A Minimal Deductive System for General Fuzzy RDF

Umberto Straccia

Istituto di Scienza e Tecnologie dell'Informazione (ISTI - CNR), Pisa, Italy
straccia@isti.cnr.it

Abstract. It is well-known that crisp RDF is not suitable to represent vague information. Fuzzy RDF variants are emerging to overcome this limitations. In this work we provide, under a very general semantics, a deductive system for a salient fragment of fuzzy RDF. We then also show how we may compute the top-k answers of the union of conjunctive queries in which answers may be scored by means of a scoring function.

1 Introduction

RDF [17] has become a quite popular Semantic Web representation formalism. The basic ingredients are triples of the form (s, p, o), such as $(tom, likes, tomato)$, stating that subject s has property p with value o.

However, under the classical semantics, RDF cannot represent vague information and, to this purpose, some *Fuzzy RDF* variants have been proposed [12,13,14,21,22]: essentially they allow to state that a triple is true to some degree, *e.g.*, $(tom, likes, tomato)$ is true to degree at least 0.9.

Our main goal of this study is to provide, under a very general semantics, a minimal deductive system for fuzzy RDF, along the lines described by [15]. The advantage is that, (i) (unlike [12,13,14,22]) we abstract from the underlying XML representation; (ii) the semantics is quite general, *i.e.* is based on a t-norm [9]; (iii) we get a clear insight of the supported inference mechanism; and (iv) we concentrate on the main ingredients of RDF from a reasoning point of view. We then also address the query answering problem and show how effectively we may compute the top-k answers of the union of conjunctive queries in which answers may be scored by means of a scoring function.

Related work: The most general work so far and closest to our work is [14], to which respect we provide additionally a more general semantics, correctness and completeness and complexity results, add the notion of top-k answers of the union of conjunctive queries in which answers may be scored by means of a scoring function, and show how to compute the top-k answers. Another related work is [21], which allows to annotate triples with truth values taken from a finite partial order, while we rely on $[0, 1]$ instead[1]. However, we provide some desired inference capabilities not provided by [21], *e.g.*, from "a sport car is a fast car to degree 0.8" and "a fast car is an expensive car to degree

[1] But we can extend the truth space to other truth-spaces as well, provided that we extend the t-norm and residuated implication accordingly [9].

A. Polleres and T. Swift (Eds.): RR 2009, LNCS 5837, pp. 166–181, 2009.

0.9" we may infer that "a sport car is an expensive car to degree 0.72" (under product t-norm). Essentially, [21] does not provide a truth combination function to propagate the truth in such inferences, while we consider a t-norm instead. Additionally, as for [14], the top-k retrieval problem for the union of conjunctive queries is not addressed.

In the next section, we recall the main aspects of classical RDF as described in [15], which we extend then to the fuzzy case.

2 Preliminaries

For the sake of our purposes, we will rely on a minimal, but significant RDF fragment, called ρdf [15], that covers the essential features of RDF. According to [15], ρdf (read rho-df, the ρ from restricted rdf) is defined as the following subset of the RDFS vocabulary:

$$\rho\text{df} = \{\mathsf{sp}, \mathsf{sc}, \mathsf{type}, \mathsf{dom}, \mathsf{range}\} .$$

Informally, the meaning of a triple (s, p, o) with $p \in \rho$df is:

- (p, sp, q) means that property p is a *sub property* of property q;
- (c, sc, d) means that class c is a *sub class* of class d;
- (a, type, b) means that a is of *type* b;
- (p, dom, c) means that the *domain* of property p is c;
- (p, range, c) means that the *range* of property p is c.

Syntax. Assume pairwise disjoint alphabets $\mathbf{U}$ (*RDF URI references*), $\mathbf{B}$ (*Blank nodes*), and $\mathbf{L}$ (*Literals*). Through the paper we assume $\mathbf{U}, \mathbf{B}$, and $\mathbf{L}$ fixed, and for simplicity we will denote unions of these sets simply concatenating their names. We call elements in $\mathbf{UBL}$ *terms* (denoted t), and elements in $\mathbf{B}$ *variables* (denoted x)[2].

An *RDF triple* (or *RDF atom*) is a triple $(s, p, o) \in \mathbf{UBL} \times \mathbf{U} \times \mathbf{UBL}$. In this tuple, s is the *subject*, p is the *predicate*, and o is the *object*. An *RDF graph* (or simply a graph, or *RDF Knowledge Base*) is a set of RDF triples τ. A subgraph is a subset of a graph. The *universe* of a graph G, denoted by $universe(G)$ is the set of elements in $\mathbf{UBL}$ that occur in the triples of G. The *vocabulary* of G, denoted by $voc(G)$ is the set $universe(G) \cap \mathbf{UL}$. A graph is *ground* if it has no blank nodes, *i.e.* variables.

In what follows we will need some technical notions. A *variable assignment* is a function $\mu : \mathbf{UBL} \to \mathbf{UBL}$ preserving URIs and literals, *i.e.*, $\mu(t) = t$, for all $t \in \mathbf{UL}$. Given a graph G, we define $\mu(G) = \{(\mu(s), \mu(p), \mu(o)) \mid (s, p, o) \in G\}$. We speak of a variable assignment μ from G_1 to G_2, and write $\mu : G_1 \to G_2$, if μ is such that $\mu(G_1) \subseteq G_2$.

Semantics. An *RDF interpretation* $\mathcal{I}$ over a vocabulary V is a tuple

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle ,$$

where $\Delta_R, \Delta_P, \Delta_C, \Delta_L$ are the interpretations domains of $\mathcal{I}$, and $P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}}$ are the interpretation functions of $\mathcal{I}$. They have to satisfy:

1. Δ_R is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. Δ_P is a set of property names (not necessarily disjoint from Δ_R);

[2] All symbols may have upper or lower script.

3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of Δ_R identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, Δ_L contains all plain literals in $\mathbf{L} \cap V$;
5. $P[\![\cdot]\!]$ maps each property name $p \in \Delta_P$ into a subset $P[\![p]\!] \subseteq \Delta_R \times \Delta_R$, *i.e.* assigns an extension to each property name;
6. $C[\![\cdot]\!]$ maps each class $c \in \Delta_C$ into a subset $C[\![c]\!] \subseteq \Delta_R$, *i.e.* assigns a set of resources to every resource denoting a class;
7. $\cdot^{\mathcal{I}}$ maps each $t \in \mathbf{UL} \cap V$ into a value $t^{\mathcal{I}} \in \Delta_R \cup \Delta_P$, *i.e.* assigns a resource or a property name to each element of $\mathbf{UL}$ in V, and such that $\cdot^{\mathcal{I}}$ is the identity for plain literals and assigns an element in Δ_R to elements in $\mathbf{L}$;
8. $\cdot^{\mathcal{I}}$ maps each variable $x \in \mathbf{B}$ into a value $x^{\mathcal{I}} \in \Delta_R$, *i.e.* assigns a resource to each variable in $\mathbf{B}$.

The notion entailment is defined using the idea of *satisfaction* of a graph under certain interpretation. Intuitively a ground triple (s, p, o) in an RDF graph G will be true under the interpretation $\mathcal{I}$ if p is interpreted as a property name, s and o are interpreted as resources, and the interpretation of the pair (s, o) belongs to the extension of the property assigned to p.

In RDF, blank nodes, *i.e.* variables, work as existential variables. Intuitively the triple (x, p, o) with $x \in \mathbf{B}$ would be true under $\mathcal{I}$ if there exists a resource s such that (s, p, o) is true under $\mathcal{I}$.

Now, let G be a graph. An interpretation $\mathcal{I}$ is a *model* of G, denoted $\mathcal{I} \models G$, iff $\mathcal{I}$ is an interpretation over the vocabulary ρdf $\cup\, universe(G)$ that satisfies the following conditions:

Simple:
1. for each $(s, p, o) \in G$, $p^{\mathcal{I}} \in \Delta_P$ and $(s^{\mathcal{I}}, o^{\mathcal{I}}) \in P[\![p^{\mathcal{I}}]\!]$;

Subproperty:
1. $P[\![\mathsf{sp}^{\mathcal{I}}]\!]$ is transitive over Δ_P;
2. if $(p, q) \in P[\![\mathsf{sp}^{\mathcal{I}}]\!]$ then $p, q \in \Delta_P$ and $P[\![p]\!] \subseteq P[\![q]\!]$;

Subclass:
1. $P[\![\mathsf{sc}^{\mathcal{I}}]\!]$ is transitive over Δ_C;
2. if $(c, d) \in P[\![\mathsf{sc}^{\mathcal{I}}]\!]$ then $c, d \in \Delta_C$ and $C[\![c]\!] \subseteq C[\![d]\!]$;

Typing I:
1. $x \in C[\![c]\!]$ iff $(x, c) \in P[\![\mathsf{type}^{\mathcal{I}}]\!]$;
2. if $(p, c) \in P[\![\mathsf{dom}^{\mathcal{I}}]\!]$ and $(x, y) \in P[\![p]\!]$ then $x \in C[\![c]\!]$;
3. if $(p, c) \in P[\![\mathsf{range}^{\mathcal{I}}]\!]$ and $(x, y) \in P[\![p]\!]$ then $y \in C[\![c]\!]$;

Typing II:
1. For each $\mathsf{e} \in \rho$df, $\mathsf{e}^{\mathcal{I}} \in \Delta_P$
2. if $(p, c) \in P[\![\mathsf{dom}^{\mathcal{I}}]\!]$ then $p \in \Delta_P$ and $c \in \Delta_C$
3. if $(p, c) \in P[\![\mathsf{range}^{\mathcal{I}}]\!]$ then $p \in \Delta_P$ and $c \in \Delta_C$
4. if $(x, c) \in P[\![\mathsf{type}^{\mathcal{I}}]\!]$ then $c \in \Delta_C$

We define G *entails* H under ρdf, denoted $G \models H$, iff every model under ρdf of G is also a model under ρdf of H.

Please note that in [15], $P[\![\mathsf{sp}^{\mathcal{I}}]\!]$ (resp. $C[\![\mathsf{sc}^{\mathcal{I}}]\!]$) besides being required to be transitive over Δ_P (resp. Δ_C), is also *reflexive* over Δ_P (resp. Δ_C). We omit this requirement and, thus, do not support inferences such as $G \models (a, \mathsf{sp}, a)$ and $G \models (a, \mathsf{sc}, a)$, which anyway are of marginal interest (see [15] for a more in depth discussion on this issue).

Deductive system. In what follows, we recall the sound and complete deductive system for the fragment of RDF presented in [15]. The system is arranged in groups of rules that captures the semantic conditions of models. In every rule, A, B, C, X, and Y are meta-variables representing elements in $\mathbf{UBL}$. An instantiation of a rule is a uniform replacement of the metavariables occurring in the triples of the rule by elements of $\mathbf{UBL}$, such that all the triples obtained after the replacement are well formed RDF triples. The rules are as follows:

1. Simple:

   (a) $\frac{G}{G'}$ for a map $\mu : G' \to G$ (b) $\frac{G}{G'}$ for $G' \subseteq G$

2. Subproperty:

   (a) $\frac{(A,\mathsf{sp},B),(B,\mathsf{sp},C)}{(A,\mathsf{sp},C)}$ (b) $\frac{(A,\mathsf{sp},B),(X,A,Y)}{(X,B,Y)}$

3. Subclass:

   (a) $\frac{(A,\mathsf{sc},B),(B,\mathsf{sc},C)}{(A,\mathsf{sc},C)}$ (b) $\frac{(A,\mathsf{sc},B),(X,\mathsf{type},A)}{(X,\mathsf{type},B)}$

4. Typing:

   (a) $\frac{(A,\mathsf{dom},B),(X,A,Y)}{(X,\mathsf{type},B)}$ (b) $\frac{(A,\mathsf{range},B),(X,A,Y)}{(Y,\mathsf{type},B)}$

5. Implicit Typing:

   (a) $\frac{(A,\mathsf{dom},B),(C,\mathsf{sp},A),(X,C,Y)}{(X,\mathsf{type},B)}$ (b) $\frac{(A,\mathsf{range},B),(C,\mathsf{sp},A),(X,C,Y)}{(Y,\mathsf{type},B)}$

A *proof* is defined in the usual way. Let G and H be graphs. Then $G \vdash H$ iff there is a sequence of graphs $P_1, \ldots, P_k$ with $P_1 = G$ and $P_k = H$, and for each j $(2 \leqslant j \leqslant k)$ one of the following holds:

1. there exists a map $\mu : P_j \to P_{j-1}$ (rule (1a));
2. $P_j \subseteq P_{j-1}$ (rule (1b));
3. there is an instantiation $\frac{R}{R'}$ of one of the rules (2)(5), such that $R \subseteq P_{j-1}$ and $P_j = P_{j-1} \cup R'$.

The sequence of rules used at each step (plus its instantiation or map), is called a *proof* of H from G.

Proposition 1 (Soundness and completeness [15]). *The proof system $\vdash$ is sound and complete for $\models$, that is, $G \vdash H$ iff $G \models H$.*

Let G be a ground graph and τ be a gound triple. The *closure* of G is defined as

$$cl(G) = \{\tau \mid \tau \text{ ground and } G \vdash \tau\} .$$

Note that the size of the closure of G is $\mathcal{O}(|G|^2)$ and, thus a naive method to answer whether $G \models \tau$ consists in computing $cl(G)$ and check whether τ is included in $cl(G)$ [15]. [15] provides also an alternative method to test $G \models \tau$ that runs in time $\mathcal{O}(|G| \log |G|)$.

Query Answering. For the sake of our purpose, we get inspired by [6][3] and we assume that a RDF graph G is *ground* and *closed*, *i.e.*, G is closed under the application of the rules (2)-(5). Then a *conjunctive query* is a Datalog-like rule of the form

$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\tau_1, \ldots, \tau_n$$

where $n \geqslant 1$, $\tau_1, \ldots, \tau_n$ are triples, $\mathbf{x}$ is a vector of variables occurring in $\tau_1, \ldots, \tau_n$, called the *distinguished variables*, $\mathbf{y}$ are so-called *non-distinguished variables* and are distinct from the variables in $\mathbf{x}$, each variable occurring in τ_i is either a distinguished variable or a non-distinguished variable. If clear from the context, we may omit the exitential quantification $\exists \mathbf{y}$. For instance, the query

$$q(x, y) \leftarrow (x, creates, y), (x, \mathsf{type}, Flemish), (y, exhibited, Uffizi)$$

has intended meaning to retrieve all the artifacts x created by Flemish artists y, being exhibited at Uffizi Gallery.

We will also write a query as

$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y}) \ ,$$

where $\varphi(\mathbf{x}, \mathbf{y})$ is $\tau_1, \ldots, \tau_n$. Furthermore, $q(\mathbf{x})$ is called the *head* of the query, while $\exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$ is called the *body* of the query.

Finally, a *disjunctive query* (or, *union of conjunctive queries*) $\mathbf{q}$ is, as usual, a finite set of conjunctive queries in which all the rules have the same head.

Given a graph G, a query $q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$, and a vector $\mathbf{t}$ of terms in $\mathbf{UL}$, we say that $q(\mathbf{t})$ is *entailed* by G, denoted $G \models q(\mathbf{t})$, iff in any model $\mathcal{I}$ of G, there is a vector $\mathbf{t}'$ of terms in $\mathbf{UL}$ such that $\mathcal{I}$ is a model of $\varphi(\mathbf{t}, \mathbf{t}')$. If $G \models q(\mathbf{t})$ then $\mathbf{t}$ is called an *answer* to q. For a disjunctive query $\mathbf{q} = \{q_1, \ldots, q_m\}$, we say that $\mathbf{q}(\mathbf{t})$ is *entailed* by G, denoted $G \models \mathbf{q}(\mathbf{t})$, iff $G \models q_i(\mathbf{t})$ for some $q_i \in \mathbf{q}$. The *answer set* of $\mathbf{q}$ w.r.t. G is defined as

$$ans(G, \mathbf{q}) = \{\mathbf{t} \mid G \models \mathbf{q}(\mathbf{t})\} \ .$$

A simple method to determine $ans(G, \mathbf{q})$ is as follows. Compute the closure $cl(G)$ of G and store it into a database, *e.g.*, using the method [1]. It is easily verified that any disjunctive query can be mapped into union of SQL queries over the underlying database schema. Hence, $ans(G, \mathbf{q})$ is determined by issuing these SQL queries to the database.

3 Fuzzy RDF

We now present fuzzy RDF in its general form, by extending [12,13,14]. To do so and to make the paper self-contained, we first recall basic notions of mathematical fuzzy logic [9].

3.1 Preliminaries: Mathematical Fuzzy Logic

In mathematical fuzzy logics, the convention prescribing that a statement is either true or false is changed and is a matter of degree taken from a *truth space* $\mathcal{S}$, usually $[0, 1]$ (in that case we speak about *Mathematical Fuzzy Logic* [9]) or $\{\frac{0}{n}, \frac{1}{n}, \ldots, \frac{n}{n}\}$ for an

[3] Note that [15] does not address conjunctive query answering.

integer $n \geqslant 1$. Often $\mathcal{S}$ may also be a complete lattice or a bilattice [3,5] (often used in logic programming [4]). In the sequel, we assume $\mathcal{S} = [0, 1]$. This degree is called *degree of truth* of the statement ϕ in the interpretation $\mathcal{I}$.

In the illustrative fuzzy logic that we consider in this section, *fuzzy statements* have the form $\phi[n]$, where $n \in [0, 1]$ [8,9] and ϕ is a statement, which encodes that the degree of truth of ϕ is *at least* n. For example, $ripe\_tomato[0.9]$ says that we have a rather ripe tomato (the degree of truth of $ripe\_tomato$ is at least 0.9). Semantically, a *fuzzy interpretation* $\mathcal{I}$ maps each basic statement p_i into $[0, 1]$ and is then extended inductively to all statements as follows:

$$\begin{array}{ll} \mathcal{I}(\phi \wedge \psi) = \mathcal{I}(\phi) \otimes \mathcal{I}(\psi) & \mathcal{I}(\phi \vee \psi) = \mathcal{I}(\phi) \oplus \mathcal{I}(\psi), \\ \mathcal{I}(\phi \rightarrow \psi) = \mathcal{I}(\phi) \Rightarrow \mathcal{I}(\psi) & \mathcal{I}(\neg\phi) = \ominus\mathcal{I}(\phi)\,, \\ \mathcal{I}(\exists x.\phi(x)) = \sup_{c \in \Delta^{\mathcal{I}}} \mathcal{I}(\phi(c)) & \mathcal{I}(\forall x.\phi(x)) = \inf_{c \in \Delta^{\mathcal{I}}} \mathcal{I}(\phi(c)) \end{array} \tag{1}$$

where $\otimes$, $\oplus$, $\Rightarrow$, and $\ominus$ are so-called *combination functions*, namely, *triangular norms* (or *t-norms*), *triangular co-norms* (or *s-norms*), *implication functions*, and *negation functions*, respectively, which extend the classical Boolean conjunction, disjunction, implication, and negation, respectively, to the fuzzy case.

Several t-norms, s-norms, implication functions, and negation functions have been given in the literature. An important aspect of such functions is that they satisfy some properties that one expects to hold for the connectives; see Tables 1 and 2. Note that in Table 1, the two properties Tautology and Contradiction follow from Identity, Commutativity, and Monotonicity. Usually, the implication function $\Rightarrow$ is defined as *r-implication*, that is, $a \Rightarrow b = \sup\{c \mid a \otimes c \leqslant b\}$.

Some t-norms, s-norms, implication functions, and negation functions of various fuzzy logics are shown in Table 3 [9]. In fuzzy logic, one usually distinguishes three different logics, namely, Łukasiewicz, Gödel, and Product logic; the popular Zadeh logic is a sublogic of Łukasiewicz logic as, $min(x, y) = x \wedge (x \Rightarrow y)$ and $\max(x, y) = (x \Rightarrow y) \Rightarrow y$. Some salient properties of these logics are shown in Table 4. For more properties, see especially [9,16]. Note also, that a fuzzy logic having all properties shown in Table 4, collapses to boolean logic, *i.e.* the truth-set can be $\{0, 1\}$ only. Also note that the importance of these three logics is due the fact that any t-norm can be obtained as a combination of Łukasiewicz, Gödel, and Product t-norm. The implication $x \Rightarrow y = \max(1 - x, y)$ is called *Kleene-Dienes implication* in the fuzzy logic literature. Note that we have the following inferences: Let $a \geqslant n$ and $a \Rightarrow b \geqslant m$. Then, under Kleene-Dienes implication, we infer that "if $n > 1 - m$ then $b \geqslant m$". More importantly, under an r-implication relative to a t-norm $\otimes$, we have that

$$\text{from } a \geqslant n \text{ and } a \Rightarrow b \geqslant m, \text{ we infer } b \geqslant n \otimes m\,. \tag{2}$$

To see this, as $a \geqslant n$ and $a \Rightarrow b = \sup\{c \mid a \otimes c \leqslant b\} = \bar{c} \geqslant m$ it follows that $b \geqslant a \otimes \bar{c} \geqslant n \otimes m$. In a similar way, under an r-implication relative to a t-norm $\otimes$, we have that

$$\text{from } a \Rightarrow b \geqslant n \text{ and } b \Rightarrow c \geqslant m, \text{ we infer that } a \Rightarrow c \geqslant n \otimes m\,. \tag{3}$$

As we will see later on, these are the main inference patterns we will rely on in this paper.

Table 1. Properties for t-norms and s-norms

Axiom Name	T-norm	S-norm
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. Properties for implication and negation functions

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

Table 3. Combination functions of various fuzzy logics

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

Table 4. Some additional properties of combination functions of various fuzzy logics

Property	Łukasiewicz Logic	Gödel Logic	Product Logic	Zadeh Logic
$x \otimes \ominus x = 0$	+	+	+	−
$x \oplus \ominus x = 1$	+	−	−	−
$x \otimes x = x$	−	+	−	+
$x \oplus x = x$	−	+	−	+
$\ominus \ominus x = x$	+	−	−	+
$x \Rightarrow y = \ominus x \oplus y$	+	−	−	+
$\ominus (x \Rightarrow y) = x \otimes \ominus y$	+	−	−	+
$\ominus (x \otimes y) = \ominus x \oplus \ominus y$	+	+	+	+
$\ominus (x \oplus y) = \ominus x \otimes \ominus y$	+	+	+	+

Note that implication functions and t-norms are also used to define the degree of subsumption between fuzzy sets and the composition of two (binary) fuzzy relations. A *fuzzy set* R over a countable crisp set X is a function $R\colon X \to [0, 1]$. The *degree of subsumption* between two fuzzy sets A and B, denoted $A \sqsubseteq B$, is defined as

$$\inf_{x \in X} A(x) \Rightarrow B(x)\ , \tag{4}$$

where $\Rightarrow$ is an implication function. Note that in First-Order-Logic terms, A is a subclass of B may be seen as the formula

$$\forall x.A(x) \Rightarrow B(x)\ ,$$

and, as in fuzzy logic $\forall$ is the $\inf$, we get equation above. Together with (2), these are the two major notions we need later on.

Note that if $A(x) \leqslant B(x)$, for all $x \in [0,1]$, then $A \sqsubseteq B$ evaluates to 1. Of course, $A \sqsubseteq B$ may evaluate to a value $v \in (0,1)$ as well. A (binary) *fuzzy relation* R over two countable crisp sets X and Y is a function $R\colon X \times Y \to [0,1]$. The *composition* of two fuzzy relations $R_1\colon X \times Y \to [0,1]$ and $R_2\colon Y \times Z \to [0,1]$ is defined as $(R_1 \circ R_2)(x,z) = \sup_{y \in Y} R_1(x,y) \otimes R_2(y,z)$. A fuzzy relation R is *transitive* iff $R(x,z) \geqslant (R \circ R)(x,z)$.

A fuzzy interpretation $\mathcal{I}$ *satisfies* a fuzzy statement $\phi[n]$ or $\mathcal{I}$ is a *model* of $\phi[n]$, denoted $\mathcal{I} \models \phi[n]$, iff $\mathcal{I}(\phi) \geqslant n$. The notions of satisfiability and logical consequence are defined in the standard way. We say $\phi[n]$ is a *tight logical consequence* of a set of fuzzy statements KB iff n is the infimum of $\mathcal{I}(\phi)$ subject to all models $\mathcal{I}$ of KB. Notice that the latter is equivalent to $n = \sup\{r \mid KB \models \phi[r]\}$. We refer the reader to [7,8,9] for reasoning algorithms for fuzzy propositional and First-Order Logics.

3.2 Generalized Fuzzy RDF

We are now ready to extend the notions introduced in the previous section to fuzzy RDF. We start with the syntax and then define the semantics.

Syntax. A *fuzzy RDF triple* is an expression $\tau[n]$, where τ is a triple and $n \in [0,1]$. The intended semantics is that the degree of truth of τ is not less than n. For instance, $(audiTT, \mathsf{type}, SportsCar)[0.8]$ is a fuzzy triple, intending that AudiTT is almost a sport car. In a fuzzy triple $\tau[n]$, the truth value n may be omitted and, in that case, the value $n = 1$ is assumed. A *fuzzy RDF graph* $\tilde{G}$ (or simply a fuzzy graph, or *fuzzy RDF Knowledge Base*) is a set of fuzzy RDF triples $\tilde{\tau}$. The notions of *universe* of a graph $\tilde{G}$, the *vocabulary* of $\tilde{G}$, *ground* graph and *variable assignment* are as for the crisp case. Without loss of generality we may assume that there are not two fuzzy triples $\tau[n]$ and $\tau[m]$ in a fuzzy graph $\tilde{G}$. If this is the case, we may just remove the fuzzy triple with the lower score.

Semantics. The fuzzy semantics is derived directly from the crisp one, where the extension functions are no longer sets, but functions assigning a truth in $[0,1]$. So, let $\otimes$ be a t-norm and let $\Rightarrow$ be its r-implication. A *fuzzy RDF interpretation* $\mathcal{I}$ over a vocabulary V is a tuple

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle,$$

where $\Delta_R, \Delta_P, \Delta_C, \Delta_L$ are the interpretations domains of $\mathcal{I}$, and $P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}}$ are the interpretation functions of $\mathcal{I}$. They have to satisfy:

1. Δ_R is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. Δ_P is a set of property names (not necessarily disjoint from Δ_R);
3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of Δ_R identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, Δ_L contains all plain literals in $\mathbf{L} \cap V$;
5. $P[\![\cdot]\!]$ maps each property name $p \in \Delta_P$ into a partial function $P[\![p]\!] : \Delta_R \times \Delta_R \to [0,1]$, *i.e.* assigns a degree to each pair of resources, denoting the degree of being the pair an instance of the property p;

6. $C[\![\cdot]\!]$ maps each class $c \in \Delta_C$ into a partial function $C[\![c]\!] : \Delta_R \to [0,1]$, *i.e.* assigns a degree to every resource, denoting the degree of of the resource being an instance of class c;
7. $\cdot^{\mathcal{I}}$ maps each $t \in \mathbf{UL} \cap V$ into a value $t^{\mathcal{I}} \in \Delta_R \cup \Delta_P$, *i.e.* assigns a resource or a property name to each element of $\mathbf{UL}$ in V, and such that $\cdot^{\mathcal{I}}$ is the identity for plain literals and assigns an element in Δ_R to elements in $\mathbf{L}$;
8. $\cdot^{\mathcal{I}}$ maps each variable $x \in \mathbf{B}$ into a value $x^{\mathcal{I}} \in \Delta_R$, *i.e.* assigns a resource to each variable in $\mathbf{B}$.

Note that the only difference so far relies on points 5. and 6., in which the extension function become now fuzzy membership functions. Note also that $C[\![\cdot]\!]$ (resp. $P[\![\cdot]\!]$) is a *partial* function and, thus, is not defined on all arguments. Alternatively, we may define it to be a total function. We use the former formulation to distinguish the case where a tuple $\mathbf{t}$ may be an answer to a query, even though the score is 0, from the case where a tuple is not retrieved, since it does not satisfy the query conditions. In particular, if a triple does not belong to a fuzzy graph, then its truth is assumed to be undefined, while if $C[\![\cdot]\!]$ (resp. $P[\![\cdot]\!]$) is total, then its truth of this triple would be 0, which is a small though fundamental difference. Please note that both [14,21] rely on total interpretations. We prefer the partial semantics approach as we believe it is better suited for applications, as it is more "database-like" in query answering. For instance, suppose we are looking for a second-hand car, which is cheap and not too old, where cheap and old are functions of the price and age, respectively, and the cheapness and oldness scores are aggregated via weighted linear combination. Then under total semantics one may retrieve a car with non zero score, despite its age is unknown (the tuple relating the car to its age is not in the graph and, thus, the degree of oldness is 0, but there may be a tuple dictating the price of the car), while under partial semantics, this car will not be retrieved (as it would happen for a top-k database engine or using *e.g.* SPARQL [18] in which the scoring component of the query is omitted).

The notion entailment is defined using the idea of *satisfaction* of a graph under certain interpretation. Intuitively a ground fuzzy triple $(s,p,o)[n]$ in a fuzzy RDF graph $\tilde{G}$ will be satisfied under the interpretation $\mathcal{I}$ if p is interpreted as a property name, s and o are interpreted as resources, and the interpretation of the pair (s,o) belongs to the extension of the property assigned to p to degree not less than n.

Now, let $\tilde{G}$ be a fuzzy graph over ρdf. A fuzzy interpretation $\mathcal{I}$ is a *model* of $\tilde{G}$ under ρdf, denoted $\mathcal{I} \models \tilde{G}$, iff $\mathcal{I}$ is a fuzzy interpretation over the vocabulary $\rho\text{df} \cup universe(\tilde{G})$ that satisfies the following conditions:

Simple:

1. for each $(s,p,o)[n] \in G$, $p^{\mathcal{I}} \in \Delta_P$ and $P[\![p^{\mathcal{I}}]\!](s^{\mathcal{I}}, o^{\mathcal{I}}) \geqslant n$;

Subproperty:

1. $P[\![\mathsf{sp}^{\mathcal{I}}]\!]$ is transitive over Δ_P;
2. if $P[\![\mathsf{sp}^{\mathcal{I}}]\!](p,q)$ is defined then $p, q \in \Delta_P$ and

$$P[\![\mathsf{sp}^{\mathcal{I}}]\!](p,q) = \inf_{(x,y) \in \Delta_R \times \Delta_R} P[\![p]\!](x,y) \Rightarrow P[\![q]\!](x,y) \; ;$$

Subclass:

1. $P[\![\mathsf{sc}^{\mathcal{I}}]\!]$ is transitive over Δ_C;

2. if $P[\![\mathsf{sc}^{\mathcal{I}}]\!](c,d)$ is defined then $c,d \in \Delta_C$ and

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](c,d) = \inf_{x\in\Delta_R} C[\![c]\!](x) \Rightarrow C[\![d]\!](x) \; ;$$

Typing I:

1. $C[\![c]\!](x) = P[\![\mathsf{type}^{\mathcal{I}}]\!](x,c)$;
2. if $P[\![\mathsf{dom}^{\mathcal{I}}]\!](p,c)$ is defined then

$$P[\![\mathsf{dom}^{\mathcal{I}}]\!](p,c) = \inf_{(x,y)\in\Delta_R\times\Delta_R} P[\![p]\!](x,y) \Rightarrow C[\![c]\!](x) \; ;$$

3. if $P[\![\mathsf{range}^{\mathcal{I}}]\!](p,c)$ is defined then

$$P[\![\mathsf{range}^{\mathcal{I}}]\!](p,c) = \inf_{(x,y)\in\Delta_R\times\Delta_R} P[\![p]\!](x,y) \Rightarrow C[\![c]\!](y) \; ;$$

Typing II:

1. For each $\mathsf{e} \in \rho\mathsf{df}$, $\mathsf{e}^{\mathcal{I}} \in \Delta_P$
2. if $P[\![\mathsf{dom}^{\mathcal{I}}]\!](p,c)$ is defined then $p \in \Delta_P$ and $c \in \Delta_C$
3. if $P[\![\mathsf{range}^{\mathcal{I}}]\!](p,c)$ is defined then $p \in \Delta_P$ and $c \in \Delta_C$
4. if $P[\![\mathsf{type}^{\mathcal{I}}]\!](x,c)$ is defined then $c \in \Delta_C$

Some explanations about the above definitions are in place. To do so, let us keep in mind Eq. (4). At first, let us explain condition 2 of the subclass condition. In the crisp case if c is a sub-class of d then we impose that $C[\![c]\!] \subseteq C[\![d]\!]$. The fuzzyfication of this subsumption condition yields the degree of subsumption and, thus, using Eq. (4), we get immediately

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](c,d) = \inf_{x\in\Delta_R} C[\![c]\!](x) \Rightarrow C[\![d]\!](x) \; .$$

i.e., $P[\![\mathsf{sc}^{\mathcal{I}}]\!](c,d)$ is evaluated as the degree of subsumption between class c and class d. In First-Order-Logic terms, we recall that c is a sub-class of d may be seen as the formula

$$\forall x.c(x) \Rightarrow d(x) \; ,$$

and, thus, as in fuzzy logic $\forall$ is the $\inf$, we get equation above. The argument for the sub-property condition is similar. Concerning condition 2 of Typing I, we may write the condition that property p has domain c in First-Order-Logic as

$$\forall x \forall y.p(x,y) \Rightarrow c(x) \; ,$$

which then gives us immediately the condition

$$P[\![\mathsf{dom}^{\mathcal{I}}]\!](p,c) = \inf_{(x,y)\in\Delta_R\times\Delta_R} P[\![p]\!](x,y) \Rightarrow C[\![c]\!](x) \; .$$

The argument for condition 3 of Typing I is similar. We define $\tilde{G}$ *entails* $\tilde{H}$ under ρdf, denoted $\tilde{G} \models \tilde{H}$, iff every fuzzy model under ρdf of $\tilde{G}$ is also a model under ρdf of $\tilde{H}$.

As for the crisp case, it can be shown that any fuzzy graph is *consistent*, *i.e.* has a model.

Proposition 2 (Consistency). *Any fuzzy RDF graph has a model.*

Therefore, unlike [21], we do not have to care about consistency checking.

Deductive system. In what follows, we present a sound and complete deductive system for our fuzzy RDF fragment. As we will see, it is an extension of the one we have seen for the crisp case. Indeed, for each crisp rule (except for group 1, which remains identical) there is a fuzzy analogue. The rules are as follows[4]:

1. Simple:

   (a) $\frac{\tilde{G}}{\tilde{G}'}$ for a map $\mu : \tilde{G}' \rightarrow \tilde{G}$ (b) $\frac{G}{\tilde{G}'}$ for $\tilde{G}' \subseteq \tilde{G}$

2. Subproperty:

   (a) $\dfrac{(A, \mathsf{sp}, B)[n], (B, \mathsf{sp}, C)[m]}{(A, \mathsf{sp}, C)[n \otimes m]}$ (b) $\dfrac{(A, \mathsf{sp}, B)[n], (X, A, Y)[m]}{(X, B, Y)[n \otimes m]}$

3. Subclass:

   (a) $\dfrac{(A, \mathsf{sc}, B)[n], (B, \mathsf{sc}, C)[m]}{(A, \mathsf{sc}, C)[n \otimes m]}$ (b) $\dfrac{(A, \mathsf{sc}, B)[n], (X, \mathsf{type}, A)[m]}{(X, \mathsf{type}, B)[n \otimes m]}$

4. Typing:

   (a) $\dfrac{(A, \mathsf{dom}, B)[n], (X, A, Y)[m]}{(X, \mathsf{type}, B)[n \otimes m]}$ (b) $\dfrac{(A, \mathsf{range}, B)[n], (X, A, Y)[m]}{(Y, \mathsf{type}, B)[n \otimes m]}$

5. Implicit Typing:

   (a) $\dfrac{(A, \mathsf{dom}, B)[n], (C, \mathsf{sp}, A)[m], (X, C, Y)[r]}{(X, \mathsf{type}, B)[n \otimes m \otimes r]}$

   (b) $\dfrac{(A, \mathsf{range}, B)[n], (C, \mathsf{sp}, A)[m], (X, C, Y)[r]}{(Y, \mathsf{type}, B)[n \otimes m \otimes r]}$

It suffices to explain the rules of the sub-class category, as all the rules of categories 2-5 follow the same schema. To do so, consider inference schemas (2) and (3).

Consider the rule

$$\frac{(A, \mathsf{sc}, B)[n], (B, \mathsf{sc}, C)[m]}{(A, \mathsf{sc}, C)[n \otimes m]} .$$

Let us show that the rule is sound, *i.e.* for a fuzzy interpretation $\mathcal{I}$, if $\mathcal{I} \models (A, \mathsf{sc}, B)[n]$ and $\mathcal{I} \models (B, \mathsf{sc}, C)[m]$ then $\mathcal{I} \models (A, \mathsf{sc}, C)[n \otimes m]$. Indeed,

1. As $P[\![\mathsf{sc}^{\mathcal{I}}]\!]$ is transitive over Δ_C, we have that

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](A^{\mathcal{I}}, C^{\mathcal{I}}) \geqslant P[\![\mathsf{sc}^{\mathcal{I}}]\!](A^{\mathcal{I}}, B^{\mathcal{I}}) \otimes P[\![\mathsf{sc}^{\mathcal{I}}]\!](B^{\mathcal{I}}, C^{\mathcal{I}}) .$$

2. As $\mathcal{I} \models (A, \mathsf{sc}, B)[n]$, it follows that

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](A^{\mathcal{I}}, B^{\mathcal{I}}) = \inf_{x \in \Delta_R} C[\![A^{\mathcal{I}}]\!](x) \Rightarrow C[\![B^{\mathcal{I}}]\!](x) \geqslant n ;$$

3. As $\mathcal{I} \models (B, \mathsf{sc}, C)[m]$, it follows that

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](B^{\mathcal{I}}, C^{\mathcal{I}}) = \inf_{x \in \Delta_R} C[\![B^{\mathcal{I}}]\!](x) \Rightarrow C[\![C^{\mathcal{I}}]\!](x) \geqslant m ;$$

[4] An excerpt of them has been provided in [19].

4. From 1-3, it follows immediately that

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](A^{\mathcal{I}}, C^{\mathcal{I}}) \geqslant n \otimes m$$

and, thus $\mathcal{I} \models (A, \mathsf{sc}, C)[n \otimes m]$.

Next, let us show that

$$\frac{(A, \mathsf{sc}, B)[n], (X, \mathsf{type}, A)[m]}{(X, \mathsf{type}, B)[n \otimes m]}$$

is correct.

1. As $\mathcal{I} \models (A, \mathsf{sc}, B)[n]$, it follows that

$$P[\![\mathsf{sc}^{\mathcal{I}}]\!](A^{\mathcal{I}}, B^{\mathcal{I}}) = \inf_{x \in \Delta_R} C[\![A^{\mathcal{I}}]\!](x) \Rightarrow C[\![B^{\mathcal{I}}]\!](x) \geqslant n \; ;$$

2. As $\mathcal{I} \models (X, \mathsf{type}, A)[m]$, it follows that

$$P[\![\mathsf{type}^{\mathcal{I}}]\!](X^{\mathcal{I}}, A^{\mathcal{I}}) \geqslant m \; ;$$

3. But $C[\![A^{\mathcal{I}}]\!](X^{\mathcal{I}}) = P[\![\mathsf{type}^{\mathcal{I}}]\!](X^{\mathcal{I}}, A^{\mathcal{I}})$ and, thus, from 2. we get

$$C[\![A^{\mathcal{I}}]\!](X^{\mathcal{I}}) \geqslant m \; .$$

4. Consider $X^{\mathcal{I}} \in \Delta_R$. As 1. holds for all $x \in \Delta_R$, we have that

$$C[\![A^{\mathcal{I}}]\!](X^{\mathcal{I}}) \Rightarrow C[\![B^{\mathcal{I}}]\!](x) \geqslant n \; .$$

5. By schema (2) and point 3. and 4., we get

$$C[\![B^{\mathcal{I}}]\!]((X^{\mathcal{I}}) \geqslant n \otimes m \; .$$

6. But, $P[\![\mathsf{type}^{\mathcal{I}}]\!](X^{\mathcal{I}}, B^{\mathcal{I}}) = C[\![B^{\mathcal{I}}]\!](X^{\mathcal{I}})$ and, thus, by 5. we get

$$P[\![\mathsf{type}^{\mathcal{I}}]\!](X^{\mathcal{I}}, B^{\mathcal{I}}) \geqslant n \otimes m$$

and, thus $\mathcal{I} \models (X, \mathsf{type}, B)[n \otimes m]$.

The notion of proof is as for the crisp case and we have:

Proposition 3 (Soundness and completeness). *For fuzzy RDF, the proof system $\vdash$ is sound and complete for $\models$, that is, $\tilde{G} \vdash \tilde{H}$ iff $\tilde{G} \models \tilde{H}$.*

Query Answering. We extend the notion of conjunctive query to the case in which a scoring function can be specified to score the answers similarly as in [11] (see also [20]).

For the sake of our purpose, we assume that a fuzzy RDF graph $\tilde{G}$ is *ground* and *closed*, *i.e.*, $\tilde{G}$ is closed under the application of the rules (2)-(5). Then a *fuzzy conjunctive query* extends a crisp query and is of the form

$$q(\mathbf{x})[s] \leftarrow \exists \mathbf{y}.\tau_1[s_1], \ldots, \tau_n[s_n], s = f(s_1, \ldots, s_n, p_1(\mathbf{z}_1), \ldots, p_h(\mathbf{z}_h))$$

where additionally

1. $\mathbf{z_i}$ are tuples of terms in $\mathbf{UL}$ or variables in $\mathbf{x}$ or $\mathbf{y}$;

2. p_j is an n_j-ary *fuzzy predicate* assigning to each n_j-ary tuple $\mathbf{t}_j$ in $\mathbf{UL}$ a *score* $p_j(\mathbf{t}_j) \in [0,1]_m$. Such predicates are called *expensive predicates* in [2] as the score is not pre-computed off-line, but is computed on query execution. We require that an n-ary fuzzy predicate p is *safe*, that is, there is not an m-ary fuzzy predicate p' such that $m < n$ and $p = p'$. Informally, all parameters are needed in the definition of p;
3. f is a *scoring* function $f\colon ([0,1])^{n+h} \to [0,1]$, which combines the scores s_i of the n triples and the h fuzzy predicates into an overall *score* to be assigned to the rule head. We assume that f is *monotone*, that is, for each $\mathbf{v}, \mathbf{v}' \in ([0,1])^{n+h}$ such that $\mathbf{v} \leqslant \mathbf{v}'$, it holds $f(\mathbf{v}) \leqslant f(\mathbf{v}')$, where $(v_1, \ldots, v_{n+h}) \leqslant (v'_1, \ldots, v'_{n+h})$ iff $v_i \leqslant v'_i$ for all i;
4. the scoring variables s and s_i are distinct from those in $\mathbf{x}$ and $\mathbf{y}$ and s is distinct from each s_i.

We may omit s_i and in that case $s_i = 1$ is assumed. $s = f(s_1, \ldots, s_n, p_1(\mathbf{z}_1), \ldots, p_h(\mathbf{z}_h))$ is called the *scoring atom*. We may also omit the scoring atom and in that case $s = 1$ is assumed. For instance, the query

$$q(x)[s] \leftarrow (x, \mathsf{type}, \mathit{SportsCar})[s_1], (x, \mathit{hasPrice}, y), s = s_1 \cdot \mathit{cheap}(y)$$

where *e.g.* $\mathit{cheap}(p) = \max(0, 1 - \frac{p}{12000})$, has intended meaning to retrieve all cheap sports car. Any answer is scored according to the product of being cheap and a sports car.

The notion of disjunctive query is as for the crisp case. We will also write a query as

$$q(\mathbf{x})[s] \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})[\mathbf{s}] \ ,$$

where $\varphi(\mathbf{x}, \mathbf{y})$ is $\tau_1[s_1], \ldots, \tau_n[s_n], s = f(\mathbf{s}, p_1(\mathbf{z}_1), \ldots, p_h(\mathbf{z}_h))$ and $\mathbf{s} = \langle s_1, \ldots, s_n \rangle$.

Consider a fuzzy graph $\tilde{G}$, a query $q(\mathbf{x})[s] \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})[\mathbf{s}]$, a vector $\mathbf{t}$ of terms in $\mathbf{UL}$ and $s \in [0,1]$. We say that $q(\mathbf{t})[s]$ is *entailed* by $\tilde{G}$, denoted $\tilde{G} \models q(\mathbf{t})[s]$, iff in any model $\mathcal{I}$ of $\tilde{G}$, there is a vector $\mathbf{t}'$ of terms in $\mathbf{UL}$, a vector $\mathbf{s}$ of scores in $[0,1]$ such that $\mathcal{I}$ is a model of $\varphi(\mathbf{t}, \mathbf{t}')[\mathbf{s}]$ (the scoring atom is satisfied iff s is the value of the evaluation of the score combination function). The definition is extended to a disjunctive query as for the crisp case.

We say that s is *tight* iff $s = \sup\{s' \mid \tilde{G} \models \mathbf{q}(\mathbf{t})[s']\}$. If $\tilde{G} \models \mathbf{q}(\mathbf{t})[s]$ and s is tight then $\mathbf{t}[s]$ is called an *answer* to $\mathbf{q}$ w.r.t. $\tilde{G}$. The *answer set*, $\mathit{ans}(\tilde{G}, \mathbf{q})$ of $\mathbf{q}$ w.r.t. $\tilde{G}$, is defined as the set of answers to $\mathbf{q}$ w.r.t. $\tilde{G}$.

As now each answer to a query has a degree of truth (*i.e.* score), the basic inference problem that is of interest in is the top-k retrieval problem, formulated as follows.

Top-k Retrieval. Given a fuzzy graph $\tilde{G}$, and a disjunctive query $\mathbf{q}$, retrieve k answers $\mathbf{t}[s]$ with maximal scores and rank them in decreasing order relative to the score s, denoted

$$\mathit{ans}_k(\tilde{G}, \mathbf{q}) = \mathit{Top}_k \ \mathit{ans}(\tilde{G}, \mathbf{q}) \ .$$

Next, we describe a method to determine $\mathit{ans}_k(\tilde{G}, \mathbf{q})$. So, let $\tilde{G}$ be a ground fuzzy graph. similarly to the crisp case, the *closure* of $\tilde{G}$ is defined as

$$\mathit{cl}(\tilde{G}) = \{\tau[n] \mid \tau \text{ ground}, \tilde{G} \vdash \tau[n] \text{ and } n \text{ is tight}\}^5.$$

[5] Note that rule (Simple a) is not required to compute the closure.

Note that by definition of $cl(\tilde{G})$, there cannot bet two fuzzy triples $\tau[n]$ and $\tau[m]$ in $cl(\tilde{G})$ such that $n < m$. The closure $cl(\tilde{G})$ can be computed by repeatedly applying the fuzzy inference rules together with the *redundancy elimination rule* below:

– Redundancy Elimination Rule (RER):

$$\frac{\tau[n],\ \tau[m]}{\text{remove } \tau[n]} \text{ if } n < m$$

Essentially, each time we generate a tuple τ, we keep the one involving τ with highest degree. This rule is necessary in order to guarantee the termination of the closure computation in case of cyclic graphs, such as *e.g.*

$$(A, \mathsf{sc}, B)[n],\ \ (B, \mathsf{sc}, C)[n],\ \ (C, \mathsf{sc}, A)[n]$$

where the t-norm is *e.g.* product. Without (RER), we may generate an infinite sequence of fuzzy triples $(A, \mathsf{sc}, A)[n^{3k}]$ $(k = 1, 2, \ldots)$ and, thus, do not terminate. Please note that with (RER), only $(X, \mathsf{sc}, X)[n^3] \in cl(\tilde{G})$, where $X \in \{A, B, C\}$.

Now, under the above closure computation, we have, as for the crisp case [15]:

Proposition 4 (Size of Closure)

1. *The size of the closure of $\tilde{G}$ is $\mathcal{O}(|\tilde{G}|^2)$.*
2. *The size of the closure of $\tilde{G}$ is in the worst case no smaller than $\Omega(|\tilde{G}|^2)$.*

Therefore, a method to determine $ans_k(\tilde{G}, \mathbf{q})$ is as follows.

1. Compute the closure $cl(\tilde{G})$ of $\tilde{G}$ and store it into a database that supports top-k retrieval (*e.g.*, RankSQL [10][6]).
2. It can be verified that any fuzzy disjunctive query can be mapped into union of top-k SQL queries [10] over the underlying database schema.
3. Hence, $ans_k(\tilde{G}, \mathbf{q})$ is determined by issuing these top-k SQL queries to the database.

4 Summary and Outlook

We have presented fuzzy RDF under a generalized semantics based on t-norms and its r-implication. We provided a minimal deductive system, top-k fuzzy disjunctive queries and showed how these can be answered by relying on the closure computation and state of the art top-k database engines. An implementation is under development, where fuzzy triples are stored as RDF triples using reification and, thus, no change to RDF is required. We follow the method [1] to store the closure in the database.

Concerning future research: so far, we considered the closure of the graph (of quadratic size) to be stored into a database and then we submit top-k SQL queries to it. While one may think of extending the method described in [15] to check entailment

[6] But, *e.g.*, Postgres `http://www.postgresql.org/`,
MonetDB `http://monetdb.cwi.nl/`
may work as well.

for ground fuzzy tuples in $\mathcal{O}(|\tilde{G}| \log |\tilde{G}|)$, it remains to see whether similar methods exist to determine the top-k answers. To this end we are looking to the techniques developed for top-k query answering in fuzzy logic programming (see, *e.g.* [11,20]).

Another topic concerns the extension and mapping of SPARQL to fuzzy disjunctive queries.

References

1. Abadi, D.J., Marcus, A., Madden, S., Hollenbach, K.: Sw-store: a vertically partitioned DBMS for semantic web data management. VLDB J. 18(2), 385–406 (2009)
2. Chang, K.C.-C., won Hwang, S.: Minimal probing: Supporting expensive predicates for top-k queries. In: SIGMOD Conference, pp. 346–357 (2002)
3. Fitting, M.C.: Bilattices are nice things. In: Conference on Self-Reference, Copenhagen, Denmark (2002)
4. Fitting, M.C.: Fixpoint semantics for logic programming - a survey. Theoretical Computer Science 21(3), 25–51 (2002)
5. Ginsberg, M.L.: Multi-valued logics: a uniform approach to reasoning in artificial intelligence. Computational Intelligence 4, 265–316 (1988)
6. Gutierrez, C., Hurtado, C., Mendelzon, A.O.: Foundations of semantic web databases. In: Proceedings of the 23rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS 2004). ACM Press, New York (2004)
7. Hähnle, R.: Many-valued logics and mixed integer programming. Annals of Mathematics and Artificial Intelligence 3, 4(12), 231–264 (1994)
8. Hähnle, R.: Advanced many-valued logics. In: Gabbay, D.M., Guenthner, F. (eds.) Handbook of Philosophical Logic, 2nd edn., vol. 2. Kluwer, Dordrecht (2001)
9. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer, Dordrecht (1998)
10. Li, C., Chang, K.C.-C., Ilyas, I.F., Song, S.: RankSQL: query algebra and optimization for relational top-k queries. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD 2005), pp. 131–142. ACM Press, New York (2005)
11. Lukasiewicz, T., Straccia, U.: Top-k retrieval in description logic programs under vagueness for the semantic web. In: Prade, H., Subrahmanian, V.S. (eds.) SUM 2007. LNCS (LNAI), vol. 4772, pp. 16–30. Springer, Heidelberg (2007)
12. Mazzieri, M.: A fuzzy RDF semantics to represent trust metadata. In: Proceedings of the 1st Italian Semantic Web Workshop: Semantic Web Applications and Perspectives, SWAP 2004 (2004)
13. Mazzieri, M., Dragoni, A.F.: A fuzzy semantics for semantic web languages. In: Proceedings of the ISWC Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2005). CEUR Workshop Proceedings (2005)
14. Mazzieri, M., Dragoni, A.F.: A fuzzy semantics for the resource description framework. In: da Costa, P.C.G., d'Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M. (eds.) URSW 2005 - 2007. LNCS (LNAI), vol. 5327, pp. 244–261. Springer, Heidelberg (2008)
15. Muñoz, S., Pérez, J., Gutierrez, C.: Minimal deductive systems for RDF. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 53–67. Springer, Heidelberg (2007)
16. Novák, V.: Which logic is the real fuzzy logic? Fuzzy Sets and Systems, 635–641 (2005)
17. RDF, `http://www.w3.org/RDF/`
18. SPARQL, `http://www.w3.org/TR/rdf-sparql-query/`

19. Straccia, U.: Basic concepts and techniques for managing uncertainty and vagueness in semantic web languages. In: Reasoning Web, 4th International Summer School (2007) (invited Lecture)
20. Straccia, U.: Managing uncertainty and vagueness in description logics, logic programs and description logic programs. In: Baroglio, C., Bonatti, P.A., Małuszyński, J., Marchiori, M., Polleres, A., Schaffert, S. (eds.) Reasoning Web. LNCS, vol. 5224, pp. 54–103. Springer, Heidelberg (2008)
21. Udrea, O., Recupero, D.R., Subrahmanian, V.S.: Annotated RDF. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 487–501. Springer, Heidelberg (2006)
22. Vaneková, V., Bella, J., Gurský, P., Horváth, T.: Fuzzy rdf in the semantic web: Deduction and induction. In: Proceedings of Workshop on Data Analysis, WDA 2005 (2005)