Reasoning in Fuzzy OWL 2 with DeLorean

Fernando Bobillo[1], Miguel Delgado[2], and Juan Gómez-Romero[3]

[1] Dpt. of Computer Science and Systems Engineering, University of Zaragoza, Spain
[2] Dpt. of Computer Science and Artificial Intelligence, University of Granada, Spain
[3] Applied Artificial Intelligence Group, University Carlos III, Madrid, Spain
`fbobillo@unizar.es, mdelgado@ugr.es, jgromero@inf.uc3m.es`

Abstract. Classical ontologies are not suitable to represent imprecise or vague information, which has led to several extensions using non-classical logics. In particular, several fuzzy extensions have been proposed in the literature. In this paper, we present the fuzzy ontology reasoner DeLorean, the first to support a fuzzy extension of OWL 2. We discuss how to use it for fuzzy ontology representation and reasoning, and describe some implementation details and optimization techniques. An empirical evaluation demonstrates that these optimizations considerably improve the performance of the reasoner.

1 Introduction

Ontologies have been successfully used as a formalism for knowledge representation in several applications. In particular, they are a core element in the layered architecture of the Semantic Web. In that regard, the language OWL 2 [1] has very recently become a W3C Recommendation for ontology representation.

Description Logics (DLs for short) [2] 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. They have proved to be very useful as ontology languages, in such a way that OWL 2 is closely equivalent to the DL $\mathcal{SROIQ}(\mathbf{D})$ [3].

Today, there is a growing interest in the development of knowledge representation formalisms able to deal with imprecise knowledge, a very common requirement in real world applications. Nevertheless, classical ontologies are not appropriate to deal with imprecision and vagueness in the knowledge, which is inherent to most real world application domains. Since fuzzy logic is a suitable formalism to handle these types of knowledge, several fuzzy extensions of DLs have been proposed [4].

The apparition of the new standard language OWL 2 has motivated a need to extend ontology editors, reasoners, and other supporting tools. The situation is similar in the fuzzy case, and having reasoners that are able to support fuzzy extensions of OWL 2 is of great importance. In this paper we report the implementation of DeLorean (DEscription LOgic REasoner with vAgueNess)[1], the first reasoner that supports the fuzzy DL $\mathcal{SROIQ}(\mathbf{D})$, and hence fuzzy OWL 2.

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

In a strict sense, DeLorean is not a reasoner but a *translator* from a fuzzy ontology language into a classical (i.e., non-fuzzy) ontology language –namely, the standard language OWL 2. The non-fuzzy ontology resulting from this translation, which preserves the semantics of the initial fuzzy representation, is afterwards processed by an integrated classical DL reasoner. According to this ability of combining the reduction procedure with the classical DL reasoning, we will simply refer to it as a reasoner.

This paper is organized as follows. Section 2 provides some background on fuzzy set theory and fuzzy logic. Then, Section 3 describes which fuzzy ontologies can be managed by the system and how they are represented. Next, Section 4 explains which reasoning tasks can be performed by using the reasoner and how they are accomplished. In Section 5, we give some implementation details. A use case and a preliminary evaluation of the implemented optimizations are discussed in Section 6. Finally, Section 7 sets out some conclusions and prospective directions for future work.

2 Fuzzy Logic

Fuzzy set theory and fuzzy logic were proposed by L. Zadeh [13] 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 an element of X.

Changing the usual true/false convention leads to a new type of propositions, called *fuzzy propositions*. Each fuzzy proposition may have a *degree of truth* in $[0, 1]$, denoting the compatibility of the fuzzy proposition with a given state of facts. For example, the truth of the proposition stating than a given tomato is a ripe tomato is clearly a matter of degree.

In this article we will consider *fuzzy formulae* (or fuzzy axioms) of the form $\phi \geq \alpha$ or $\phi \leq \beta$, where ϕ is a fuzzy proposition and $\alpha, \beta \in [0, 1]$ [14]. This imposes that the degree of truth of ϕ is *at least* α (resp. *at most* β). For example, $\mathsf{x\ is\ a\ ripe\ tomato} \geq 0.9$ says that we have a rather ripe tomato (the degree of truth of x being a ripe tomato is at least 0.9).

All classical set operations are extended to fuzzy sets. The intersection, union, complement and implication set operations are performed by corresponding functions: a t-norm, a t-conorm, a negation, and an implication, respectively. The combination of them is called a fuzzy logic.

There are three main fuzzy logics: Łukasiewicz, Gödel, and Product. The importance of these three fuzzy logics is due to the fact that any continuous t-norm can be obtained as a combination of Łukasiewicz, Gödel, and Product t-norms. It is also common to consider the fuzzy connectives originally considered by Zadeh (Gödel conjunction and disjunction, Łukasiewicz negation and Kleene-Dienes

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	$\min\{\alpha, \beta\}$	$\max\{\alpha, \beta\}$	$\begin{cases} 1, \ \alpha = 0 \\ 0, \ \alpha > 0 \end{cases}$	$\begin{cases} 1 \quad \alpha \leq \beta \\ \beta, \ \alpha > \beta \end{cases}$
Łukasiewicz	$\max\{\alpha + \beta - 1, 0\}$	$\min\{\alpha + \beta, 1\}$	$1 - \alpha$	$\min\{1 - \alpha + \beta, 1\}$
Product	$\alpha \cdot \beta$	$\alpha + \beta - \alpha \cdot \beta$	$\begin{cases} 1, \ \alpha = 0 \\ 0, \ \alpha > 0 \end{cases}$	$\begin{cases} 1 \qquad \alpha \leq \beta \\ \beta/\alpha, \ \alpha > \beta \end{cases}$

implication), which is known as Zadeh fuzzy logic. Table 1 shows these four fuzzy logics: Zadeh, Łukasiewicz, Gödel, and Product.

For every $\alpha \in [0, 1]$, 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. $\{x \mid \mu_A(x) \geq \alpha\}$.

Relations can also be extended to the fuzzy case. A (binary) *fuzzy relation* R over two countable sets X and Y is a function $R\colon X \times Y \rightarrow [0, 1]$. Several properties of the relations (such as reflexive, irreflexive, symmetric, asymmetric, transitive, or disjointness) and operations (inverse, composition) can be trivially extended to the fuzzy case.

3 Representing Fuzzy Ontologies

In this section we discuss the fuzzy ontologies that DeLorean is able to manage. Section 3.1 describes the elements of the supported fuzzy DL $\mathcal{SROIQ}(\mathbf{D})$. Section 3.2 discusses how to create these fuzzy ontologies.

3.1 Fuzzy $\mathcal{SROIQ}(\mathbf{D})$

Fuzzy $\mathcal{SROIQ}(\mathbf{D})$ [15,16], extends $\mathcal{SROIQ}(\mathbf{D})$ to the fuzzy case by letting concepts denote fuzzy sets of individuals and roles denote fuzzy binary relations.

Notation. In the following, we will use $\otimes$ for denoting a t-norm, $\oplus$ for a t-conorm, $\ominus$ for a negation, an $\Rightarrow$ for an implication. The subscript $_Z$ denotes Zadeh fuzzy logic, and $_G$ denotes Gödel fuzzy logic.

We will assume that the *degrees of truth* are rational numbers of the form $\alpha \in (0, 1]$, $\beta \in [0, 1)$ and $\gamma \in [0, 1]$. Moreover, we will assume a set of *inequalities* $\bowtie \in \{\geq, >, \leq, <\}$, $\rhd \in \{\geq, >\}$, $\lhd \in \{\leq, <\}$.

Fuzzy $\mathcal{SROIQ}(\mathbf{D})$ has three *alphabets* of symbols for concepts, roles and individuals.

The *concepts* of the language are denoted as C, D (if they are complex concepts) and A (if atomic). Some complex concepts will use natural numbers n, m such that $n \geq 0, m > 0$.

The *roles* can be abstract (denoted R) or concrete (denoted T). R_A denotes an atomic abstract role, R^- the inverse role, S a simple role[2], and U the universal role (a relation which is true for every pair of individuals).

[2] Simple roles are needed to guarantee the decidability of the logic. Intuitively, simple roles cannot take part in cyclic role inclusion axioms (see [15] for a formal definition).

A *fuzzy concrete domain* (also called a fuzzy *datatype*) $\mathbf{D}$ is a pair $\langle \Delta_{\mathbf{D}}, \Phi_{\mathbf{D}} \rangle$, where $\Delta_{\mathbf{D}}$ is a concrete interpretation domain (for instance, the set of rational numbers in a given interval), and $\Phi_{\mathbf{D}}$ is a set of fuzzy concrete predicates $\mathbf{d}$ [18]. Typical examples of fuzzy concrete predicates are the *trapezoidal*, the *triangular*, the *left-shoulder*, and the *right-shoulder* membership functions. We will restrict to *trapezoidal* membership functions, which are more general than the other mentioned predicates.

The *individuals* are denoted as a, b (if abstract), and v (if concrete). Abstract individuals are elements of the interpretation domain, whereas concrete individuals are instances of the concrete interpretation domain.

Syntax. The syntax of fuzzy concepts, roles, and axioms is shown in Table 2. A *fuzzy Knowledge Base* (KB) is a finite set of fuzzy axioms, which can be grouped into a fuzzy ABox with axioms (A1)–(A7), a fuzzy TBox with axioms (A8)–(A9), and a fuzzy RBox with axioms (A10)–(A20). Note that the only syntactic differences with respect to crisp $\mathcal{SROIQ}(\mathbf{D})$ are (C11), (A1)–(A5), (A8), (A10)–(A11).

Most axioms are better known by a name. (A1) are named concept assertions, (A2)–(A5) are role assertions, (A6) are inequality assertions, (A7) are equality assertions, (A8) are General Concept Inclusions (GCIs, or subclass axioms), (A9) are concept equivalences, (A10)–(A11) are Role Inclusion Axioms (RIAs, or sub-role axioms), (A12)–(A13) are role equivalences, (A14) are transitive role axioms, (A15)–(16) are disjoint role axioms, (A17) are reflexive role axioms, (A18) are irreflexive role axioms, (A19) are symmetric role axioms, and (A20) are asymmetric role axioms.

As in the crisp case, GCIs can be used to express some interesting axioms, such as *disjointness* of concepts or *domain*, *range* and *functionality* of a role [17].

Example 1. The fuzzy concept assertion $\langle \mathsf{RoseDAnjou} : \mathsf{RoseWine} \geq 0.75 \rangle$ states that it is almost true that the Rosé D'Anjou wine is a rose wine. $\mathsf{RoseDAnjou}$ is an abstract individual, and $\mathsf{RoseWine}$ is a fuzzy concept. $\square$

Semantics. The semantics of the logic is given using the notion of fuzzy interpretation. 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:

– An *abstract individual* a to an element $a^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$.
– A *concrete individual* v to an element $v_{\mathbf{D}} \subseteq \Delta_{\mathbf{D}}$.
– A fuzzy *concept* C to a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0, 1]$.
– A fuzzy *abstract role* R to a function $R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0, 1]$.
– A fuzzy *concrete role* T to a function $T^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta_{\mathbf{D}} \to [0, 1]$.
– An n-ary fuzzy *concrete predicate* $\mathbf{d}$ to a function $\mathbf{d}^{\mathcal{I}} : \Delta_{\mathbf{D}}^{n} \to [0, 1]$.

$C^{\mathcal{I}}$ denotes the membership function of the fuzzy concept C w.r.t. $\mathcal{I}$. $C^{\mathcal{I}}(a^{\mathcal{I}})$ denotes to what extent the individual a can be considered an element of the fuzzy concept C. $R^{\mathcal{I}}$ denotes the membership function of the fuzzy role R w.r.t.

Table 2. Syntax and semantics of the fuzzy DL $\mathcal{SROIQ}(\mathbf{D})$

Concept	**Syntax** (C)	**Semantics of** $C^{\mathcal{I}}(x)$
(C1)	A	$A^{\mathcal{I}}(x)$
(C2)	$\top$	1
(C3)	$\bot$	0
(C4)	$C \sqcap D$	$C^{\mathcal{I}}(x) \otimes D^{\mathcal{I}}(x)$
(C5)	$C \sqcup D$	$C^{\mathcal{I}}(x) \oplus D^{\mathcal{I}}(x)$
(C6)	$\neg C$	$\ominus C^{\mathcal{I}}(x)$
(C7)	$\forall R.C$	$\inf_{y \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)\}$
(C8)	$\exists R.C$	$\sup_{y \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)\}$
(C9)	$\forall T.\mathbf{d}$	$\inf_{v \in \Delta_{\mathbf{D}}} \{T^{\mathcal{I}}(x,v) \Rightarrow \mathbf{d}^{\mathcal{I}}(v)\}$
(C10)	$\exists T.\mathbf{d}$	$\sup_{v \in \Delta_{\mathbf{D}}} \{T^{\mathcal{I}}(x,v) \otimes \mathbf{d}^{\mathcal{I}}(v)\}$
(C11)	$\{\alpha/a\}$	α if $x = a^{\mathcal{I}}$, 0 otherwise
(C12)	$\geq m\ S.C$	$\sup_{y_1,\ldots,y_m \in \Delta^{\mathcal{I}}} \{\min_{i=1}^{m}\{S^{\mathcal{I}}(x,y_i) \otimes C^{\mathcal{I}}(y_i)\} \bigotimes (\otimes_{1 \leq j < k \leq m}\{y_j \neq y_k\})\}$
(C13)	$\leq n\ S.C$	$\inf_{y_1,\ldots,y_{n+1} \in \Delta^{\mathcal{I}}} \{\min_{i=1}^{n+1}\{S^{\mathcal{I}}(x,y_i) \otimes C^{\mathcal{I}}(y_i)\} \Rightarrow \oplus_{1 \leq j < k \leq n+1}\{y_j = y_k\}\}$
(C14)	$\geq m\ T.\mathbf{d}$	$\sup_{v_1,\ldots,v_m \in \Delta_{\mathbf{D}}} \{\min_{i=1}^{m}\{T^{\mathcal{I}}(x,v_i) \otimes \mathbf{d}^{\mathcal{I}}(v_i)\} \bigotimes (\otimes_{j<k}\{v_j \neq v_k\})\{$
(C15)	$\leq n\ T.\mathbf{d}$	$\inf_{v_1,\ldots,v_{n+1} \in \Delta_{\mathbf{D}}} \{\min_{i=1}^{n+1}\{T^{\mathcal{I}}(x,v_i) \otimes \mathbf{d}^{\mathcal{I}}(v_i)\} \Rightarrow (\oplus_{j<k}\{v_j = v_k\})\}$
(C16)	$\exists S.\mathtt{Self}$	$S^{\mathcal{I}}(x,x)$
Role	**Syntax** (R)	**Semantics of** $R^{\mathcal{I}}(x,y)$
(R1)	R_A	$R_A^{\mathcal{I}}(x,y)$
(R2)	R^-	$R^{\mathcal{I}}(y,x)$
(R3)	U	1
(R4)	T	$T^{\mathcal{I}}(x,y)$
Axiom	**Syntax** (τ)	**Semantics** ($\mathcal{I}$ satisfies τ if ...)
(A1)	$\langle a : C \bowtie \alpha \rangle$	$C^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie \alpha$
(A2)	$\langle (a,b) : R \bowtie \alpha \rangle$	$R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \bowtie \alpha$
(A3)	$\langle (a,b) : \neg R \bowtie \alpha \rangle$	$\ominus R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \bowtie \alpha$
(A4)	$\langle (a,v) : T \bowtie \alpha \rangle$	$T^{\mathcal{I}}(a^{\mathcal{I}}, v_{\mathbf{D}}) \bowtie \alpha$
(A5)	$\langle (a,v) : \neg T \bowtie \alpha \rangle$	$\ominus T^{\mathcal{I}}(a^{\mathcal{I}}, v_{\mathbf{D}}) \bowtie \alpha$
(A6)	$\langle a \neq b \rangle$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$
(A7)	$\langle a = b \rangle$	$a^{\mathcal{I}} = b^{\mathcal{I}}$
(A8)	$\langle C \sqsubseteq D \rhd \alpha \rangle$	$\inf_{x \in \Delta^{\mathcal{I}}} \{C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x)\} \rhd \alpha$
(A9)	$C_1 \equiv \cdots \equiv C_m$	$\forall_{x \in \Delta^{\mathcal{I}}} C_1^{\mathcal{I}}(x) = \cdots = C_m^{\mathcal{I}}(x)$
(A10)	$\langle R_1 \ldots R_m \sqsubseteq R \rhd \alpha \rangle$	$\inf_{x_1,x_{m+1} \in \Delta^{\mathcal{I}}} \{\sup_{x_2 \ldots x_m \in \Delta^{\mathcal{I}}} \{(R_1^{\mathcal{I}}(x_1,x_2) \otimes \cdots \otimes R_m^{\mathcal{I}}(x_m, x_{m+1})) \Rightarrow R^{\mathcal{I}}(x_1, x_{m+1})\}\} \rhd \alpha$
(A11)	$\langle T_1 \sqsubseteq T_2 \rhd \alpha \rangle$	$\inf_{x \in \Delta^{\mathcal{I}}, v \in \Delta_{\mathbf{D}}} \{T_1^{\mathcal{I}}(x,v) \Rightarrow T_2^{\mathcal{I}}(x,v)\} \rhd \alpha$
(A12)	$R_1 \equiv \ldots R_m$	$\forall_{x,y \in \Delta^{\mathcal{I}}} R_1^{\mathcal{I}}(x,y) = \cdots = R_m^{\mathcal{I}}(x,y)$
(A13)	$T_1 \equiv \ldots T_m$	$\forall_{x \in \Delta^{\mathcal{I}}, v \in \Delta_{\mathbf{D}}} R_1^{\mathcal{I}}(x,v) = \cdots = R_m^{\mathcal{I}}(x,v)$
(A14)	$\mathtt{trans}(R)$	$\forall x,y,z \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(x,z) \otimes R^{\mathcal{I}}(z,y) \leq R^{\mathcal{I}}(x,y)$
(A15)	$\mathtt{dis}(S_1,\ldots,S_m)$	$\forall x,y \in \Delta^{\mathcal{I}}, \min\{S_i^{\mathcal{I}}(x,y), S_j^{\mathcal{I}}(x,y)\} = 0, \forall 1 \leq i < j \leq m$
(A16)	$\mathtt{dis}(T_1,\ldots,T_m)$	$\forall x \in \Delta^{\mathcal{I}}, v \in \Delta_{\mathbf{D}}, \min\{T_i^{\mathcal{I}}(x,v), T_j^{\mathcal{I}}(x,v)\} = 0, \forall 1 \leq i < j \leq m$
(A17)	$\mathtt{ref}(R)$	$\forall x \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(x,x) = 1$
(A18)	$\mathtt{irr}(S)$	$\forall x \in \Delta^{\mathcal{I}}, S^{\mathcal{I}}(x,x) = 0$
(A19)	$\mathtt{sym}(R)$	$\forall x,y \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(x,y) = R^{\mathcal{I}}(y,x)$
(A20)	$\mathtt{asy}(S)$	$\forall x,y \in \Delta^{\mathcal{I}}$, $S^{\mathcal{I}}(x,y) > 0$ then $S^{\mathcal{I}}(y,x) = 0$

$\mathcal{I}$. $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}})$ denotes to what extent (a, b) can be considered an element of the fuzzy role R.

Given the operators $\otimes, \oplus, \ominus, \Rightarrow$, the fuzzy interpretation function is defined for fuzzy concepts, roles, concrete domains and axioms as shown in Table 2. The fuzzy DL $Z\ \mathcal{SROIQ}(\mathbf{D})$ uses the operators in Zadeh fuzzy logic, whereas $G\ \mathcal{SROIQ}(\mathbf{D})$ uses the operators in Gödel fuzzy logic.

We say that a fuzzy interpretation $\mathcal{I}$ satisfies a fuzzy KB $\mathcal{K}$ iff $\mathcal{I}$ satisfies each element in $\mathcal{K}$.

3.2 A Fuzzy Ontology Editor

The input fuzzy ontologies supported by DeLorean can be created in 3 ways:

- Encoding the fuzzy ontology in the specific language of the reasoner (referred in this paper as "DeLorean syntax"). The details of DeLorean syntax can be found in the web page of the reasoner.
- Creating programmatically a new ontology by using the DeLorean API. The DeLorean API is a Java library that allows fuzzy ontology management (by loading existing ontologies or by creating and populating them) and solving reasoning tasks. The Javadoc documentation of the API can be found along with the distribution.
- Using a fuzzy ontology editor tool and then translating the ontology into DeLorean syntax.

In this section, we focus on the third option, which is the recommended one, since it allows us to edit fuzzy ontologies in a more abstract way.

A methodology for fuzzy ontology representation using OWL 2 has been recently proposed [26]. The key idea of this representation is to use an OWL 2 ontology and extend their elements with annotations representing the features of the fuzzy ontology that OWL 2 cannot directly encode. In order to separate the annotations including fuzzy information from other annotations, a new annotation property called `fuzzyLabel` is used, and every annotation is identified by the tag `fuzzyOwl2`.

Example 2. The fuzzy concept assertion of Example 1 is represented by annotating the axiom with the degree ≥ 0.75 as follows:

```
<ClassAssertion>
   <Class IRI='#RoseWine'/>
   <NamedIndividual IRI='#RoseDAnjou'/>
   <Annotation>
      <AnnotationProperty IRI='#fuzzyLabel'/>
      <Literal datatypeIRI='&rdf;PlainLiteral'>
         <fuzzyOwl2 fuzzyType="axiom">
            <Degree value="0.75" />
         </fuzzyOwl2>
      </Literal>
   </Annotation>
</ClassAssertion>
```

□

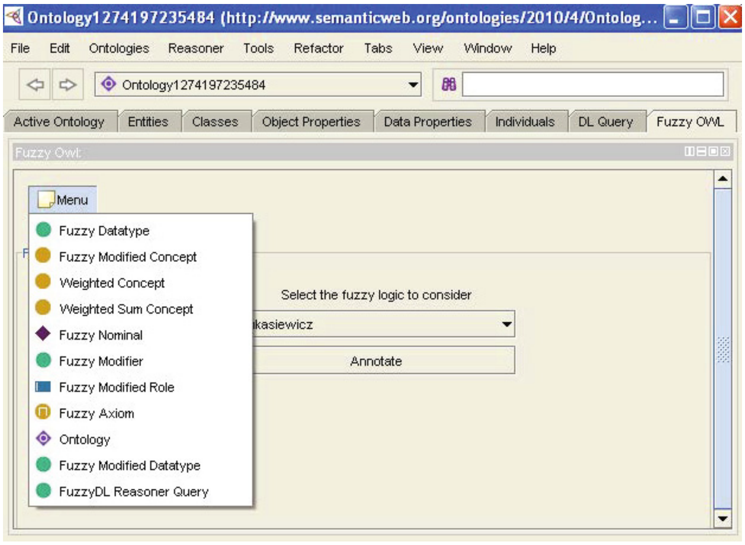

Ontology1274197235484 (http://www.semanticweb.org/ontologies/2010/4/Ontolog...
File Edit Ontologies Reasoner Tools Refactor Tabs View Window Help
Ontology1274197235484
Active Ontology Entities Classes Object Properties Data Properties Individuals DL Query Fuzzy OWL
Fuzzy Owl:
Menu
Fuzzy Datatype
Fuzzy Modified Concept
Weighted Concept
Weighted Sum Concept
Fuzzy Nominal
Fuzzy Modifier
Fuzzy Modified Role
Fuzzy Axiom
Ontology
Fuzzy Modified Datatype
FuzzyDL Reasoner Query
Select the fuzzy logic to consider
kasiewicz
Annotate

Fig. 1. Menu options of the plug-in

Since typing such annotations is a tedious and error-prone task, a Protégé plug-in has been implemented to make the syntax of the annotations transparent to the users. The plug-in is freely available[3]. Once it is installed, a new tab named `Fuzzy OWL` enables to use the plug-in. Figure 1 shows the available options.

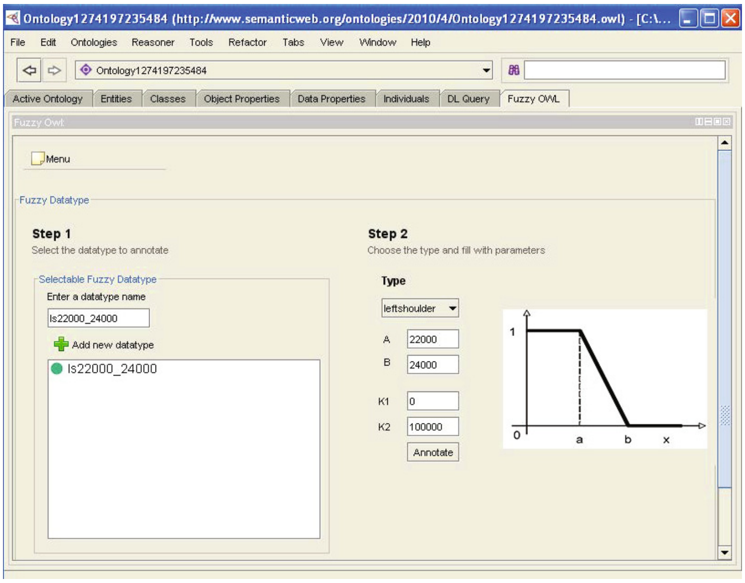
It is important to remark that the plug-in is generic and not specific to our reasoner, so it offers the possibility of adding other elements that are not yet supported by DeLorean; e.g., weighted concepts, weighted sum concepts, fuzzy modified roles, fuzzy modified datatypes, Łukasiewicz fuzzy logic ...

Firstly, the user can create the non-fuzzy part of the ontology using the editor as usual. Then, the user can define the fuzzy elements of the ontology by using the plug-in; namely, fuzzy axioms, fuzzy datatypes, fuzzy modifiers, fuzzy modified concepts, and fuzzy nominals.

Figure 2 illustrates the plug-in use by showing how to create a new fuzzy datatype. The user specifies the name of the datatype, and the type of the membership function. Then, the plug-in asks for the necessary parameters according to the type. A picture is displayed to help the user recall the meaning of the parameters. Then, after some error checks, the new datatype is created and can be used in the ontology.

Once the fuzzy ontology has been created with `Fuzzy OWL`, it has to be translated into the language supported by a specific fuzzy DL reasoner –DeLorean, in this case– to allow reasoning with it. For instance, the datatype created in Figure 2 would be represented by using a trapezoidal function in DeLorean syntax.

[3] `http://webdiis.unizar.es/~fbobillo/fuzzyOWL2`



Ontology1274197235484 (http://www.semanticweb.org/ontologies/2010/4/Ontology1274197235484.owl) - [C:\...

Ontology1274197235484

Fuzzy Owl

Fuzzy Datatype

Step 1
Select the datatype to annotate

Selectable Fuzzy Datatype
Enter a datatype name: ls22000_24000

● ls22000_24000

Step 2
Choose the type and fill with parameters

Type: leftshoulder
A: 22000
B: 24000
K1: 0
K2: 100000

Fig. 2. Creation of a fuzzy datatype

For this purpose, the plug-in includes a general parser that can be customized to any reasoner by adapting a template code. The parser browses the contents of the ontology –with *OWL API 3*[4] [21], which allows iterating over the elements of the ontology in a transparent way– and prints an informative message. The output of the process is a fuzzy ontology that can be printed in the standard output or saved in a text file. If the user selects one of these elements, they will be discarded when translated into DeLorean syntax, and an informative message will be displayed to the users.

The template code has been adapted to build two parsers, one for `fuzzyDL`, and one for `DeLorean`. Both the template and the parsers can be freely obtained from the plug-in web page. Furthermore, similar parsers for other fuzzy DL reasoners can be easily obtained. To do so, we can replace the default messages by well-formed axioms, according to the desired fuzzy ontology syntax.

4 Reasoning with DeLorean

In this section we discuss how to reason with the fuzzy ontologies by using DeLorean. Firstly, Section 4.1 describes the supported reasoning tasks. Then, Section 4.2 explains how to interact with the application.

4.1 Reasoning Tasks

There are several reasoning tasks in fuzzy $\mathcal{SROIQ}(\mathbf{D})$ [19]. We will focus on the following ones:

[4] `http://owlapi.sourceforge.net`

- *Fuzzy KB satisfiability.* A fuzzy interpretation $\mathcal{I}$ *satisfies* (is a model of) a fuzzy KB $\mathcal{K}$ iff it satisfies each axiom in $\mathcal{K}$.
- *Concept satisfiability.* C is α-satisfiable w.r.t. a fuzzy KB $\mathcal{K}$ iff there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $C^{\mathcal{I}}(x) \geq \alpha$ for some $x \in \Delta^{\mathcal{I}}$.
- *Entailment*: An axiom τ of the forms (A1)–(A5) is entailed by a fuzzy KB $\mathcal{K}$ iff every model of $\mathcal{K}$ satisfies τ.
- *Concept subsumption*: D subsumes C (denoted $C \sqsubseteq D$) w.r.t. a fuzzy KB $\mathcal{K}$ iff every model $\mathcal{I}$ of $\mathcal{K}$ satisfies $\forall x \in \Delta^{\mathcal{I}}, C^{\mathcal{I}}(x) \leq D^{\mathcal{I}}(x)$.
- *Best degree bound* (BDB) of an axiom τ of the forms (A1)–(A5) is defined as the $\sup\{\alpha : \mathcal{K} \models \langle \tau \geq \alpha \rangle\}$.

DeLorean reasoning algorithms are based on the computation of a crisp ontology that preserves the semantics of the qoriginal fuzzy ontology, and therefore reasoning with the former is equivalent to reasoning with the latter. This kind of reduction has already been considered in the literature (see for instance [15] for Zadeh fuzzy DLs and [16] for Gödel fuzzy DLs).

The equivalent crisp ontology has a larger size than the original fuzzy ontology, because some axioms must be added to keep the same semantics. If we assume a fixed set of degrees of truth, the size of the equivalent crisp ontology depends linearly on the size of the fuzzy ontology.

An interesting property is that the computation of the equivalent crisp ontology can be reused when adding a new axiom. If the new axiom does not introduce new atomic concepts, atomic roles, nor a new degree of truth, we just need to add the reduction of the new axiom.

Example 3. Assume a set of degrees of truth $\mathcal{N} = \{0, 0.25, 0.5, 0.75, 1\}$. Consider the fuzzy KB $\mathcal{K} = \{\langle \mathsf{RoseDAnjou} : \neg\mathsf{RedWine} \geq 0.5 \rangle\}$, stating that it is almost true that Rosé D'Anjou is not a red wine. Let us show how to compute the crisp representation of $\mathcal{K}$ in Zadeh fuzzy logic.

To start with, we create 8 new crisp atomic concepts: $\mathsf{RoseDAnjou}_{\geq 0.25}$, $\mathsf{RedWine}_{\geq 0.25}$, $\mathsf{RoseDAnjou}_{\geq 0.5}$, $\mathsf{RedWine}_{\geq 0.5}$, $\mathsf{RoseDAnjou}_{\geq 0.75}$, $\mathsf{RedWine}_{\geq 0.75}$, $\mathsf{RoseDAnjou}_{\geq 1}$, $\mathsf{RedWine}_{\geq 1}$.

Next, we add some axioms keeping the semantics of these new concepts: $\mathsf{RoseDAnjou}_{\geq 0.5} \sqsubseteq \mathsf{RoseDAnjou}_{\geq 0.25}$, $\mathsf{RoseDAnjou}_{\geq 0.75} \sqsubseteq \mathsf{RoseDAnjou}_{\geq 0.5}$, $\mathsf{RoseDAnjou}_{\geq 1} \sqsubseteq \mathsf{RoseDAnjou}_{\geq 0.75}$, $\mathsf{RedWine}_{\geq 0.5} \sqsubseteq \mathsf{RedWine}_{\geq 0.25}$, $\mathsf{RedWine}_{\geq 0.75} \sqsubseteq \mathsf{RedWine}_{\geq 0.5}$, $\mathsf{RedWine}_{\geq 1} \sqsubseteq \mathsf{RedWine}_{\geq 0.75}$.

Finally, we represent the axiom in $\mathcal{K}$ as $\mathsf{RoseDAnjou} : \neg\mathsf{RedWine}_{\geq 0.75}$.

Now, we shall discuss the case of Gödel fuzzy logic. The procedure is very similar, but the representation of the axioms in the fuzzy ontology is different, as it must take into account the different semantics of the fuzzy logical operators. In particular, the axiom would be represented as $\mathsf{RoseDAnjou} : \neg\mathsf{RedWine}_{\geq 0.25}$.

□

4.2 Using DeLorean

DeLorean can be used as a stand-alone application. In addition, DeLorean reasoning services can also be used from other programs by means of the DeLorean API. For details about the API, we refer the reader to the Javadoc

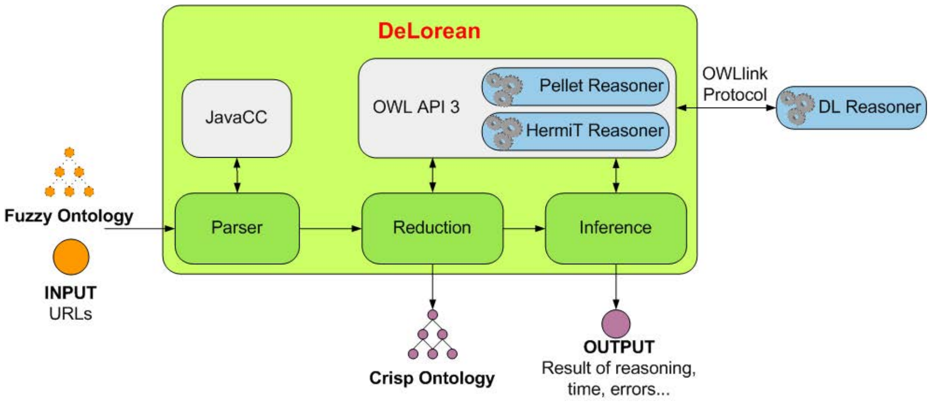


Fig. 3. Architecture of DeLorean reasoner

documentation of the package. In this section, we will focus on the user of the reasoner through its graphical interface.

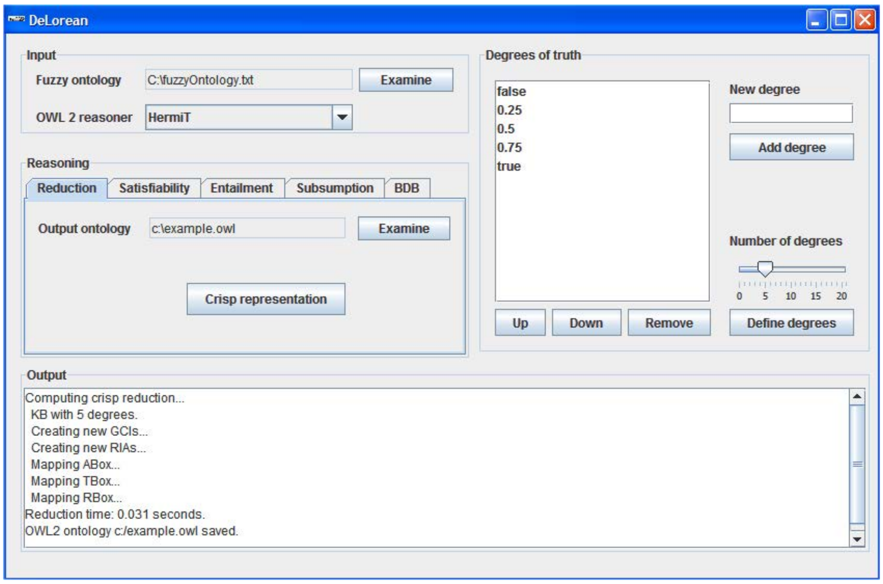
Figure 3 illustrates the architecture of the system:

– The *Parser* reads an input file with a fuzzy ontology in DeLorean syntax and translates it into an internal representation[5]. It is important to remark that we can use any language as long as there is a parser that can obtain an internal representation. Also, we could have several parsers to support different input languages.
– The *Reduction* module implements the reduction procedures described in the previous section. It builds an OWL API model representing an equivalent crisp ontology that can be exported to an OWL 2 file. The implementation also takes into account all the optimizations already discussed along this document.
– The *Inference* module communicates with a non-fuzzy reasoner (either one of the integrated reasoners or a reasoner via the OWLlink [25][6] protocol) in order to perform the reasoning tasks.
– A simple *User interface* manages inputs and outputs (see details below).

Figure 4 shows a snapshot of the user interface, structured in 4 sections:

Input. Here, the user can specify the input fuzzy ontology and the DL reasoner that will be used in the reasoning. The possible choices are *HermiT* [24][7], *Pellet* [22][8], and an 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.

[5] This parser should not be confused with the translator from the Protégé plug-in into DeLorean syntax discussed in Section 3.2.
[6] `http://www.owllink.org`
[7] `http://hermit-reasoner.com`
[8] `http://clarkparsia.com/pellet`

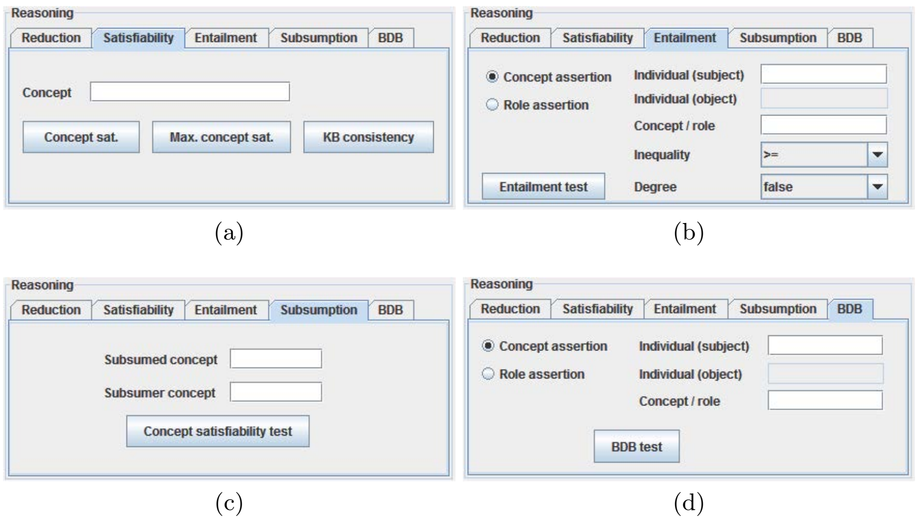


Fuzzy ontology: C:\fuzzyOntology.txt
OWL 2 reasoner: HermiT
Output ontology: c:\example.owl

```
false
0.25
0.5
0.75
true
```

```
Computing crisp reduction...
 KB with 5 degrees.
 Creating new GCIs...
 Creating new RIAs...
 Mapping ABox...
 Mapping TBox...
 Mapping RBox...
Reduction time: 0.031 seconds.
OWL2 ontology c:/example.owl saved.
```

Fig. 4. User interface of DeLorean reasoner



(b) [● Concept assertion] ○ Role assertion; Inequality: >=; Degree: false

(d) [● Concept assertion] ○ Role assertion

Fig. 5. (a) Concept satisfiability and KB consistency; (b) Concept and role entailment; (c) Concept subsumption; (d) BDB

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's convenience, it is possible to directly specify a number of degrees of truth, and they will be automatically generated.

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 process.

Reasoning. This part is used to perform the different reasoning tasks that DeLorean supports. The panel 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 3.1).

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 Figure 4.

Satisfiability. In this tab (see Figure 5 (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 Figure 5 (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 Figure 5 (c)), after specifying the names of the subsumed and the subsumer fuzzy concepts, it is possible to compute the *fuzzy concept subsumption*.

BDB. Finally, in the fifth tab (see Figure 5 (d)), the user can compute the BDB of a fuzzy concept assertion or a fuzzy role assertion. As in the case of entailment, the user has to specify previously 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.

5 Implementation Details

In this section we briefly explain some implementation details of DeLorean. In Section 5.1 we run through the different versions of the reasoner and comment their main differences. Then, Section 5.2 summarizes some optimization techniques that are implemented in order to make the reasoning more efficient.

5.1 Some Historical Notes

The first version of the reasoner was based on Jena API[9]. It was developed in Java by relying on the parser generator JavaCC[10] and DIG 1.1 interface [20] (to communicate with crisp DL reasoners). The use of DIG limited the expressivity of the logic supported by DeLorean to $Z\,\mathcal{SHOIN}$ (OWL DL). From a historical point of view, this version was the first reasoner that supported a fuzzy extension of the OWL DL language. Only a few optimizations were implemented.

With the aim of augmenting the expressivity of the logic, we changed the subjacent API to OWL API 3 [21]. OWL API 3 is supported by several ontology reasoners, such as *Pellet* [22], *Fact++* [23] and *HermiT*. Now, DeLorean supports both $Z\,\mathcal{SROIQ}(\mathbf{D})$ and $G\,\mathcal{SROIQ}(\mathbf{D})$.

One of the most important differences of OWL API 3 is that it replaces DIG by OWLlink support. *OWLlink* is an extensible protocol for communication between OWL 2 systems that supersedes DIG 1.1 [25]. Since OWLlink is not widely supported yet, we have also integrated natively *Pellet* and *HermiT* reasoners with DeLorean. Hence, the user is free to choose either one of these reasoners or a generic one via OWLlink protocol.

5.2 Optimizations

We will summarize here the main optimizations that the reasoner implements. The interested reader is referred to [15] for details.

Optimizing the Number of New Elements and Axioms. As seen in Section 4.1, we must add some new concepts, roles, and axioms to compute an equivalent crisp ontology. DeLorean reduces the number of new concepts and roles introduced with respect to a direct translation. For instance, a crisp concept denoting individuals that belong to a fuzzy concept with a degree less than α is not needed, since we can use the negation of the α-cut of the fuzzy concept. As a consequence of the reduction of the number of concepts and roles, the number of new necessary axioms is reduced as well.

Optimizing GCI Reductions. In some particular cases, the crisp representation of fuzzy GCIs can be optimized. For example, domain role axioms, range role axioms, functional role axioms and disjoint concept axioms can be optimized

[9] `http://jena.sourceforge.net`
[10] `https://javacc.dev.java.net`

if we manage them as particular cases, instead of considering them as GCIs. Furthermore, some axioms in the resulting TBox may be unnecessary since they can be entailed by other axioms.

Allowing Crisp Concepts and Roles. In order to represent a fuzzy concept, we need to introduce several new concepts, and some new axioms keeping the semantics among them. In real applications, not all concepts and roles are fuzzy. If a concept is declared as crisp, we just need one concept to represent it and no new axioms. The case for fuzzy roles is exactly the same. DeLorean makes it possible to define crisp concepts and roles. Of course, this optimization requires some manual intervention during the identification of the crisp elements.

Ignoring Superfluous Elements While Reasoning. The computation of the equivalent crisp ontology can be designed to promote reusing or efficiency. A direct translation into the crisp case makes ontology reuse easier when new axioms are added. The drawback is that reasoning is less efficient. Depending on the reasoning task, DeLorean promotes reusing or avoiding superfluous elements and recomputing the crisp representation when necessary.

6 Use Case: A Fuzzy Wine Ontology

This section describes a concrete use case: a fuzzy extension of the well-known Wine ontology[11], a highly expressive $\mathcal{SHOIN}(\mathbf{D})$ ontology. Some metrics of the ontology are shown in the first column of Table 3.

There is a previous empirical evaluation of the reductions of fuzzy DLs to crisp DLs [27], but the only optimization thereby considered applies to the number of new elements and axioms. We will show that the additional optimizations hereby proposed, specially the (natural) assumption that there are some crisp elements, reduce significantly the number of axioms.

A Fuzzy Extension of the Ontology. We have defined a fuzzy version of the Wine ontology by adding a degree to the axioms. Given a variable set of degrees $N^{\mathcal{K}}$, the degrees of truth for fuzzy assertions is randomly chosen in $N^{\mathcal{K}}$. In the case of fuzzy GCIs and RIAs, the degree is always 1 in special GCIs (namely concept equivalences and disjointness, domain, range and functional role axioms) or if there is a crisp element in the left side; otherwise, the degree is 0.5.

Most fuzzy assertions are of the form $\langle \tau \rhd \beta \rangle$ with $\beta \neq 1$. This favors the use of elements of the forms $C_{\rhd \beta}$ and $R_{\rhd \beta}$, which reduces the number of superfluous concepts as seen in Section 5.2. Once again, this leads to the worst case from the point of view of the size of the resulting crisp ontology. Nonetheless, in practice we will be often able to say that an individual fully belongs to a fuzzy concept, or that two individuals are fully related by means of a fuzzy role, which is not the worst case.

[11] `http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine.rdf`

Note that our objective is to build a test case to experiment with and not to build a fuzzy Wine ontology with interest in the real world. For a serious attempt to build a fuzzy Wine ontology, we refer the reader to [28].

Crisp Concepts and Roles. A careful analysis of the fuzzy $\mathcal{SROIQ}(\mathbf{D})$ KB brings about that most concepts and roles should be indeed interpreted as crisp –actually, the classification of concepts and roles into fuzzy or crisp in most cases may be very subjective and application-dependent.

For example, most subclasses of the class Wine refer to a well-defined geographical origin of the wines. For instance, Alsatian wine is a wine which has been produced in the French region of Alsace: $\mathsf{AlsatianWine} \equiv \mathsf{Wine} \sqcap \exists \mathsf{locatedAt}.\{\mathsf{alsaceRegion}\}$. In other applications there could be examples of fuzzy regions, but this is not our case.

Another important number of subclasses of Wine refer to the type of grape used, which is also a crisp concept. For instance, Riesling is a wine which has been produced from Riesling grapes: $\mathsf{Riesling} \equiv \mathsf{Wine} \sqcap \exists \mathsf{madeFromGrape}.\{\mathsf{RieslingGrape}\} \sqcap \geq 1\ \mathsf{madeFromGrape}.\top$.

The result of our study has identified 50 fuzzy concepts in the Wine ontology. The source of the vagueness is summarized in several categories[12]:

– Color of the wine: WineColor, RedWine, RoseWine, WhiteWine, RedBordeaux, RedBurgundy, RedTableWine, WhiteBordeaux, WhiteBurgundy, WhiteLoire, WhiteTableWine.
– Sweetness of the wine: WineSugar, SweetWine, SweetRiesling, WhiteNonSweetWine, DryWine, DryRedWine, DryRiesling, DryWhiteWine.
– Body of the wine: WineBody, FullBodiedWine.
– Flavor of the wine: WineFlavor, WineTaste.
– Age of the harvest: LateHarvest, EarlyHarvest.
– Spiciness of the food: NonSpicyRedMeat, NonSpicyRedMeatCourse, SpicyRedMeat, PastaWithSpicyRedSauce, PastaWithSpicyRedSauceCourse, PastaWithNonSpicyRedSauce, PastaWithNonSpicyRedSauceCourse, SpicyRedMeatCourse.
– Sweetness of the food: SweetFruit, SweetFruitCourse, SweetDessert, SweetDessertCourse, NonSweetFruit, NonSweetFruitCourse.
– Type of the meat: RedMeat, NonRedMeat, RedMeatCourse, NonRedMeatCourse. They are fuzzy because, according to the age of the animal, pork and lamb are classified as red (old animals) or white (young animals) meat.
– Heaviness of the cream: PastaWithHeavyCreamSauce, PastaWithLightCreamSauce. In this case the terms “heavy” and “light” depend on the fat percentage, and thus can be a matter of degree.
– Desserts: Dessert, DessertWine, CheeseNutsDessert, DessertCourse, CheeseNutsDessertCourse. We make these concepts fuzzy as the question whether something is a dessert or not does not have a clear answer.

[12] Clearly, these categories are not disjoint and some concepts may belong to more than one, meaning that they are fuzzy for several reasons. For example, DryRedWine is a fuzzy concept because both “dry” and “red” are vague terms.

As already discussed, the color, the sweetness, the body and the flavor of a wine are fuzzy. As a consequence, we can identify 5 fuzzy roles: hasColor, hasSugar, hasBody, hasFlavor, and hasWineDescriptor, where the role hasWineDescriptor is a super-role of the others.

Measuring the Importance of the Optimizations. Here, we will restrict to $Z\ \mathcal{SROIQ}$ (omitting the concrete role yearValue), but allowing the use of Kleene-Dienes and Gödel implications in the semantics of the axioms (A8), (A10), (A11).

Table 3 shows some metrics of the crisp ontologies obtained in the reduction of the fuzzy ontology after applying different optimizations.

1. Column "Original" shows some metrics of the original ontology.
2. "None" considers the reduction obtained after applying no optimizations.
3. "(NEW)" considers the reduction obtained after optimizing the number of new elements and axioms.
4. "(GCI)" considers the reduction obtained after optimizing GCI reductions.
5. "(C/S)" considers the reduction obtained after allowing crisp concepts and roles and ignoring superfluous elements.
6. Finally, "All" applies all the previous optimizations.

Table 3. Metrics of the Wine ontology and its fuzzy versions using 5 degrees

	Original	None	(NEW)	(GCI)	(C/S)	All
Individuals	206	206	206	206	206	206
Named concepts	136	2176	486	2176	800	191
Abstract roles	16	128	128	128	51	20
Concept assertions	194	194	194	194	194	194
Role assertions	246	246	246	246	246	246
Inequality assertions	3	3	3	3	3	3
Equality assertions	0	0	0	0	0	0
New GCIs	0	4352	952	4352	1686	324
Subclass axioms	275	1288	1288	931	390	390
Concept equivalences	87	696	696	696	318	318
Disjoint concepts	19	152	152	19	152	19
Domain role axioms	13	104	104	97	104	97
Range role axioms	10	80	80	10	80	10
Functional role axioms	6	48	48	6	48	6
New RIAs	0	136	119	136	34	34
Sub-role axioms	5	40	40	40	33	33
Role equivalences	0	0	0	0	0	0
Inverse role axioms	2	16	16	16	2	2
Transitive role axioms	1	8	8	8	1	1

Note that the size of the ABox is always the same, because every axiom in the fuzzy ABox generates exactly one axiom in the reduced ontology.

The number of new GCIs and RIAs added to preserve the semantics of the new elements is much smaller in the optimized versions. In particular, we reduce from 4352 to 324 GCIs (7.44%) and from 136 to 34 RIAs (25%). This shows the importance of reducing the number of new crisp elements and their corresponding axioms, defining crisp concepts and roles, and (to a lesser extent) handling superfluous concepts.

Optimizing GCI reductions turns out to be very useful in reducing the number of disjoint concepts, domain, range and functional role axioms: 152 to 19 (12.5 %), 104 to 97 (93.27 %), 80 to 10 (12.5 %), and 48 to 6 (12.5 %), respectively. In the case of domain role, axioms the reduction is not very high because we need an inverse role to be defined in order to apply the reduction. However, this happens for only one axiom.

Every fuzzy GCI or RIA generates several axioms in the reduced ontology. Combining the optimization of GCI reductions with the definition of crisp concepts and roles reduces the number of new axioms: from 1288 to 390 subclass axioms (30.28 %), from 696 to 318 concept equivalences (45.69 %), and from 40 to 33 sub-role axioms (82.5 %).

Finally, the number of inverse and transitive role axioms is reduced in the optimized version because fuzzy roles interpreted as crisp introduce one inverse or transitive axiom instead of several ones. This allows a reduction from 16 to 2 axioms, and from 8 to 1, respectively, which corresponds to the 12.5 %.

Table 4 shows the influence of the number of degrees on the size of the resulting crisp ontology, as well as on the reduction time (which is shown in seconds), when all the described optimizations are used. The reduction time is small enough to allow us to recompute the reduction of an ontology when necessary, thus allowing us to avoid superfluous concepts and roles.

Table 4. Influence of the number of degrees in the reduction

	Crisp	3	5	7	9	11	21
Number of axioms	811	1166	1674	2182	2690	3198	5738
Reduction time	-	0.343	0.453	0.64	0.782	0.859	1.75

7 Conclusions and Future Work

This paper has presented the main features of the fuzzy ontology reasoner DeLorean, the first one that supports a fuzzy extension of OWL 2. DeLorean integrates translation and reasoning tasks. Given a fuzzy ontology, DeLorean computes its equivalent non-fuzzy representation. Then, it uses a classical DL reasoner to perform the reasoning procedures.

We have also discussed how to create fuzzy ontologies by using a related Protégé plug-in, as well as the implemented optimizations. Optimizations allow us to define crisp concepts and roles and to remove superfluous concepts and roles before applying crisp reasoning. A preliminary evaluation shows that these optimizations help to reduce significantly the size of the resulting ontology.

Other fuzzy ontology reasoners can be found in the literature, e.g. Fire [5][13], fuzzyDL [6][14], GURDL [7], GERDS [8][15], YADLR [9], DLMedia [11][16], FRESG [12] or ONTOSEARCH2 [10][17]. On the one hand, the advantages of DeLorean are that it supports all the constructors of fuzzy $\mathcal{SROIQ}(\mathbf{D})$, and makes it possible to reuse existing ontology languages, editors, reasoners, and other existing resources. On the other hand, the equivalent crisp ontologies computed by DeLorean are larger than the original fuzzy ontologies. Hence, other reasoners, such as ONTOSEARCH2 and DLMedia –which are specifically designed for scalable reasoning–, would very likely show a better performance for the fuzzy OWL 2 profiles. Furthermore, some of these reasoners implement some features that DeLorean currently does not support. Some reasoners solve new alternative reasoning tasks, such as classification (Fire) or defuzzification (fuzzyDL), or support different constructors, such as aggregation operators (fuzzyDL), extended fuzzy concrete domains (FRESG), or alternative uncertainty operators (GURDL).

In future work, we plan to develop a more detailed benchmark and, eventually, to compare DeLorean against other fuzzy DL reasoners. This is a difficult task, since different reasoners support different features and expressivities. Moreover, as far as we know, nowadays there are no public real-world fuzzy ontologies to test with.

References

1. Cuenca-Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P.F., Sattler, U.: OWL 2: The next step for OWL. Journal of Web Semantics 6(4), 309–322 (2008)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
3. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proceedings of the 10th International Conference of Knowledge Representation and Reasoning, KR 2006, pp. 452–457 (2006)
4. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in Description Logics for the Semantic Web. Journal of Web Semantics 6(4), 291–308 (2008)
5. Stoilos, G., Simou, N., Stamou, G., Kollias, S.: Uncertainty and the Semantic Web. IEEE Intelligent Systems 21(5), 84–87 (2006)
6. Bobillo, F., Straccia, U.: 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 (2008)
7. Haarslev, V., Pai, H.I., Shiri, N.: A formal framework for Description Logics with uncertainty. Int. Journal of Approximate Reasoning 50(9), 1399–1415 (2009)

[13] `http://www.image.ece.ntua.gr/~nsimou/FiRE`
[14] `http://straccia.info/software/fuzzyDL/fuzzyDL.html`
[15] `http://www1.osu.cz/home/habibal/page8.html`
[16] `http://straccia.info/software/DL-Media/DL-Media.html`
[17] `http://dipper.csd.abdn.ac.uk/OntoSearch`

8. Habiballa, H.: Resolution 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 (2007)
9. Konstantopoulos, S., Apostolikas, G.: Fuzzy-DL Reasoning over Unknown Fuzzy Degrees. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2007 Ws, Part II. LNCS, vol. 4806, pp. 1312–1318. Springer, Heidelberg (2007)
10. Pan, J.Z., Thomas, E., Sleeman, D.: ONTOSEARCH2: Searching and querying web ontologies. In: Proceeedings of the IADIS International Conference WWW/Internet 2006 (2006)
11. Straccia, U., Visco, G.: 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, vol. 423. CEUR Workshop Proceedings (2008)
12. Wang, H., Ma, Z.M., Yin, J.: FRESG: A Kind of Fuzzy Description Logic Reasoner. In: Bhowmick, S.S., Küng, J., Wagner, R. (eds.) DEXA 2009. LNCS, vol. 5690, pp. 443–450. Springer, Heidelberg (2009)
13. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
14. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer (1998)
15. Bobillo, F., Delgado, M., Gómez-Romero, J.: Crisp representations and reasoning for fuzzy ontologies. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 17(4), 501–530 (2009)
16. Bobillo, F., Delgado, M., Gómez-Romero, J., Straccia, U.: Fuzzy Description Logics under Gödel semantics. Int. J. of Approximate Reasoning 50(3), 494–514 (2009)
17. Stoilos, G., Stamou, G., Pan, J.Z.: Fuzzy extensions of OWL: Logical properties and reduction to fuzzy description logics. International Journal of Approximate Reasoning 51(6), 656–679 (2010)
18. Straccia, U.: Description Logics with fuzzy concrete domains. In: Proc. of the 21st Conf. on Uncertainty in Artificial Intelligence, UAI 2005. AUAI Press (2005)
19. Straccia, U.: Reasoning within fuzzy Description Logics. Journal of Artificial Intelligence Research 14, 137–166 (2001)
20. Bechhofer, S., Möller, R., Crowther, P.: The DIG Description Logic interface: DIG/1.1. In: Proceedings of the 16th International Workshop on Description Logics, DL 2003 (2003)
21. Horridge, M., Bechhofer, S.: The OWL API: A Java API for OWL Ontologies. Semantic Web 2(1), 11–21 (2011)
22. Sirin, E., Parsia, B., Cuenca-Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics 5(2), 51–53 (2007)
23. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic Reasoner: System Description. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 292–297. Springer, Heidelberg (2006)
24. Shearer, R., Motik, B., Horrocks, I.: HermiT: A highly-efficient OWL reasoner. In: Proceedings of the 5th International Workshop on OWL: Experiences and Directions, OWLED 2008 (2008)
25. Liebig, T., Luther, M., Noppens, O., Wessel, M.: OWLlink. Semantic Web 2(1), 23–32 (2011)
26. Bobillo, F., Straccia, U.: Fuzzy ontology representation using OWL 2. International Journal of Approximate Reasoning 52(7), 1073–1094 (2011)

27. Cimiano, P., Haase, P., Ji, Q., Mailis, T., Stamou, G.B., Stoilos, G., Tran, T., Tzouvaras, V.: 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, ARea 2008. CEUR Workshop Proceedings, vol. 350 (2008)
28. Carlsson, C., Brunelli, M., Mezei, J.: Fuzzy ontologies and knowledge mobilisation: Turning amateurs into wine connoisseurs. In: Proceedings of the 19th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2010, pp. 1718–1723 (2010)