INSTITUT FÜR INFORMATIONSSYSTEME

ARBEITSBEREICH WISSENSBASIERTE SYSTEME

UNCERTAINTY AND VAGUENESS IN DESCRIPTION LOGIC PROGRAMS FOR THE SEMANTIC WEB

THOMAS LUKASIEWICZ and UMBERTO STRACCIA

INFSYS RESEARCH REPORT 1843-07-02

FEBRUARY & APRIL 2007

INFSYS RESEARCH REPORT

INFSYS RESEARCH REPORT 1843-07-02, FEBRUARY & APRIL 2007

UNCERTAINTY AND VAGUENESS IN DESCRIPTION LOGIC PROGRAMS FOR THE SEMANTIC WEB

APRIL 30, 2007

Thomas Lukasiewicz [1]   Umberto Straccia [2]

Abstract. This paper is directed towards an infrastructure for handling both uncertainty and vagueness in the Rules, Logic, and Proof layers of the Semantic Web. More concretely, we present probabilistic fuzzy description logic programs, which combine fuzzy description logics, fuzzy logic programs (with stratified nonmonotonic negation), and probabilistic uncertainty in a uniform framework for the Semantic Web. We define important concepts dealing with both probabilistic uncertainty and fuzzy vagueness, such as the expected truth value of a crisp sentence and the probability of a vague sentence. We then provide algorithms for query processing in probabilistic fuzzy description logic programs, and we also delineate a special case where query processing has a polynomial data complexity. Furthermore, we describe a shopping agent example, which gives evidence of the usefulness of probabilistic fuzzy description logic programs in realistic web applications.

[1]Dipartimento di Informatica e Sistemistica, Sapienza Università di Roma, Via Salaria 113, I-00198 Rome, Italy; e-mail: lukasiewicz@dis.uniroma1.it. Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, A-1040 Vienna, Austria; e-mail: lukasiewicz@kr.tuwien.ac.at.

[2]ISTI-CNR, Via G. Moruzzi 1, I-56124 Pisa, Italy; e-mail: straccia@isti.cnr.it.

Acknowledgements: This work has been partially supported by a Heisenberg Professorship of the German Research Foundation (DFG).

Contents

1 Introduction

The *Semantic Web* [1, 7] aims at an extension of the current World Wide Web by standards and technologies that help machines to understand the information on the Web so that they can support richer discovery, data integration, navigation, and automation of tasks. The main ideas behind it are to add a machine-understandable meaning to Web pages, to use ontologies for a precise definition of shared terms in Web resources, to use KR technology for automated reasoning from Web resources, and to apply cooperative agent technology for processing the information of the Web.

The Semantic Web consists of several hierarchical layers, where the *Ontology layer*, in form of the *OWL Web Ontology Language* [31, 13], is currently the highest layer of sufficient maturity. OWL consists of three increasingly expressive sublanguages, namely, *OWL Lite*, *OWL DL*, and *OWL Full*. OWL Lite and OWL DL are essentially very expressive description logics with an RDF syntax [13]. As shown in [12], ontology entailment in OWL Lite (resp., OWL DL) reduces to knowledge base (un)satisfiability in the description logic $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$). On top of the Ontology layer, sophisticated representation and reasoning capabilities for the *Rules*, *Logic*, and *Proof layers* of the Semantic Web are being developed next.

In particular, a significant body of recent research is trying to address a key requirement of the layered architecture of the Semantic Web, which is to integrate the Rules and the Ontology layer. Here, it is crucial to allow for building rules on top of ontologies, that is, for rule-based systems that use vocabulary from ontology knowledge bases. Another type of combination is to build ontologies on top of rules, which means that ontological definitions are supplemented by rules or imported from rules. Both types of integration have been realized in recent hybrid integrations of rules and ontologies under the loose coupling, called *description logic programs* (or simply *dl-programs*), which have the form $KB = (L, P)$, where L is a description logic knowledge base and P is a finite set of rules involving queries to L [4].

Other research efforts are directed towards formalisms for *handling uncertainty and vagueness in the Semantic Web*, which are motivated by important web and semantic web applications. In particular, formalisms for handling uncertainty are used in data integration, ontology mapping, and information retrieval, while dealing with vagueness is motivated by multimedia information processing / retrieval and natural language interfaces to the Web. There are several extensions of description logics and web ontology languages by probabilistic uncertainty and fuzzy vagueness. Similarly, there are also extensions of description logic programs by probabilistic uncertainty [14] and fuzzy vagueness [26, 15].

Clearly, since uncertainty and vagueness are semantically quite different, it is important to have a unifying formalism for the Semantic Web, which allows for dealing with both uncertainty and vagueness. But even though there has been some important work in the fuzzy logic community in this direction [9], to date there are no description logic programs that allow for handling both uncertainty and vagueness.

In this paper, we try to fill this gap. We present a novel approach to description logic programs, where probabilistic rules are defined on top of fuzzy rules, which are in turn defined on top of fuzzy description logics. This approach allows for handling both probabilistic uncertainty and fuzzy vagueness. Intuitively, it essentially allows for defining several rankings on ground atoms using fuzzy vagueness, and then for merging these rankings using probabilistic uncertainty (by associating with each ranking a probabilistic weight and building the weighted sum of all rankings). The main contributions are as follows:

- We present probabilistic fuzzy description logic programs, which combine (i) fuzzy description logics, (ii) fuzzy logic programs (with stratified nonmonotonic negation), and (iii) probabilistic uncertainty in a uniform framework for the Semantic Web. Such programs allow for handling both probabilistic uncertainty (especially for probabilistic ontology mapping and probabilistic data integration) and fuzzy vagueness (especially for dealing with vague concepts).

- We define important concepts dealing with both probabilistic uncertainty and fuzzy vagueness, such as the expected truth value of a crisp sentence and the probability of a vague sentence.
- We also give algorithms for query processing in probabilistic fuzzy description logic programs, and we delineate a special case where query processing has a polynomial data complexity (under suitable assumptions about the underlying fuzzy description logics), which is an important feature for the Web.
- Furthermore, we describe a shopping agent example, which gives evidence of the usefulness of probabilistic fuzzy description logic programs in realistic web applications.

The rest of this paper is organized as follows. Section 2 gives a motivating example. In Sections 3 and 4, we recall combination strategies and fuzzy description logics. Section 5 defines fuzzy dl-programs on top of fuzzy description logics. In Sections 6 and 7, we define probabilistic fuzzy dl-programs and provide algorithms for query processing in such programs. In Section 8, we delineate a special case where query processing has a polynomial data complexity. Section 9 summarizes our main results and gives an outlook on future research.

2 Motivating Example

In this section, we describe a shopping agent example, where we encounter both probabilistic uncertainty (in resource selection, ontology mapping / query transformation, and data integration) and fuzzy vagueness (in query matching with vague concepts).

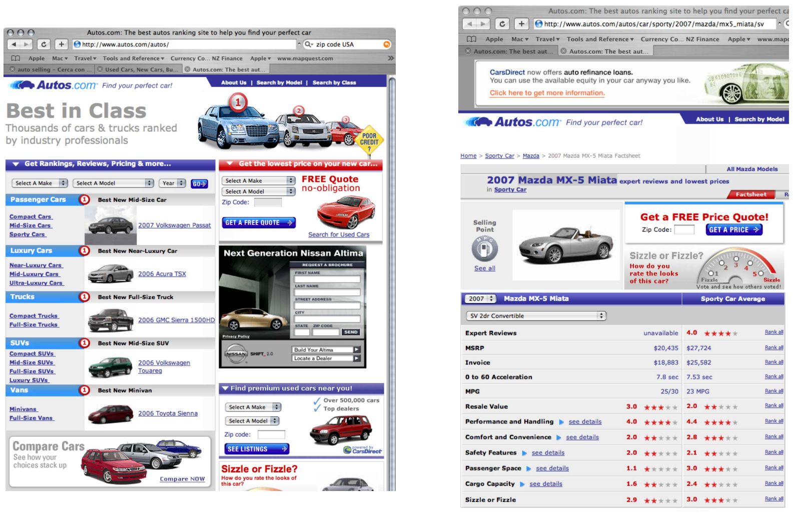
Example 2.1 (Shopping Agent) Suppose a person would like to buy "a sports car that costs at most about 22 000 € and that has a power of around 150 HP".

In todays Web, the buyer has to *manually* (i) search for car selling sites, e.g., using Google, (ii) select the most promising sites (Fig. 1 shows an excerpt of such a site; see `http://www.autos.com`), (iii) browse through them, query them to see the cars that they sell, and match the cars with our requirements, (iv) select the offers in each web site that match our requirements, and (v) eventually merge all the best offers from each site and select the best ones.

It is obvious that the whole process is rather *tedious* and *time consuming*, since e.g. (i) the buyer has to visit many sites, (ii) the browsing in each site is very time consuming, (iii) finding the right information in a site (which has to match the requirements) is not simple, and (iv) the way of browsing and querying may differ from site to site.

A *shopping agent* may now support us as follows, *automatizing* the whole selection process once it receives the request / query q from the buyer:

- *Probabilistic Resource Selection.* The agent selects some sites / resources S that it considers as promising for the buyer's request. The agent has to select a subset of some *relevant* resources, since it is not reasonable to assume that it will access and query all the resources known to him. The relevance of a resource S to a query is usually (automatically) estimated as the probability $Pr(q|S)$ (the probability that the information need represented by the query q is satisfied by the searching resource S, see e.g. [2, 8]). It is not difficult to see that such probabilities can be expressed by probabilistic rules.
- *Probabilistic Ontology Mapping / Query Reformulation.* For the top-k selected sites, the agent has to reformulate the buyer's query using the terminology / ontology of the specific car selling site. For this task, the agent relies on so-called transformation rules, which say how to translate a concept or


Autos.com: The best autos ranking site to help you find your perfect car
Autos.com Find your perfect car!
Best in Class
Thousands of cars & trucks ranked by industry professionals
Get Rankings, Reviews, Pricing & more...
Get the lowest price on your new car...
FREE Quote no-obligation
Passenger Cars
Luxury Cars
Trucks
SUVs
Vans
Next Generation Nissan Altima
Find premium used cars near you!
Compare Cars
Sizzle or Fizzle?
2007 Mazda MX-5 Miata expert reviews and lowest prices
Get a FREE Price Quote!
GET A PRICE
Sizzle or Fizzle? How do you rate the looks of this car?

2007 Mazda MX-5 Miata SV 2dr Convertible		Sporty Car Average	
Expert Reviews	unavailable	4.0 ★★★★☆	Rank all
MSRP	$20,435	$27,724	Rank all
Invoice	$18,883	$25,582	Rank all
0 to 60 Acceleration	7.8 sec	7.53 sec	Rank all
MPG	25/30	23 MPG	Rank all
Resale Value	3.0 ★★★☆☆	2.0 ★★☆☆☆	Rank all
Performance and Handling see details	4.0 ★★★★☆	4.4 ★★★★☆	Rank all
Comfort and Convenience see details	2.0 ★★☆☆☆	2.8 ★★★☆☆	Rank all
Safety Features see details	2.0 ★★☆☆☆	2.1 ★★☆☆☆	Rank all
Passenger Space see details	1.1 ★☆☆☆☆	3.0 ★★★☆☆	Rank all
Cargo Capacity see details	1.6 ★★☆☆☆	2.4 ★★☆☆☆	Rank all
Sizzle or Fizzle	2.9 ★★☆☆☆	3.0 ★★★☆☆	Rank all

Figure 1: A car shopping site

property of the agent's ontology into the ontology of the information resource. Once the set of rules is given, the query transformation is relatively easy. What is difficult is to learn the *ontology mapping* rules automatically. This task is called *ontology alignment* in the Semantic Web, and some tools for this exist (e.g., oMap [27, 28]). Often, to relate a concept B of the buyer's ontology to a concept S of the seller's ontology, one automatically estimates the probability $P(B|S)$ that an instance of S is also an instance of B. For example, oMap represents such rules as probabilistic rules (see also [20]).

- *Vague Query Matching.* Once the agent has translated the buyer's request for the specific site's terminology, the agent submits the query. But the buyer's request often contains many so-called *vague / fuzzy* concepts such as "the prize is around 22 000 € or less", rather than strict conditions, and thus a car may *match* the buyer's condition to a *degree*. As a consequence, a site / resource / web service may return a ranked list of cars, where the ranks depend on the degrees to which the sold items match the buyer's requests q.

- *Probabilistic Data Integration.* Eventually, the agent has to combine the ranked lists (see e.g. [23]) by considering the matching degrees, that is, truth degrees (vagueness) and probability degrees (uncertainty) involved and show the top-n items to the buyer.

3 Combination Strategies

Rather than being restricted to an ordinary binary truth value among **false** and **true**, *vague propositions* may also have a truth value strictly between **false** and **true**. In the sequel, we use the unit interval $[0, 1]$ as the set of all possible truth values, where 0 and 1 represent the ordinary binary truth values **false** and **true**, respectively. For example, the vague proposition "John is a tall man" may be more or less true, and it is thus associated with a truth value in $[0, 1]$, depending on the body height of John.

In order to combine and modify the truth values in $[0, 1]$, we assume *combination strategies*, namely, *conjunction*, *disjunction*, *implication*, and *negation strategies*, denoted $\otimes$, $\oplus$, $\rhd$, and $\ominus$, respectively, which

Table 1: Axioms for conjunction and disjunction strategies.

Axiom Name	Conjunction Strategy	Disjunction Strategy
Tautology / Contradiction	$a \otimes 0 = 0$	$a \oplus 1 = 1$
Identity	$a \otimes 1 = a$	$a \oplus 0 = a$
Commutativity	$a \otimes b = b \otimes a$	$a \oplus b = b \oplus a$
Associativity	$(a \otimes b) \otimes c = a \otimes (b \otimes c)$	$(a \oplus b) \oplus c = a \oplus (b \oplus c)$
Monotonicity	if $b \leqslant c$, then $a \otimes b \leqslant a \otimes c$	if $b \leqslant c$, then $a \oplus b \leqslant a \oplus c$

Table 2: Axioms for implication and negation strategies.

Axiom Name	Implication Strategy	Negation Strategy
Tautology / Contradiction	$0 \rhd b = 1,\ a \rhd 1 = 1,\ 1 \rhd 0 = 0$	$\ominus 0 = 1,\ \ominus 1 = 0$
Antitonicity	if $a \leqslant b$, then $a \rhd c \geqslant b \rhd c$	if $a \leqslant b$, then $\ominus a \geqslant \ominus b$
Monotonicity	if $b \leqslant c$, then $a \rhd b \leqslant a \rhd c$	

Table 3: Combination strategies of various fuzzy logics.

	Łukasiewicz Logic	Gödel Logic	Product Logic	Zadeh Logic
$a \otimes b$	$\max(a+b-1, 0)$	$\min(a, b)$	$a \cdot b$	$\min(a, b)$
$a \oplus b$	$\min(a+b, 1)$	$\max(a, b)$	$a + b - a \cdot b$	$\max(a, b)$
$a \rhd b$	$\min(1-a+b, 1)$	$\begin{cases} 1 & \text{if } a \leqslant b \\ b & \text{otherwise} \end{cases}$	$\min(1, b/a)$	$\max(1-a, b)$
$\ominus a$	$1-a$	$\begin{cases} 1 & \text{if } a = 0 \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} 1 & \text{if } a = 0 \\ 0 & \text{otherwise} \end{cases}$	$1-a$

are functions $\otimes$, $\oplus$, $\rhd\colon [0,1] \times [0,1] \rightarrow [0,1]$ and $\ominus\colon [0,1] \rightarrow [0,1]$ that generalize the ordinary Boolean operators $\wedge$, $\vee$, $\rightarrow$, and $\neg$, respectively, to the set of truth values $[0,1]$. For $a, b \in [0,1]$, we then call $a \otimes b$ (resp., $a \oplus b$, $a \rhd b$) the *conjunction* (resp., *disjunction*, *implication*) of a and b, and we call $\ominus a$ the *negation* of a. As usual, we assume that combination strategies have some natural algebraic properties, namely, the properties shown in Tables 1 and 2. Note that in Table 1, Tautology and Contradiction follow from Identity, Commutativity, and Monotonicity. Conjunction and disjunction strategies (with the properties in Table 1) are also called *triangular norms* and *triangular co-norms* [10], respectively. We do not assume properties that relate the combination strategies to each other (such as de Morgan's law). Even though one may additionally assume such properties, they are not required here.

Example 3.1 The combination strategies of various well-known fuzzy logics are shown in Table 3.

4 Fuzzy Description Logics

In this section, we recall fuzzy generalizations of the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, which stand behind OWL Lite and OWL DL, respectively; see especially [24, 25, 16]. Intuitively, description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals resp. binary relations between classes of individuals. A knowledge base encodes in particular subset relationships

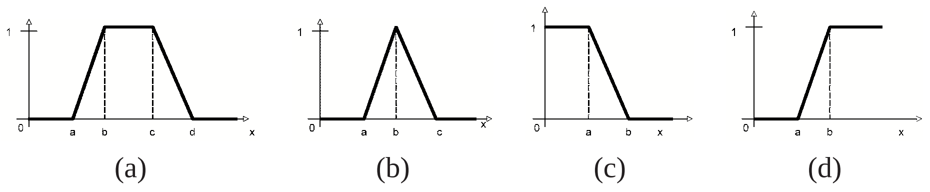


Figure 2: (a) Trapezoidal function $trz(x;a,b,c,d)$, (b) triangular function $tri(x;a,b,c)$, (c) left shoulder function $L(x;a,b)$, and (d) right shoulder function $R(x;a,b)$.

between concepts, subset relationships between roles, the membership of individuals to concepts, and the membership of pairs of individuals to roles. In fuzzy description logics, these relationships and memberships then have a degree of truth in $[0,1]$.

We now describe the syntax and the semantics of fuzzy $\mathcal{SHIF}(\mathbf{D})$ and fuzzy $\mathcal{SHOIN}(\mathbf{D})$ and illustrate them through an example. For an implementation of fuzzy $\mathcal{SHIF}(\mathbf{D})$, the *fuzzyDL* system, see http://gaia.isti.cnr.it/∼straccia.

4.1 Syntax

The elementary ingredients are as follows. We assume a set of *data values*, a set of *elementary datatypes*, and a set of *datatype predicates* (each with a predefined arity $n \geqslant 1$). A *datatype* is an elementary datatype or a finite set of data values. A *fuzzy datatype theory* $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a datatype domain $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each data value an element of $\Delta^{\mathbf{D}}$, to each elementary datatype a subset of $\Delta^{\mathbf{D}}$, and to each datatype predicate of arity n a fuzzy relation over $\Delta^{\mathbf{D}}$ of arity n (that is, a mapping $(\Delta^{\mathbf{D}})^n \rightarrow [0,1]$). We extend $\cdot^{\mathbf{D}}$ to all datatypes by $\{v_1,\ldots,v_n\}^{\mathbf{D}} = \{v_1^{\mathbf{D}},\ldots,v_n^{\mathbf{D}}\}$. For example, a crisp unary datatype predicate $\leqslant_{18}$ over the natural numbers denoting the integers of at most 18 may be defined by $\leqslant_{18}(x) = 1$, if $x \leqslant 18$, and $\leqslant_{18}(x) = 0$, otherwise. Then, $Minor = Person \sqcap \exists age.\leqslant_{18}$ defines a person of age at most 18. Non-crisp predicates are usually defined by functions for specifying fuzzy set membership degrees, such as the trapezoidal, the triangular, the left shoulder, and the right shoulder functions (see Fig. 2). For example, a fuzzy unary datatype predicate $Young$ over the natural numbers denoting the degree of youngness of a person's age may be defined by $Young(x) = L(x;10,30)$. Then, $YoungPerson = Person \sqcap \exists age.Young$ denotes a young person.

Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, $\mathbf{I}$, and $\mathbf{M}$ be pairwise disjoint sets of *atomic concepts*, *abstract roles*, *concrete roles*, *individuals*, and *fuzzy modifiers*, respectively. Note that a *fuzzy modifier* m [11, 30] represents a function $f_m\colon [0,1] \rightarrow [0,1]$, which applies to fuzzy sets to change their membership function. For example, we may have the fuzzy modifiers $very$ and $slightly$, which represent the functions $very(x) = x^2$ and $slightly(x) = \sqrt{x}$, respectively. Then, the concept of sports cars may be defined as $SportsCar = Car \sqcap \exists speed.very(High)$, where $High$ is a fuzzy datatype predicate over the domain of speed in km/h, which may be defined as $High(x) = R(x;80,250)$.

A *role* is any element of $\mathbf{R}_A \cup \mathbf{R}_A^- \cup \mathbf{R}_D$ (where $\mathbf{R}_A^-$ is the set of *inverses* R^- of all $R \in \mathbf{R}_A$). We define *concepts* inductively as follows. Each $A \in \mathbf{A}$ is a concept, $\bot$ and $\top$ are concepts, and if $a_1,\ldots,a_n \in \mathbf{I}$, then $\{a_1,\ldots,a_n\}$ is a concept (called *oneOf*). If C, C_1, C_2 are concepts, $R,S \in \mathbf{R}_A \cup \mathbf{R}_A^-$, and $m \in \mathbf{M}$, then $(C_1 \sqcap C_2)$, $(C_1 \sqcup C_2)$, $\neg C$, and $m(C)$ are concepts (called *conjunction*, *disjunction*, *negation*, and *fuzzy modification*, respectively), as well as $\exists R.C$, $\forall R.C$, $\geqslant nS$, and $\leqslant nS$ (called *exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. If D is a datatype and $T, T_1,\ldots,T_n \in \mathbf{R}_D$, then

$\exists T_1, \ldots, T_n.D$, $\forall T_1, \ldots, T_n.D$, $\geqslant nT$, and $\leqslant nT$ are concepts (called *datatype exists, value, atleast,* and *atmost restriction*, resp.) for an integer $n \geqslant 0$. We eliminate parentheses as usual.

A *crisp axiom* has one of the following forms: (1) $C \sqsubseteq D$ (called *concept inclusion axiom*), where C and D are concepts; (2) $R \sqsubseteq S$ (called *role inclusion axiom*), where either $R, S \in \mathbf{R}_A \cup \mathbf{R}_A^-$ or $R, S \in \mathbf{R}_D$; (3) $\mathrm{Trans}(R)$ (called *transitivity axiom*), where $R \in \mathbf{R}_A$; (4) $C(a)$ (called *concept assertion axiom*), where C is a concept and $a \in \mathbf{I}$; (5) $R(a,b)$ (resp., $U(a,v)$) (called *role assertion axiom*), where $R \in \mathbf{R}_A$ (resp., $U \in \mathbf{R}_D$) and $a, b \in \mathbf{I}$ (resp., $a \in \mathbf{I}$ and v is a data value); and (6) $a = b$ (resp., $a \neq b$) (*equality* (resp., *inequality*) *axiom*), where $a, b \in \mathbf{I}$. We define *fuzzy axioms* as follows: A *fuzzy concept inclusion* (resp., *fuzzy role inclusion, fuzzy concept assertion, fuzzy role assertion*) *axiom* is of the form $\alpha\,\theta\,n$, where α is a concept inclusion (resp., role inclusion, concept assertion, role assertion) axiom, $\theta \in \{\leqslant, =, \geqslant\}$, and $n \in [0,1]$. Informally, $\alpha \leqslant n$ (resp., $\alpha = n$, $\alpha \geqslant n$) encodes that the truth value of α is at most (resp., equal to, at least) n. We often use α to abbreviate $\alpha = 1$. A *fuzzy (description logic) knowledge base* L is a finite set of fuzzy axioms, transitivity axioms, and equality and inequality axioms. For decidability, number restrictions in L are restricted to simple abstract roles.

Fuzzy $\mathcal{SHIF}(\mathbf{D})$ has the same syntax as fuzzy $\mathcal{SHOIN}(\mathbf{D})$, but without the oneOf constructor and with the atleast and atmost constructors limited to 0 and 1.

Example 4.1 (Shopping Agent cont'd) The following axioms are an excerpt of the fuzzy description logic knowledge base L that conceptualizes the site in Example 2.1:

$$\mathit{Cars} \sqcup \mathit{Trucks} \sqcup \mathit{Vans} \sqcup \mathit{SUVs} \sqsubseteq \mathit{Vehicles}; \tag{1}$$

$$\mathit{PassengerCars} \sqcup \mathit{LuxuryCars} \sqsubseteq \mathit{Cars}; \tag{2}$$

$$\mathit{CompactCars} \sqcup \mathit{MidSizeCars} \sqcup \mathit{SportyCars} \sqsubseteq \mathit{PassengerCars}; \tag{3}$$

$$\begin{aligned}\mathit{Cars} \sqsubseteq\ & (\exists \mathit{hasReview}.\mathbf{Integer}) \sqcap (\exists \mathit{hasInvoice}.\mathbf{Integer}) \sqcap (\exists \mathit{hasHP}.\mathbf{Integer}) \\ & \sqcap (\exists \mathit{hasResellValue}.\mathbf{Integer}) \sqcap (\exists \mathit{hasSafetyFeatures}.\mathbf{Integer}) \sqcap \ldots;\end{aligned} \tag{4}$$

$$(\mathit{SportyCar} \sqcap (\exists \mathit{hasInvoice}.\{18883\}) \sqcap (\exists \mathit{hasHP}.\{166\}) \sqcap \ldots)(\mathit{MazdaMX5Miata}); \tag{5}$$

$$(\mathit{SportyCar} \sqcap (\exists \mathit{hasInvoice}.\{20341\}) \sqcap (\exists \mathit{hasHP}.\{200\}) \sqcap \ldots)(\mathit{VolkswagenGTI}); \tag{6}$$

$$(\mathit{SportyCar} \sqcap (\exists \mathit{hasInvoice}.\{24029\}) \sqcap (\exists \mathit{hasHP}.\{162\}) \sqcap \ldots)(\mathit{MitsubishiES}). \tag{7}$$

Here, axioms (1)–(3) describe the concept taxonomy of the site, while axiom (4) describes the datatype attributes of the cars sold in the site. For example, every passenger or luxury car is also a car, and every car has a resell value. Axioms (5)–(7) describe the properties of some sold cars. For example, the $\mathit{MazdaMX5Miata}$ is a sports car, costing 18 883 €. Note that $\mathbf{Integer}$ denotes the datatype of all integers.

We may now encode "costs at most about 22 000 € " and "has a power of around 150 HP" in the buyer's request through the following concepts C and D, respectively:

$$C = \exists \mathit{hasInvoice}.\mathit{LeqAbout22000} \quad \text{and} \quad D = \exists \mathit{hasHP}.\mathit{Around150HP},$$

where $\mathit{LeqAbout22000} = L(22000, 25000)$ and $\mathit{Around150HP} = \mathit{Tri}(125, 150, 175)$ (see Fig. 2). The latter two equations define the fuzzy concepts of "at most about 22 000 € " and "around 150 HP". The former is modeled as a left shoulder function stating that if the prize is less than 22 000, then the degree of truth (degree of buyer's satisfaction) is 1, else the truth is linearly decreasing to 0 (reached at the cost of 25 000). In fact, we are modeling a case were the buyer would like to pay less than 22 000, though may still accept a higher price (up to 25 000) to a lesser degree. Similarly, the latter models the fuzzy concept "around 150 HP" as a triangular function with vertice in 150 HP.

4.2 Semantics

Concerning the semantics of fuzzy $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ [25], the main idea is that concepts and roles are interpreted as fuzzy subsets of an interpretation's domain. Therefore, concept inclusion, role inclusion, concept assertion, and role assertion axioms, rather than being satisfied (true) or unsatisfied (false) in an interpretation, have a degree of truth in $[0,1]$. In the sequel, we assume that $\otimes$, $\oplus$, $\rhd$, and $\ominus$ are some arbitrary but fixed conjunction, disjunction, implication, and negation strategies, respectively. A *fuzzy interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ relative to a fuzzy datatype theory $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a nonempty set $\Delta^{\mathcal{I}}$ (called the *domain*), disjoint from $\Delta^{\mathbf{D}}$, and a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$, which (i) coincides with $\cdot^{\mathbf{D}}$ on every data value, datatype, and fuzzy datatype predicate, (ii) assigns to each modifier $m \in \mathbf{M}$ its modifier function $f_m\colon [0,1] \to [0,1]$, and (iii) assigns

- to each individual $a \in \mathbf{I}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$;
- to each atomic concept $C \in \mathbf{A}$ a function $C^{\mathcal{I}}\colon \Delta^{\mathcal{I}} \to [0,1]$;
- to each abstract role $R \in \mathbf{R}_A$ a function $R^{\mathcal{I}}\colon \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1]$;
- to each concrete role $T \in \mathbf{R}_D$ a function $T^{\mathcal{I}}\colon \Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}} \to [0,1]$.

The mapping $\cdot^{\mathcal{I}}$ is extended to all roles and concepts as follows (where $x, y \in \Delta^{\mathcal{I}}$):

$$\begin{array}{rcl}
(S^-)^{\mathcal{I}}(x,y) & = & S^{\mathcal{I}}(y,x)\,; \\
\top^{\mathcal{I}}(x) & = & 1\,; \\
\bot^{\mathcal{I}}(x) & = & 0\,; \\
\{a_1,\ldots,a_n\}^{\mathcal{I}}(x) & = & \begin{cases} 1 & \text{if } x \in \{a_1{}^{\mathcal{I}},\ldots,a_n{}^{\mathcal{I}}\}\,; \\ 0 & \text{otherwise}\,; \end{cases} \\
(C_1 \sqcap C_2)^{\mathcal{I}}(x) & = & C_1{}^{\mathcal{I}}(x) \otimes C_2{}^{\mathcal{I}}(x)\,; \\
(C_1 \sqcup C_2)^{\mathcal{I}}(x) & = & C_1{}^{\mathcal{I}}(x) \oplus C_2{}^{\mathcal{I}}(x)\,; \\
(\neg C)^{\mathcal{I}}(x) & = & \ominus\, C^{\mathcal{I}}(x)\,; \\
(m(C))^{\mathcal{I}}(x) & = & f_m(C^{\mathcal{I}}(x))\,; \\
(\exists R.C)^{\mathcal{I}}(x) & = & \sup_{y \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)\,; \\
(\forall R.C)^{\mathcal{I}}(x) & = & \inf_{y \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,y) \rhd C^{\mathcal{I}}(y)\,; \\
({\geqslant}\, n\, S)^{\mathcal{I}}(x) & = & \sup_{y_1,\ldots,y_n \in \Delta^{\mathcal{I}},\, |\{y_1,\ldots,y_n\}| = n} \bigotimes_{i=1}^{n} S^{\mathcal{I}}(x,y_i)\,; \\
({\leqslant}\, n\, S)^{\mathcal{I}}(x) & = & \inf_{y_1,\ldots,y_{n+1} \in \Delta^{\mathcal{I}},\, |\{y_1,\ldots,y_{n+1}\}| = n+1} \bigoplus_{i=1}^{n+1} \ominus\, S^{\mathcal{I}}(x,y_i)\,; \\
(\exists T_1,\ldots,T_n.D)^{\mathcal{I}}(x) & = & \sup_{y_1,\ldots,y_n \in \Delta^{\mathbf{D}}} (\bigotimes_{i=1}^{n} T_i^{\mathcal{I}}(x,y_i)) \otimes D^{\mathbf{D}}(y_1,\ldots,y_n)\,; \\
(\forall T_1,\ldots,T_n.D)^{\mathcal{I}}(x) & = & \inf_{y_1,\ldots,y_n \in \Delta^{\mathbf{D}}} (\bigotimes_{i=1}^{n} T_i^{\mathcal{I}}(x,y_i)) \rhd D^{\mathbf{D}}(y_1,\ldots,y_n)\,.
\end{array}$$

Note here that individuals are "crisply" interpreted, as opposed to concepts and roles. The mapping $\cdot^{\mathcal{I}}$ is extended to concept inclusion, role inclusion, concept assertion, and role assertion axioms as follows:

$$\begin{array}{rcl}
(C_1 \sqsubseteq C_2)^{\mathcal{I}} & = & \inf_{x \in \Delta^{\mathcal{I}}} C_1{}^{\mathcal{I}}(x) \rhd C_2{}^{\mathcal{I}}(x)\,; \\
(R_1 \sqsubseteq R_2)^{\mathcal{I}} & = & \inf_{x,y \in \Delta^{\mathcal{I}}} R_1{}^{\mathcal{I}}(x,y) \rhd R_2{}^{\mathcal{I}}(x,y)\,; \\
(T_1 \sqsubseteq T_2)^{\mathcal{I}} & = & \inf_{(x,y) \in \Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}}} T_1{}^{\mathcal{I}}(x,y) \rhd T_2{}^{\mathcal{I}}(x,y)\,; \\
(C(a))^{\mathcal{I}} & = & C^{\mathcal{I}}(a^{\mathcal{I}})\,; \\
(R(a,b))^{\mathcal{I}} & = & R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}})\,; \\
(T(a,v))^{\mathcal{I}} & = & T^{\mathcal{I}}(a^{\mathcal{I}}, v^{\mathbf{D}})\,.
\end{array}$$

The notion of a fuzzy interpretation $\mathcal{I}$ *satisfying* a transitivity, equality, inequality, or fuzzy axiom E, or $\mathcal{I}$ being a *model* of E, denoted $\mathcal{I} \models E$, is defined as follows: (i) $\mathcal{I} \models \mathit{trans}(R)$ iff $R^{\mathcal{I}}(x,y) \geqslant \sup_{z \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,z) \otimes R^{\mathcal{I}}(z,y)$ for all $x, y \in \Delta^{\mathcal{I}}$; (ii) $\mathcal{I} \models a = b$ iff $a^{\mathcal{I}} = b^{\mathcal{I}}$, and $\mathcal{I} \models a \neq b$ iff $a^{\mathcal{I}} \neq b^{\mathcal{I}}$; and (iii) $\mathcal{I} \models \alpha\,\theta\,n$ iff $\alpha^{\mathcal{I}}\,\theta\,n$. We say $\mathcal{I}$ *satisfies* a fuzzy knowledge base L, or $\mathcal{I}$ is a *model* of L, denoted $\mathcal{I} \models L$, iff $\mathcal{I}$ is a model of all $E \in L$. We say L is *satisfiable* iff L has a model. A fuzzy axiom E is a *logical consequence* of L, denoted $L \models E$, iff every model of L satisfies E. A fuzzy axiom $\alpha \geqslant n$ is a *tight logical consequence* of L, denoted $L \models_{\mathit{tight}} \alpha \geqslant n$, iff n is the supremum of $m \in [0,1]$ subject to $L \models \alpha \geqslant m$.

Example 4.2 (Shopping Agent cont'd) The following fuzzy axioms are (tight) logical consequences of the above description logic knowledge base L (under the Zadeh semantics of the connectives):

$$\begin{array}{lll} C(\mathit{MazdaMX5Miata}) = 1.0, & C(\mathit{VolkswagenGTI}) = 1.0, & C(\mathit{MitsubishiES}) = 0.32, \\ D(\mathit{MazdaMX5Miata}) = 0.36, & D(\mathit{VolkswagenGTI}) = 0.0, & D(\mathit{MitsubishiES}) = 0.56. \end{array}$$

5 Fuzzy Description Logic Programs

In this section, we define fuzzy dl-programs, which are similar to the fuzzy dl-programs in [15], except that they are based on fuzzy description logics as in [25], and that we consider only stratified fuzzy dl-programs here. Their canonical model associates with every ground atom a truth value, and so defines a ranking on the Herbrand base. We first introduce the syntax, and we then define the semantics of positive and stratified fuzzy dl-programs in terms of a least model semantics resp. an iterative least model semantics.

5.1 Syntax of Fuzzy Programs

Informally, a normal fuzzy program is a finite collection of normal fuzzy rules, which are similar to ordinary normal rules, except that (i) they have a lower bound for their truth value, and (ii) they refer to fuzzy rather than binary interpretations, and thus every of their logical operators is associated with a combination strategy to specify how the operator combines truth values. Formally, we assume a first-order vocabulary Φ with nonempty finite sets of constant and predicate symbols, but no function symbols. We use Φ_c to denote the set of all constant symbols in Φ. Let $\mathcal{X}$ be a set of variables. A *term* is a constant symbol from Φ or a variable from $\mathcal{X}$. If p is a predicate symbol of arity $k \geqslant 0$ from Φ, and $t_1, \ldots, t_k$ are terms, then $p(t_1, \ldots, t_k)$ is an *atom*. A *literal* is an atom a or a default-negated atom $\mathit{not}\, a$. A *(normal) fuzzy rule* r has the form

$$\begin{array}{l} a \leftarrow_{\otimes_0} b_1 \wedge_{\otimes_1} b_2 \wedge_{\otimes_2} \cdots \wedge_{\otimes_{k-1}} b_k \wedge_{\otimes_k} \\ \qquad\qquad \mathit{not}_{\ominus_{k+1}}\, b_{k+1} \wedge_{\otimes_{k+1}} \cdots \wedge_{\otimes_{m-1}} \mathit{not}_{\ominus_m}\, b_m \;\geqslant\; v\,, \end{array} \tag{8}$$

where $m \geqslant k \geqslant 0$, $a, b_{k+1}, \ldots, b_m$ are atoms, $b_1, \ldots, b_k$ are either atoms or truth values from $[0,1]$, $\otimes_0, \ldots, \otimes_{m-1}$ are conjunction strategies, $\ominus_{k+1}, \ldots, \ominus_m$ are negation strategies, and $v \in (0,1]$. We call a the *head* of r, denoted $H(r)$, while $b_1 \wedge_{\otimes_1} \ldots \wedge_{\otimes_{m-1}} \mathit{not}_{\ominus_m}\, b_m$ is the *body* of r, and v is the *truth value* of r. We denote by $B(r)$ the set of body literals $B^+(r) \cup B^-(r)$, where $B^+(r) = \{b_1, \ldots, b_k\}$ and $B^-(r) = \{b_{k+1}, \ldots, b_m\}$. We call a fuzzy rule of the form (8) a *fuzzy fact* iff $m = 0$. A *normal fuzzy program* P is a finite set of fuzzy rules. We say that P is *positive* iff no fuzzy rule in P contains default-negated atoms.

5.2 Syntax of Fuzzy DL-Programs

Informally, a fuzzy dl-program consists of a fuzzy description logic knowledge base L and a generalized normal fuzzy program P, which may contain queries to L. In such a query, it is asked whether a concept or a role assertion logically follows from L or not (see [4] for more background and examples of such queries). Formally, a *dl-query* $Q(\mathbf{t})$ is either (a) of the form $C(t)$, where C is a concept, and t is a term, or (b) of the form $R(t_1, t_2)$, where R is a role, and t_1 and t_2 are terms. A *dl-atom* has the form $DL[S_1 \uplus p_1, \ldots, S_m \uplus p_m; Q](\mathbf{t})$, where each S_i is an atomic concept or a role, p_i is a unary resp. binary predicate symbol, $Q(\mathbf{t})$ is a dl-query, and $m \geqslant 0$. We call $p_1, \ldots, p_m$ its *input predicate symbols*. Intuitively, $S_i \uplus p_i$ encodes that the truth value of every $S_i(\mathbf{e})$ is at least the truth value of $p_i(\mathbf{e})$, where $\mathbf{e}$ is a constant (resp., pair of constants) from Φ when S_i is a concept (resp., role) (and thus p_i is a unary (resp., binary) predicate symbol). A *fuzzy dl-rule* r is of the form (8), where any b_i in the body of r may be a dl-atom. A *fuzzy dl-program* $KB = (L, P)$ consists of a satisfiable fuzzy description logic knowledge base L and a finite set of fuzzy dl-rules P. *Substitutions*, *ground substitutions*, *ground terms*, *ground atoms*, etc., are defined as usual. We denote by $ground(P)$ the set of all ground instances of fuzzy dl-rules in P relative to Φ.

Example 5.1 (Shopping Agent cont'd) A fuzzy dl-program $KB = (L, P)$ is given by the fuzzy description logic knowledge base L in Example 4.1, and the set of fuzzy dl-rules P, which contains only the following fuzzy dl-rule encoding the buyer's request (where $\otimes$ is the Gödel conjunction strategy, that is, $x \otimes y = \min(x, y)$):

$$\begin{aligned} query(x) \quad \leftarrow_{\otimes} \quad & SportyCar(x) \wedge_{\otimes} hasInvoice(x, y_1) \wedge_{\otimes} hasHP(x, y_2) \wedge_{\otimes} \\ & DL[LeqAbout22000](y_1) \wedge_{\otimes} DL[Around150HP](y_2) \geqslant 1\,. \end{aligned}$$

5.3 Models of Fuzzy DL-Programs

We first define fuzzy interpretations, and the semantics of dl-queries and the truth of fuzzy dl-rules and dl-programs in such interpretations. In the sequel, let $KB = (L, P)$ be a (fully general) fuzzy dl-program.

We use HB_Φ (resp., HU_Φ) to denote the Herbrand base (resp., universe) over Φ. In the sequel, we assume that HB_Φ is nonempty. A *fuzzy interpretation* I is a mapping $I\colon HB_\Phi \rightarrow [0, 1]$. We write $\boldsymbol{HB}_\Phi$ to denote the fuzzy interpretation I such that $I(a) = 1$ for all $a \in HB_\Phi$. For fuzzy interpretations I and J, we write $I \subseteq J$ iff $I(a) \leqslant J(a)$ for all $a \in HB_\Phi$, and we define the *intersection* of I and J, denoted $I \cap J$, by $(I \cap J)(a) = \min(I(a), J(a))$ for all $a \in HB_\Phi$. Note that $I \subseteq \boldsymbol{HB}_\Phi$ for all fuzzy interpretations I. The truth value of $a \in HB_\Phi$ in I under L, denoted $I_L(a)$, is defined as $I(a)$. The truth value of a ground dl-atom $a = DL[S_1 \uplus p_1, \ldots, S_m \uplus p_m; Q](\mathbf{c})$ in I under L, denoted $I_L(a)$, is the supremum of v subject to $L \cup \bigcup_{i=1}^{m} A_i(I) \models Q(\mathbf{c}) \geqslant v$ and $v \in [0, 1]$, where

$$A_i(I) \quad = \quad \{S_i(\mathbf{e}) \geqslant I(p_i(\mathbf{e})) \mid I(p_i(\mathbf{e})) > 0,\ p_i(\mathbf{e}) \in HB_\Phi\}\,.$$

We say I is a *model* of a ground fuzzy dl-rule r of form (8) under L, denoted $I \models_L r$, iff

$$I_L(a) \quad \geqslant \quad \begin{cases} I_L(b_1) \otimes_1 I_L(b_2) \otimes_2 \cdots \otimes_{k-1} I_L(b_k) \otimes_k & \text{if } m \geqslant 1\,; \\ \qquad \ominus_{k+1} I_L(b_{k+1}) \otimes_{k+1} \cdots \otimes_{m-1} \ominus_m I_L(b_m) \otimes_0 v & \\ v & \text{otherwise.} \end{cases}$$

We say I is a *model* of $KB = (L, P)$, denoted $I \models KB$, iff $I \models_L r$ for all $r \in ground(P)$.

5.4 Semantics of Positive Fuzzy DL-Programs

We now define the semantics of positive fuzzy dl-programs, which are fuzzy dl-programs without default negation: A fuzzy dl-program $KB = (L, P)$ is *positive* iff P is "not"-free.

For ordinary positive programs, as well as positive dl-programs KB, the intersection of two models of KB is also a model of KB. A similar result holds for positive fuzzy dl-programs KB. Hence, every positive fuzzy dl-program KB has as its *canonical model* a unique least model, denoted M_{KB}, which is contained in every model of KB.

Example 5.2 (Shopping Agent cont'd) The fuzzy dl-program $KB = (L, P)$ of Example 5.1 is positive, and its minimal model M_{KB} is given as follows:

$$M_{KB}(\mathit{query}(\mathit{MazdaMX5Miata})) \;=\; 0.36\,, \quad M_{KB}(\mathit{query}(\mathit{MitsubishiES})) \;=\; 0.32\,,$$

and all other ground instances of $\mathit{query}(x)$ have the truth value 0 under M_{KB}.

5.5 Semantics of Stratified Fuzzy DL-Programs

We next define stratified fuzzy dl-programs, which are informally composed of hierarchic layers of positive fuzzy dl-programs that are linked via default negation. Like for ordinary stratified programs, as well as stratified dl-programs, a minimal model can be defined by a finite number of iterative least models, which naturally describes as the *canonical model* the semantics of stratified fuzzy dl-programs.

For any fuzzy dl-program $KB = (L, P)$, let DL_P denote the set of all ground dl-atoms that occur in $\mathit{ground}(P)$. An *input atom* of $a \in DL_P$ is a ground atom with an input predicate of a and constant symbols in Φ. A *stratification of* $KB = (L, P)$ (*with respect to* DL_P) is a mapping $\lambda \colon HB_\Phi \cup DL_P \to \{0, 1, \ldots, k\}$ such that

(i) $\lambda(H(r)) \geqslant \lambda(a)$ (resp., $\lambda(H(r)) > \lambda(a)$) for each $r \in \mathit{ground}(P)$ and $a \in B^+(r)$ (resp., $a \in B^-(r)$), and

(ii) $\lambda(a) \geqslant \lambda(a')$ for each input atom a' of each $a \in DL_P$,

where $k \geqslant 0$ is the *length* of λ. For $i \in \{0, \ldots, k\}$, we define $KB_i = (L, P_i) = (L, \{r \in \mathit{ground}(P) \mid \lambda(H(r)) = i\})$, and we define HB_{P_i} (resp., $HB^\star_{P_i}$) as the set of all $a \in HB_\Phi$ such that $\lambda(a) = i$ (resp., $\lambda(a) \leqslant i$).

A fuzzy dl-program $KB = (L, P)$ is *stratified* iff it has a stratification λ of some length $k \geqslant 0$. We define its iterative least models $M_i \subseteq \boldsymbol{HB}_\Phi$ with $i \in \{0, \ldots, k\}$ by:

(i) M_0 is the least model of KB_0;

(ii) if $i > 0$, then M_i is the least model of KB_i such that $M_i | HB^\star_{P_{i-1}} = M_{i-1} | HB^\star_{P_{i-1}}$, where $M_i | HB^\star_{P_{i-1}}$ and $M_{i-1} | HB^\star_{P_{i-1}}$ denote the restrictions of the mappings M_i and M_{i-1} to $HB^\star_{P_{i-1}}$, respectively.

Then, M_{KB} denotes M_k. Note that M_{KB} is well-defined, since it does not depend on a particular stratification λ. Furthermore, M_{KB} is in fact a minimal model of KB.

6 Probabilistic Fuzzy Description Logic Programs

In this section, we introduce probabilistic fuzzy dl-programs as a combination of stratified fuzzy dl-programs with Poole's independent choice logic (ICL) [21]. This will allow us to express probabilistic rules. Poole's ICL is based on ordinary acyclic logic programs P under different "atomic choices", where each atomic choice along with P produces a first-order model, and one then obtains a probability distribution on the set of first-order models by placing a probability distribution on the different atomic choices. Here, we use stratified fuzzy dl-programs rather than ordinary acyclic logic programs, and thus we define a probability distribution on a set of fuzzy interpretations. In other words, we define a probability distribution on a set of rankings on the Herbrand base.

6.1 Syntax

We now define the syntax of probabilistic fuzzy dl-programs and probabilistic queries addressed to them. We first introduce fuzzy formulas, query constraints, and probabilistic formulas, and we define choice spaces and probabilities on choice spaces.

We define *fuzzy formulas* by induction as follows. The propositional constants *false* and *true*, denoted $\bot$ and $\top$, respectively, and all atoms $p(t_1, \ldots, t_k)$ are fuzzy formulas. If ϕ and ψ are fuzzy formulas, and $\otimes$, $\oplus$, $\rhd$, and $\ominus$ are conjunction, disjunction, implication, resp. negation strategies, then $(\phi \wedge_\otimes \psi)$, $(\phi \vee_\oplus \psi)$, $(\phi \Rightarrow_\rhd \psi)$, and $\neg_\ominus \phi$ are also fuzzy formulas. A *query constraint* has the form $(\phi\,\theta\, r)[l, u]$ or $(\mathbf{E}[\phi])[l, u]$ with $\theta \in \{\geqslant, >, <, \leqslant\}$, $r, l, u \in [0, 1]$, and fuzzy formulas ϕ. Informally, the former asks for the interval of the probability that the truth value v of ϕ satisfies $v\,\theta\, r$, while the latter asks for the interval of the expected truth value of ϕ. We define *probabilistic formulas* inductively as follows. Each query constraint is a probabilistic formula. If F and G are probabilistic formulas, then also $\neg F$ and $(F \wedge G)$. We use $(F \vee G)$ and $(F \Rightarrow G)$ to abbreviate $\neg(\neg F \wedge \neg G)$ and $\neg(F \wedge \neg G)$, respectively, and eliminate parentheses as usual.

A *choice space* C is a set of pairwise disjoint and nonempty sets $A \subseteq HB_\Phi$. Any $A \in C$ is an *alternative* of C and any $a \in A$ an *atomic choice* of C. Intuitively, every $A \in C$ represents a random variable and every $a \in A$ one of its possible values. A *total choice* of C is a set $B \subseteq HB_\Phi$ such that $|B \cap A| = 1$ for all $A \in C$. Intuitively, every total choice B of C represents an assignment of values to all the random variables. A *probability* μ on a choice space C is a probability function on the set of all total choices of C. Intuitively, every probability μ is a probability distribution over the set of all variable assignments. Since C and all its alternatives are finite, μ can be defined by (i) a mapping $\mu\colon \bigcup C \to [0, 1]$ such that $\sum_{a \in A} \mu(a) = 1$ for all $A \in C$, and (ii) $\mu(B) = \Pi_{b \in B}\mu(b)$ for all total choices B of C. Intuitively, (i) defines a probability over the values of each random variable of C, and (ii) assumes independence between the random variables.

A *probabilistic fuzzy dl-program* $KB = (L, P, C, \mu)$ consists of a stratified fuzzy dl-program (L, P), a choice space C such that (i) $\bigcup C \subseteq HB_\Phi$ and (ii) no atomic choice in C coincides with the head of any fuzzy dl-rule in $ground(P)$, and a probability μ on C. Intuitively, since the total choices of C select subsets of P, and μ is a probability distribution on the total choices of C, every probabilistic fuzzy dl-program is the compact representation of a probability distribution on a finite set of stratified fuzzy dl-programs. A *probabilistic query* to KB has the form $\exists F$, or $\exists(\alpha\,\theta\, r)[L, U]$, or $\exists(\mathbf{E}[\alpha])[L, U]$, where F is a probabilistic formula, α is a fuzzy formulas, $r \in [0, 1]$, and L, U are variables.

Example 6.1 (Shopping Agent cont'd) A probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$ is given by L of Example 4.1, the following set of fuzzy dl-rules P, which model the query reformulation and retrieval steps using ontology mapping rules:

$$\begin{aligned}
query(x) &\leftarrow_\otimes & & SportsCar(x) \wedge_\otimes hasPrize(x, y_1) \wedge_\otimes hasPower(x, y_2) \wedge_\otimes \\
& & & DL[LeqAbout22000](y_1) \wedge_\otimes DL[Around150HP](y_2) \geqslant 1\,, && (9)\\
SportsCar(x) &\leftarrow_\otimes & & DL[SportyCar](x) \wedge_\otimes sc_{pos} \geqslant 0.9\,, && (10)\\
hasPrize(x) &\leftarrow_\otimes & & DL[hasInvoice](x) \wedge_\otimes hi_{pos} \geqslant 0.8\,, && (11)\\
hasPower(x) &\leftarrow_\otimes & & DL[hasHP](x) \wedge_\otimes hhp_{pos} \geqslant 0.8\,, && (12)
\end{aligned}$$

the choice space $C = \{\{sc_{pos}, sc_{neg}\}, \{hi_{pos}, hi_{neg}\}, \{hhp_{pos}, hhp_{neg}\}\}$, and the probability distribution μ, which is given by the following probabilities for the atomic choices (and then extended to all total choices by assuming independence):

$$\begin{array}{lll} \mu(sc_{pos}) = 0.91\,, & \mu(sc_{neg}) = 0.09\,, & \mu(hi_{pos}) = 0.78\,, \\ \mu(hi_{neg}) = 0.22\,, & \mu(hhp_{pos}) = 0.83\,, & \mu(hhp_{neg}) = 0.17\,. \end{array}$$

Rule 9 is the buyer's request, but in a "different" terminology than the one of the car selling site. Rules 10–12 are so-called ontology alignment mapping rules. For example, rule 10 states that the predicate "SportsCar" of the buyer's terminology refers to the concept "SportyCar" of the selected side, with probability 0.91. Such mapping rules can be automatically built by relying on ontology alignment tools, such as oMap [27, 28], whose main purpose is to find relations among the concepts and roles of two different ontologies. oMap is particularly suited for our case, as it is based on a probabilistic model, and thus the mappings have a probabilistic reading (see also [20]).

6.2 Semantics

A *world* I is a fuzzy interpretation over HB_Φ. We denote by $\mathcal{I}_\Phi$ the set of all worlds over Φ. A *variable assignment* σ maps each $X \in \mathcal{X}$ to some $t \in HU_\Phi$. It is extended to all terms by $\sigma(c) = c$ for all constant symbols c from Φ. The *truth value* of fuzzy formulas ϕ in I under σ, denoted $I_\sigma(\phi)$ (or $I(\phi)$ when ϕ is ground), is inductively defined by (1) $I_\sigma(\phi \wedge_\otimes \psi) = I_\sigma(\phi) \otimes I_\sigma(\psi)$, (2) $I_\sigma(\phi \vee_\oplus \psi) = I_\sigma(\phi) \oplus I_\sigma(\psi)$, (3) $I_\sigma(\phi \Rightarrow_\rhd \psi) = I_\sigma(\phi) \rhd I_\sigma(\psi)$, and (4) $I_\sigma(\neg_\ominus \phi) = \ominus\, I_\sigma(\phi)$,

A *probabilistic interpretation* Pr is a probability function on $\mathcal{I}_\Phi$ (that is, a mapping $Pr\colon \mathcal{I}_\Phi \to [0,1]$ such that (i) the set of all $I \in \mathcal{I}_\Phi$ with $Pr(I) > 0$ is denumerable, and (ii) all $Pr(I)$ with $I \in \mathcal{I}_\Phi$ sum up to 1). The *probability* of a formula $\phi\,\theta\,r$ in Pr under a variable assignment σ, denoted $Pr_\sigma(\phi\,\theta\,r)$ (or $Pr(\phi\,\theta\,r)$ when ϕ is ground), is the sum of all $Pr(I)$ such that $I \in \mathcal{I}_\Phi$ and $I_\sigma(\phi)\,\theta\,r$. The *expected truth value* of a formula ϕ under Pr and σ, denoted $\mathbf{E}_{Pr,\sigma}[\phi]$, is the sum of all $Pr(I) \cdot I_\sigma(\phi)$ such that $I \in \mathcal{I}_\Phi$. The *truth* of probabilistic formulas F in Pr under σ, denoted $Pr \models_\sigma F$, is inductively defined by (1) $Pr \models_\sigma (\phi\,\theta\,r)[l,u]$ iff $Pr_\sigma(\phi\,\theta\,r) \in [l,u]$, (2) $Pr \models_\sigma (\mathbf{E}[\phi])[l,u]$ iff $\mathbf{E}_{Pr,\sigma}[\phi] \in [l,u]$, (3) $Pr \models_\sigma \neg F$ iff not $Pr \models_\sigma F$, and (4) $Pr \models_\sigma (F \wedge G)$ iff $Pr \models_\sigma F$ and $Pr \models_\sigma G$.

A probabilistic interpretation Pr is a *model* of a probabilistic formula F iff $Pr \models_\sigma F$ for every variable assignment σ. We say Pr is the *canonical model* of a probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$ iff every world $I \in \mathcal{I}_\Phi$ with $Pr(I) > 0$ is the canonical model of $(L, P \cup \{p \leftarrow \mid p \in B\})$ for some total choice B of C such that $Pr(I) = \mu(B)$. Notice that every KB has a unique canonical model Pr. We say F is a *consequence* of KB, denoted $KB \mathrel{\|\!\!\sim} F$, iff the canonical model of KB is also a model of F. A query constraint $(\phi\,\theta\,r)[l,u]$ (resp., $(\mathbf{E}[\phi])[l,u]$) is a *tight consequence* of KB, denoted $KB \mathrel{\|\!\!\sim}_{tight} (\phi\,\theta r)[l,u]$ (resp., $KB \mathrel{\|\!\!\sim}_{tight} (\mathbf{E}[\phi])[l,u]$), iff l (resp., u) is the infimum (resp., supremum) of $Pr_\sigma(\phi\,\theta\,r)$ (resp., $\mathbf{E}_{Pr,\sigma}[\phi]$) subject to the canonical model Pr of KB and all σ. A *correct answer* to $\exists F$ is a substitution σ such that $F\sigma$ is a consequence of KB. A *tight answer* to $\exists(\alpha\,\theta\,r)[L,U]$ (resp., $\exists(\mathbf{E}[\alpha])[L,U]$) is a substitution σ such that $(\alpha\,\theta\,r)[L,U]\sigma$ (resp., $(\mathbf{E}[\alpha])[L,U]\sigma$) is a tight consequence of KB.

Example 6.2 (Shopping Agent cont'd) The following are some tight consequences of the probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$ in Example 6.1:

$$(\mathbf{E}[query(MazdaMX5Miata)])[0.21, 0.21]\,, \quad (\mathbf{E}[query(MitsubishiES)])[0.19, 0.19]\,.$$

So, the shopping agent ranks the $MazdaMX5Miata$ first with degree 0.21 ($= 0.36 \cdot 0.91 \cdot 0.78 \cdot 0.83$) and the $MitsubishiES$ second with degree 0.19 ($= 0.32 \cdot 0.91 \cdot 0.78 \cdot 0.83$).

7 Query Processing in Probabilistic Fuzzy DL-Programs

The canonical model of an ordinary positive resp. stratified normal program KB, as well as of a positive resp. stratified dl-program KB has a well-known fixpoint characterization in terms of an immediate consequence operator T_{KB}, which generalizes to fuzzy dl-programs. This can be exploited for a bottom-up computation of the canonical model of a positive resp. stratified fuzzy dl-program, and thus for query processing in probabilistic fuzzy dl-programs.

7.1 Positive Fuzzy DL-Programs

We first define the immediate consequence operator for fuzzy dl-programs. For any fuzzy dl-program $KB=(L,P)$, we define the operator T_{KB} on the subsets of $\boldsymbol{HB}_\Phi$ as follows. For every $I \subseteq \boldsymbol{HB}_\Phi$ and $a \in HB_\Phi$, let $T_{KB}(I)(a)$ be the maximum of v subject to $r \in ground(P)$, $H(r)=a$, and v being the truth value of r's body under I and L. If there is no such rule r, then $T_{KB}(I)(a)=0$.

The following lemma shows that for positive fuzzy dl-programs KB, the operator T_{KB} is monotonic, that is, $I \subseteq I' \subseteq \boldsymbol{HB}_\Phi$ implies $T_{KB}(I) \subseteq T_{KB}(I')$. This result follows immediately from the fact that every dl-atom and every conjunction strategy in $ground(P)$ is monotonic.

Lemma 7.1 *Let $KB=(L,P)$ be a positive fuzzy dl-program. Then, the operator T_{KB} is monotonic.*

The next result gives a characterization of the pre-fixpoints of T_{KB}, which coincide with the models of KB. We recall here that $I \subseteq \boldsymbol{HB}_\Phi$ is a pre-fixpoint of T_{KB} iff $T_{KB}(I) \subseteq I$.

Proposition 7.2 *Let $KB=(L,P)$ be a positive fuzzy dl-program. Then, $I \subseteq \boldsymbol{HB}_\Phi$ is a pre-fixpoint of T_{KB} iff I is a model of KB.*

Since every monotonic operator has a least fixpoint, which coincides with its least pre-fixpoint, we immediately obtain as a corollary that also T_{KB} has a least fixpoint, denoted $\mathit{lfp}(T_{KB})$, and that this least fixpoint is given by the least model of KB.

The next result shows that the least fixpoint of T_{KB} can be computed by a finite fixpoint iteration, if KB is *closed* under a finite set of truth values $TV \subseteq [0,1]$ (with $|TV| \geqslant 2$), which means that (i) each datatype predicate in KB is interpreted by a mapping to TV, (ii) each fuzzy modifier m in KB is interpreted by a mapping $f_m\colon TV \rightarrow TV$, (iii) each truth value in KB is from TV, and (iv) each combination strategy in KB is closed under TV (note that the combination strategies of Łukasiewicz, Gödel, and Zadeh Logic are closed under every $TV_n=\{0,\frac{1}{n},\ldots,\frac{n}{n}\}$ with $n>0$). Note that for every $I \subseteq \boldsymbol{HB}_\Phi$, we define $T^i_{KB}(I)=I$, if $i=0$, and $T^i_{KB}(I)=T_{KB}(T^{i-1}_{KB}(I))$, if $i>0$.

Theorem 7.3 *Let $KB=(L,P)$ be a positive fuzzy dl-program that is closed under a finite set of truth values $TV \subseteq [0,1]$ (with $|TV| \geqslant 2$). Then, $\mathit{lfp}(T_{KB})=M_{KB}$. Furthermore, $\mathit{lfp}(T_{KB})=\bigcup_{i=0}^{n} T^i_{KB}(\emptyset)=T^n_{KB}(\emptyset)$, for some $n \geqslant 0$.*

7.2 Stratified Fuzzy DL-Programs

We finally describe a sequence of finite fixpoint iterations for stratified fuzzy dl-programs. Using Theorem 7.3, we can characterize the answer set M_{KB} of a stratified fuzzy dl-program $KB=(L,P)$ by a sequence of finite fixpoint iterations along a stratification of KB as follows. Let the operator $\widehat{T}^{i}_{KB}$ on interpretations $I \subseteq \boldsymbol{HB}_{\Phi}$ be defined by $\widehat{T}^{i}_{KB}(I) = T^{i}_{KB}(I) \cup I$, for all $i \geqslant 0$. Here, $I \cup J$ for $I, J \subseteq \boldsymbol{HB}_{\Phi}$ denotes the *union* of I and J, which is defined by $(I \cup J)(a) = \max(I(a), J(a))$ for all $a \in HB_{\Phi}$.

Theorem 7.4 *Let $KB = (L, P)$ be a fuzzy dl-program with stratification λ of length $k \geqslant 0$. Suppose that KB is closed under a finite set of truth values $TV \subseteq [0,1]$ (with $|TV| \geqslant 2$). Let $M_i \subseteq \boldsymbol{HB}_{\Phi}$, $i \in \{-1, 0, \ldots, k\}$, by $M_{-1} = \emptyset$, and $M_i = \widehat{T}^{n_i}_{KB_i}(M_{i-1})$ for each $i \geqslant 0$, where $n_i \geqslant 0$ such that $\widehat{T}^{n_i}_{KB_i}(M_{i-1}) = \widehat{T}^{n_i+1}_{KB_i}(M_{i-1})$. Then, $M_k = M_{KB}$.*

7.3 Probabilistic Fuzzy DL-Programs

Fig. 3 shows Algorithm canonical_model, which computes the canonical model Pr of a given probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$. This algorithm is essentially based on a reduction to computing the canonical model of stratified fuzzy dl-programs (see line 2), which can be done using the above finite sequence of finite fixpoint iterations.

Algorithm canonical_model

Input: probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$.
Output: canonical model Pr of KB (represented as $\{(I, Pr(I)) \mid I \in \mathcal{I}_{\Phi},\ Pr(I) > 0\}$).

```
1. for every total choice $B$ of $C$ do begin
2.   compute the canonical model $I$ of the stratified fuzzy dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$;
3.   $Pr(I) := \mu(B)$;
4. end;
5. return $Pr$.
```

Figure 3: Algorithm canonical_model

Algorithm tight_answer in Fig. 4 computes the tight answer $\theta = \{L/l, U/u\}$ for a given probabilistic query $Q = \exists(\alpha\,\theta\,r)[L, U]$ (resp., $Q = \exists(\mathbf{E}[\alpha])[L, U]$) to a given probabilistic fuzzy dl-program KB. The algorithm first computes the canonical model of KB in line 1 and then the tight answer $\theta = \{L/l,\, U/u\}$ in lines 2–8.

8 Tractability Results

Deciding whether a knowledge base in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) is satisfiable is complete for the complexity class EXP (resp., NEXP, assuming unary number encoding; see [12] and the NEXP-hardness proof for $\mathcal{ALCQI}$ in [29], which implies the NEXP-hardness of $\mathcal{SHOIN}(\mathbf{D})$). Recall that EXP (resp., NEXP) is the class of all decision problems that can be solved in exponential time on a deterministic (resp., nondeterministic) Turing machine. Hence, also deciding whether a more general fuzzy knowledge base in fuzzy $\mathcal{SHIF}(\mathbf{D})$ (resp., fuzzy $\mathcal{SHOIN}(\mathbf{D})$) is satisfiable is hard for EXP (resp., NEXP). Since the latter can be done via dl-queries in probabilistic fuzzy dl-programs, it thus follows that query processing from probabilistic fuzzy dl-programs is in general intractable.

```
Algorithm tight_answer
Input: probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$ and
       probabilistic query $Q = \exists(\alpha\,\theta\,r)[L, U]$ (resp., $Q = \exists(\mathbf{E}[\alpha])[L, U]$).
Output: tight answer $\theta = \{L/l, U/u\}$ for $Q$ to $KB$.
1. $Pr :=$ canonical_model($KB$);
2. $l := 1$;
3. $u := 0$;
4. for every ground instance $\alpha'$ of $\alpha$ do begin
5.    $l := \min(l, Pr(\alpha'\,\theta\,r))$; (resp., $l := \min(l, \mathbf{E}[\alpha'])$;)
6.    $u := \max(u, Pr(\alpha'\,\theta\,r))$; (resp., $u := \max(u, \mathbf{E}[\alpha'])$;)
7. end;
8. return $\theta = \{L/l, U/u\}$.
```

Figure 4: Algorithm tight_answer

In this section, we describe a special class of stratified probabilistic fuzzy dl-programs KB for which query processing has a polynomial data complexity. These programs are defined relative to *fuzzy DL-Lite* [26], which is a fuzzy generalization of the description logic *DL-Lite* [3]. By [26] (resp., [3]), deciding whether a knowledge base in *DL-Lite* (resp., *fuzzy DL-Lite*) is satisfiable can be done in polynomial time, and conjunctive query processing from a knowledge base in *DL-Lite* (resp., *fuzzy DL-Lite*) has a polynomial data complexity.

We first recall *DL-Lite* and *fuzzy DL-Lite*. Let $\mathbf{A}$, $\mathbf{R}_A$, and $\mathbf{I}$ be pairwise disjoint sets of atomic concepts, abstract roles, and individuals, respectively. A *basic concept in fuzzy DL-Lite* is either an atomic concept from $\mathbf{A}$ or an exists restriction on roles $\exists R.\top$ (abbreviated as $\exists R$), where $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$. A *literal in DL-Lite* is either a basic concept b or the negation of a basic concept $\neg b$. *Concepts in DL-Lite* are defined by induction as follows. Every basic concept in *DL-Lite* is a concept in *DL-Lite*. If b is a basic concept in *DL-Lite*, and ϕ_1 and ϕ_2 are concepts in *DL-Lite*, then $\neg b$ and $\phi_1 \sqcap \phi_2$ are also concepts in *DL-Lite*. An *axiom in DL-Lite* is either (1) a concept inclusion axiom $b \sqsubseteq \psi$, where b is a basic concept in *DL-Lite*, and ϕ is a concept in *DL-Lite*, or (2) a *functionality axiom* $(\mathsf{funct}\ R)$, where $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, or (3) a concept assertion axiom $b(a)$, where b is a basic concept in *DL-Lite* and $a \in \mathbf{I}$, or (4) a role assertion axiom $R(a, c)$, where $R \in \mathbf{R}_A$ and $a, c \in \mathbf{I}$. A *fuzzy concept* (resp., *role*) *assertion axiom* is of the form $b(a) \geqslant n$ (resp., $R(a, c) \geqslant n$), where $b(a)$ (resp., $R(a, c)$) is a concept (resp., role) assertion axiom in *DL-Lite*, and $n \in (0, 1]$. A *fuzzy axiom in DL-Lite* is either a fuzzy concept assertion axiom or a fuzzy role assertion axiom. A *fuzzy knowledge base in DL-Lite* L is a finite set of concept inclusion, functionality, fuzzy concept assertion, and fuzzy role assertion axioms in *DL-Lite*. Like in [26], we here assume that L is interpreted using the combination strategies of Zadeh Logic.

We are now ready to define probabilistic fuzzy dl-programs in *DL-Lite* as follows. We say that a fuzzy dl-program $KB = (L, P)$ is defined in *DL-Lite* iff (i) KB is closed under $TV_n = \{0, \frac{1}{n}, \ldots, \frac{n}{n}\}$ for some $n > 0$, (ii) KB is stratified, (iii) L is defined in *DL-Lite*, and (iv) P contains only dl-queries of the form $DL[\lambda; Q](\mathbf{t})$, where Q is either a concept or a role. Note that we assume that the above n is an explicit part of KB. We say that a probabilistic fuzzy dl-program $KB = (L, P, C, \mu)$ is in *DL-Lite* iff $(L, P \cup \{p \leftarrow \mid p \in B\})$ is in *DL-Lite* for every total choice B of C. The following theorem shows that for probabilistic fuzzy dl-programs in *DL-Lite* $KB = (L, P, C, \mu)$, computing the tight answer to a ground probabilistic query has a polynomial data complexity.

Theorem 8.1 *Let $KB = (L, P, C, \mu)$ be a probabilistic fuzzy dl-program in* DL-Lite, *and let $Q = \exists(\alpha\,\theta$*

$r)[L, U]$ (resp., $Q = \exists(\mathbf{E}[\alpha])[L, U]$) be a ground probabilistic query. Then, computing the tight answer $\theta = \{L/l, U/u\}$ for Q to KB has a polynomial data complexity.

9 Summary and Outlook

We have presented probabilistic fuzzy dl-programs for the Semantic Web, which allow for handling both probabilistic uncertainty (especially for probabilistic ontology mapping and probabilistic data integration) and fuzzy vagueness (especially for dealing with vague concepts) in a uniform framework. We have defined important concepts related to both probabilistic uncertainty and fuzzy vagueness. We have then provided algorithms for query processing in such programs, and we have also delineated a special case where query processing has a polynomial data complexity. Finally, we have described a shopping agent example, which gives evidence of the usefulness of probabilistic fuzzy dl-programs in realistic web applications.

An interesting topic of future research is to generalize probabilistic fuzzy dl-programs by non-stratified default negations, classical negations, and disjunctions.

References

[1] T. Berners-Lee. *Weaving the Web*. Harper, San Francisco, 1999.

[2] J. Callan. Distributed information retrieval. In W. B. Croft, editor, *Advances in Information Retrieval*, pp. 127–150. Kluwer, Hingham, MA, USA, 2000.

[3] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. *DL-Lite*: Tractable description logics for ontologies. In *Proc. AAAI-2005*, pp. 602–607, 2005.

[4] T. Eiter, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Combining answer set programming with description logics for the Semantic Web. In *Proc. KR-2004*, pp. 141–151, 2004. Extended Report RR-1843-07-04, Institut für Informationssysteme, TU Wien, 2007.

[5] T. Eiter, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Well-founded semantics for description logic programs in the Semantic Web. In *Proc. RuleML-2004*, pp. 81–97, 2004.

[6] T. Eiter, G. Ianni, R. Schindlauer, H. Tompits. Effective integration of declarative rules with external evaluations for semantic-web reasoning. In *Proc. ESWC-2006*, pp. 273–287, 2006.

[7] D. Fensel, W. Wahlster, H. Lieberman, and J. Hendler, editors. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, 2002.

[8] N. Fuhr. A decision-theoretic approach to database selection in networked IR. *ACM Transactions on Information Systems*, 3(17):229–249, 1999.

[9] T. Flaminio and L. Godo. A logic for reasoning about the probability of fuzzy events. *Fuzzy Sets and Systems*, 158(6):625–638, 2007.

[10] P. Hájek. *Metamathematics of Fuzzy Logic*. Kluwer, 1998.

[11] S. Hölldobler, H.-P. Störr, and T. D. Khang. The subsumption problem of the fuzzy description logic ALC_{FH}. In *Proc. IPMU-2004*, pp. 243–250, 2004.

[12] I. Horrocks and P. F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. In *Proc. ISWC-2003*, pp. 17–29, 2003.

[13] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. Web Sem.*, 1(1):7–26, 2003.

[14] T. Lukasiewicz. Probabilistic description logic programs. In *Proc. ECSQARU-2005*, pp. 737–749, 2005. Extended version in *Int. J. Approx. Reason.*, 2007. In press.

[15] T. Lukasiewicz. Fuzzy description logic programs under the answer set semantics for the Semantic Web. In *Proc. RuleML-2006*, pp. 89–96, 2006. Extended version accepted for publication in *Fundamenta Informaticae*.

[16] T. Lukasiewicz and U. Straccia. An overview of uncertainty and vagueness in description logics for the Semantic Web. Technical Report INFSYS RR-1843-06-07, Institut für Informationssysteme, TU Wien, October 2006.

[17] H. Nottelmann and U. Straccia. Information retrieval and machine learning for probabilistic schema matching. In *Proc. CIKM-2005*, pp. 295–296, 2005.

[18] H. Nottelmann and U. Straccia. A probabilistic approach to schema matching. In *Proc. ECIR-2005*, pp. 81–95, 2005.

[19] H. Nottelmann and U. Straccia. A probabilistic, logic-based framework for automated web directory alignment. In Z. Ma, editor, *Soft Computing in Ontologies and the Semantic Web*, volume 204 of *Studies in Fuzziness and Soft Computing*, pp. 47–77. Springer, 2006.

[20] H. Nottelmann and U. Straccia. Information retrieval and machine learning for probabilistic schema matching. *Information Processing & Management*, 2007. To appear.

[21] D. Poole. The independent choice logic for modelling multiple agents under uncertainty. *Artif. Intell.*, 94(1–2):7–56, 1997.

[22] D. Poole. Logic, knowledge representation, and Bayesian decision theory. In *Proc. CL-2000*, pp. 70–86, 2000.

[23] M. E. Renda and U. Straccia. Web metasearch: Rank vs. score-based rank aggregation methods. In *Proc. SAC-2003*, pp. 841–846, 2003.

[24] U. Straccia. Towards a fuzzy description logic for the Semantic Web (preliminary report). In *Proc. ESWC-2005*, pp. 167–181, 2005.

[25] U. Straccia. A fuzzy description logic for the Semantic Web. In E. Sanchez, editor, *Fuzzy Logic and the Semantic Web*, Capturing Intelligence, chapter 4, pp. 73–90. Elsevier, 2006.

[26] U. Straccia. Fuzzy description logic programs. In *Proc. IPMU-2006*, pp. 1818–1825, 2006.

[27] U. Straccia and R. Troncy. oMAP: Combining classifiers for aligning automatically OWL ontologies. In *Proc. WISE-2005*, pp. 133–147, 2005.

[28] U. Straccia and R. Troncy. Towards distributed information retrieval in the Semantic Web. In *Proc. ESWC-2006*, pp. 378–392, 2006.

[29] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD thesis, RWTH Aachen, Germany, 2001.

[30] C. Tresp and R. Molitor. A description logic for vague knowledge. In *Proc. ECAI-1998*, pp. 361–365, 1998.

[31] W3C. OWL web ontology language overview, 2004. W3C Recommendation (10 Feb. 2004). Available at www.w3.org/TR/2004/REC-owl-features-20040210/.