 5th international workshop on OWL: experiences and directions (OWLED 2008)*.

Sirin, E., Parsia, B., Cuenca-Grau, B., Kalyanpur, A., & Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics, 5*(2), 51–53.

Stoilos, G., Simou, N., Stamou, G., & Kollias, S. (2006). Uncertainty and the Semantic Web. *IEEE Intelligent Systems, 21*(5), 84–87.

Straccia, U. (2001). Reasoning within fuzzy Description Logics. *Journal of Artificial Intelligence Research, 14*, 137–166.

Straccia, U. (2004). Transforming fuzzy Description Logics into classical description logics. In *Proceedings of the 9th european conference on logics in artificial intelligence (JELIA 2004). Lecture Notes in Computer Science* (Vol. 3229, pp. 385–399). Springer-Verlag.

Straccia, U. (2005). Description logics with fuzzy concrete domains. In *Proceedings of the 21st conference on uncertainty in artificial intelligence (UAI 2005)*. AUAI Press.

Straccia, U., Visco, G. (2008). DL-Media: An ontology mediated multimedia information retrieval system. In *Proceedings of the 4th international workshop on uncertainty reasoning for the semantic web (URSW 2008), CEUR workshop proceedings (Vol. 423)*.

Thomas, E., Pan, J.Z., Sleeman, D. (2007). ONTOSEARCH2: Searching ontologies semantically. In *Proceedings of the 3rd international workshop on OWL: experiences and directions (OWLED 2007), CEUR workshop proceedings (Vol. 258)*.

Tsarkov, D., Horrocks, I. (2006). FaCT++ Description Logic reasoner: System description. In *Proceedings of the 3rd international joint conference on automated reasoning (IJCAR 2006)*.

Turhan, A.Y., Bechhofer, S., Kaplunova, A., Liebig, T., Luther, M., Möller, R., Noppens, O., Patel-Schneider, P., Suntisrivaraporn, B., and Weithöner, T. (2006). DIG 2.0 - Towards a flexible interface for Description Logic reasoners. In *Proceedings of the 2nd international workshop on OWL: experience and directions (OWLED 2006), CEUR workshop proceedings (Vol. 216)*.

W3C. OWL Web Ontology Language overview, 2004. Available: http://www.w3.org/TR/owl-features.

Wang, H., Ma, Z. M., & Yin, J. (2009). Fresg: A kind of fuzzy description logic reasoner. In *Proceedings of the 20th international conference on database and expert systems applications (DEXA 2009)* (Vol. 5690, pp. 443–450). Lecture notes in computer science. Springer-Verlag.

Yen, J. (1991). Generalizing term subsumption languages to fuzzy logic. In *Proceedings of the 12th international joint conference on artificial intelligence (IJCAI 1991)* (pp. 472–477). Morgan Kaufman.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control, 8*, 338–353.