Fuzzy Reasoning over RDF Data using OWL Vocabulary

Chang Liu*, Guilin Qi†, Haofen Wang*, Yong Yu*
*Shanghai Jiaotong University, Shanghai, China
{liuchang, whfcarter,yyu}@apex.sjtu.edu.cn
†Southeast University, China, Nanjing, China
gqi@seu.edu.cn

***Abstract*—In this paper, we propose fuzzy pD^* semantics which generalizes pD^* semantics to reason over fuzzy RDF data using OWL vocabulary. We first define the notions of fuzzy RDF graph and fuzzy pD^* interpretation. We then present a set of fuzzy pD^* entailment rules and define the *Best Degree Bound* (BDB) of a triple derived from a fuzzy RDF graph. We show the existence of the BDB of an arbitrary triple. After that, we generalize the partial and full pD^* closures to obtain the BDBs of derived triples. We show that the partial fuzzy closure exists and can be computed within polynomial time. Finally, we prove soundness and completeness results for the entailment relation. We also prove that the consistency checking is in P, the entailment is *NP-complete*, and in P if the target fuzzy RDF graph is ground. Therefore, extending the pD^* semantics with fuzzy semantics does not increase the computational complexity.**

***Keywords*-Semantic Web; OWL; Fuzzy Logic**

I. Introduction

The Resource Description Framework (RDF) [1] is one of the major representation standards for the Semantic Web. In an RDF document, one can declare a statement using a triple of the form (s, p, o), such as $(Tom, eat, pizza)$, representing a subject s has property p with value o. RDF Schema (RDFS) [2] is used to describe vocabularies used in RDF descriptions. However, RDF and RDFS only provide a very limited expressiveness. In [3], a subset of Ontology Web Language (OWL) vocabulary (e.g., owl:sameAs) was introduced, which extends the RDFS semantics to so-called pD^* semantics. Unlike the standard OWL (DL or Full) semantics which provides the full iff semantics, pD^* semantics follows RDF(S)'s if semantics. That is, the pD^* fragment of OWL provides a complete set of entailment rules, which guarantees that the entailment relationship can be determined within polynomial time under a non-trivial condition (if the target graph is ground). It has become a very promising ontology language for Semantic Web as it trades off the high computational complexity of OWL Full and the limited expressiveness of RDFS rules.

Recently, there is an increasing interest in extending RDF to represent vague information on Web (see, for example [4], [5], [6], [7], [8], [9]). Fuzzy RDF allow us to state to a certain degree, a triple is true, i.e. $(Tom, eat, pizza)$ is true with degree at least 0.8. However, fuzzy RDF (or fuzzy RDFS) has limited expressive power to represent information on the Web. Let us consider a multimedia application to illustrate the weakness of fuzzy RDF.

Example 1: Given four pictures p1, p2, p3, p4, we consider the following fuzzy statements: (1) p1 is a bright picture to degree 0.7, (2) p1 and p2 look like the same to degree 0.9, (3) p2 contains p3 as a part to degree 0.6, and (4) p3 contains p4 as a part to degree 0.5. Statement (1) can be represented as a fuzzy RDF triple. To represent statement (2), we will have to use OWL vocabulary owl:sameAs, which is interpreted as a symmetric and transitive predicate under pD^* semantics. For statements (3) and (4), we have to define a new property, e.g. term "contain", as the predicates, and in commonsense this property is modeled as a transitive property so that we can infer the statement "p2 contains p4 as a part to degree 0.3" from statements (3) and (4) (under product t-norm in fuzzy logic). Therefore, additional OWL vocabularies like owl:TransitiveProperty need to be introduced.

One may think that we can use fuzzy extension of description logics (see, for example [10], [11], [12]) to enhance the expressive power of fuzzy RDF(S) as description logics underpin OWL DL or its fragments. While this may be a possible solution, the high computational complexity of expressive fuzzy description logics hinders their application to deal with large scalable data on the Semantic Web. So far, most of the works on fuzzy description logics stay in a theoretical level, with very few deal with scalable reasoning services over fuzzy ontologies [13]. However, the work done in [13] is based on fuzzy DL-Lite, where DL-Lite is an inexpressive DL language, which cannot represent the transitive property given in our example. Surprisingly, although the pD^* fragment of OWL has been considered as well-suited ontology language for Semantic Web, there is still no work on its fuzzy extension.

In this work, we provide a fuzzy extension of the pD^* semantics to deal with vagueness in OWL ontology. We first propose a fuzzy pD^* semantics based on an arbitrary *t-norm function* to generalize the pD^* semantics, so that we can reason over fuzzy RDF data using OWL vocabulary. We then provide the fuzzy pD^* entailment rules by generalizing the pD^* entailment rules. We define the notion of Best Degree Bound (BDB) of a triple by using the fuzzy pD^* entailment rules. The notion of BDB was originally proposed in fuzzy description logics [12], where the BDB of a DL axiom is the greatest lower bound of the fuzzy degrees of this axiom. After that, we define the partial and full fuzzy pD^* closures to obtain the BDBs of derived triples. We show that the partial fuzzy closure exists for any fuzzy RDF graph and can be computed within polynomial time. Finally, we prove that the consistency checking is in P, the entailment is *NP-complete*, and in P if the target fuzzy RDF graph is ground. These computational results show that extending the pD^* semantics with fuzzy semantics does not increase the computational complexity.

The main difficulty of this work is to show the computational complexity of the reasoning tasks in our fuzzy pD^* semantics. As it turns out, it is even nontrivial to show that calculating the partial closure by applying fuzzy pD^* entailment rules in arbitrary order will eventually terminate at a fix point. Furthermore, to show the polynomial time complexity to calculate the partial closure, we have to develop a sophisticated algorithm by introducing a notion called *local BDB*, which is slightly different from the notion of BDB.

The rest of this paper is organized as follows. We provide some background knowledge in Section III. In Section IV, we define the fuzzy RDF graph and the fuzzy pD^* semantics. We define the fuzzy pD^* entailment relation and discuss its related problems in Section V. Finally, we conclude our work and provide some future work in Section VI.

II. Related work

Our work on extending the pD^* semantics to deal with vagueness is inspired from the work given in [4] which provides a minimal deductive system for fuzzy RDF under a very general semantics. However, [4] is based on ρdf [14], while we consider the pD^* semantics [3]. To meet this end, new challenges arise, such as developing a complete entailment system such that a polynomial time algorithm for calculating the partial closure of a generalized RDF graph exists. Moreover, we disallow a fuzzy triple with fuzzy degree 0 appearing in a fuzzy RDF graph, while [4] allows it for the purpose of query answering. In [8], [9], the work in [4] was extended to a more general framework which allows attaching meta information such as temporal validity and a fuzzy value to a triple. However, these general frameworks are based on ρdf, thus do not allow OWL constructors.

III. Background knowledge

A. RDF and pD^* semantics

1) Syntax: An RDF graph is defined as a subset of the set $\mathbf{UB}\times\mathbf{UB}\times\mathbf{UBL}$[1][2], where $\mathbf{U}$, $\mathbf{B}$ and $\mathbf{L}$ represent the set of *URI references*, *Blank nodes*, and *Literals respectively*. Furthermore $\mathbf{L}$ is composed of the set of plain literals L_p, well-typed literals L_t and ill-typed literals. The element (s,p,o) of an RDF graph is a *triple*. Given an RDF graph G, $T(G)$ is the set of elements in $\mathbf{UBL}$ that occur in the triples of G. The set of blank nodes of G is denoted as $bl(G) = T(G) \cap B$. The vocabulary of G is denoted as $V(G) = T(G) \cap \mathbf{UL}$. If $bl(G) = \emptyset$, we say G is *ground*.

[1]We assume $\mathbf{U}$, $\mathbf{B}$ and $\mathbf{L}$ fixed, and we concatenate the names to denote the union of them for ease

[2]As in [3], we allow predicate to be blank node

In this work, we rely on the pD^* semantics [3] which contains RDFS as a fragment and extends the semantics to contain a subset of OWL vocabulary, such as owl:sameAs and owl:allValuesFrom, while it guarantees a sound and complete entailment rule set. For the space limitation, we do not list all vocabularies here, but use $rdfV$, $rdfsV$ and pV to denote the RDF, RDFS and OWL vocabulary respectively as used in [3].

2) Semantics: A *simple interpretation* I over a vocabulary V is a tuple $I = \langle R_I, P_I, E_I, S_I, L_I, LV_I\rangle$ where

1. R_I is a non-empty set, called the set of resources or the universe of I;
2. P_I is the set of properties, which is not necessarily disjoint from R_I;
3. $\mathrm{LV}_I \subset R_I$ is the set of literal values, which contains at least all plain literals in $V \cap \mathbf{L}$;
4. E_I maps every property $p \in P_I$ to a subset of $R_I \times R_I$;
5. S_I maps each $t \in V \cap \mathbf{U}$ into $R_I \cup P_I$ that defines interpretation of URI references;
6. L_I maps each $t \in V \cap L_\mathrm{t}$ into R_I that defines interpretation of typed literals;

If I is a simple interpretation of a vocabulary V, then I can also be treated as a function with domain V in the following way

$$I(a) = \begin{cases} a & a \in L_\mathrm{p} \cap V \\ L_I(a) & a \in L_\mathrm{t} \cap V \\ S_I(a) & a \in U \cap V \end{cases} \tag{1}$$

Given a simple interpretation I and a partial function $A : B \rightharpoonup R_I$, we can define a function I_A that extends the domain of I to the domain of A:

$$I_A(a) = \begin{cases} A(v) & v \in B \cap \mathrm{dom}(A) \\ I(a) & a \in \mathrm{dom}(I) \end{cases} \tag{2}$$

Let G be an arbitrary RDF graph, we say I satisfies G if there is some function $A : bl(G) \to R_I$, such that for each triple $(s,p,o) \in G$, we have $I_A(p) \in P_I$ and $(I_A(s), I_A(o)) \in E_I(I_A(p))$. Intuitively, a triple (s,p,o) in a graph G is satisfied by the interpretation I, if p is interpreted as a property name, s and o are interpreted as resources, and the pair of resources assigned to (s,o) belongs to the extension of the interpreted property of p. Furthermore, blank nodes are treated as variables, e.g. the triple (x,p,o) with $x \in \mathbf{B}$ is satisfied by I if there is a function A maps x to $I(s)$, for some s such that (s,p,o) is satisfied by I.

A datatype d is composed by three parts: the lexical space $L(d)$, the value space $V(d)$ and the lexical-to-value mapping function $L2V(d) : L(d) \rightarrow V(d)$. Both $L(d)$ and $V(d)$ are non-empty sets. A datatype map is a partial function D from U to the set of datatypes. Given a datatype map D, the D-vocabulary is the domain $\mathbf{dom}(D)$ of D. The set of all well-typed D-literals with type in D is defined by $L_D^+ = \{(s, a) \in L_t : a \in dom(D), s \in L(D(a))\}$.

A simple interpretation I over the vocabulary *rdfV* $\cup$ *rdfsV* $\cup$ $\mathbf{dom}(D) \cup pV$ is a said to be a pD^*-interpretation if I satisfies all D-constraints and P-constraints mentioned in [3]. Because of the space limitation, we do not list these constraints here. An RDF graph S is said to pD^* *entail* an RDF graph G, denoted as $S \models_p G$, if every pD^* interpretation that satisfies S also satisfies G.

B. Fuzzy Logic

Fuzzy logics [15] are introduced to deal with vagueness. The true and false value of a statement in classical logic is generalized to a *fuzzy degree* from a truth space $\mathcal{S}$, which is usually [0, 1].

Fuzzy statements have the form $\phi[n]$, where ϕ is a statement and $n \in [0, 1]$. $\phi[n]$ means the degree of truth of ϕ is at least n. Semantically, a fuzzy interpretation I maps each basic statement ϕ into $[0, 1]$ and can be extended to all statements as shown in Table I:

Table I
FUZZY INTERPRETATION EXTENSION

$I(\phi \wedge \psi) = I(\phi) \otimes I(\psi)$	$I(\phi \vee \psi) = I(\phi) \oplus I(\psi)$
$I(\phi \rightarrow \psi) = I(\phi) \Rightarrow I(\psi)$	$I(\neg \phi) = \ominus I(\phi)$
$I(\exists x.\phi(x)) = \sup_{c \in \Delta^I} I(\phi(c))$	$I(\forall x.\psi(x)) = \inf_{c \in \Delta^I} I(\phi(c))$

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

Several t-norms, s-norms, implication functions, and negation functions have been introduced in the literature. Table II lists some important properties that all functions should satisfy. Three important fuzzy logics are shown in Table III.

Usually the implication function $\Rightarrow$ is defined as *r-implication*, which means $a \Rightarrow b = \sup\{c | a \otimes c \leq b\}$. It is easy to see that in the three logics in Table III, the implication functions are r-implications relative to their t-norms. Under r-implication function relative to a t-norm $\otimes$, we know the following hold:

- If $a \geq n$ and $a \Rightarrow b \geq m$, then we have $b \geq n \otimes m$,
- If $a \Rightarrow b \geq n$ and $b \Rightarrow c \geq m$, then we have $a \Rightarrow c \geq n \otimes m$,
- $n \Rightarrow (n \otimes m) \geq m$.

These are the main inference patterns we will use in the rest of this paper.

Table II
PROPERTIES FOR COMBINATION FUNCTION

Axiom Name	T-norm
Tautology/Contradiction	$a \otimes 0 = 0$
Identity	$a \otimes 1 = a$
Commutativity	$a \otimes b = b \otimes a$
Associativity	$(a \otimes b) \otimes c = a \otimes (b \otimes c)$
Monotonicity	if $b \leq c$, then $a \otimes b \leq a \otimes c$
	Implication Function
Tautology/Contradiction	$0 \Rightarrow b = 1$, $a \Rightarrow 1 = 1$, $1 \Rightarrow 0 = 0$
Antitonicity	if $a \leq b$, then $a \Rightarrow c \geq b \Rightarrow c$
Monotonicity	if $b \leq c$, then $a \Rightarrow b \leq a \Rightarrow c$

Table III
COMBINATION FUNCTIONS OF VARIOUS FUZZY LOGICS

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

Furthermore we can use implication functions and t-norms to define the degree of subsumption between fuzzy sets and the degree of composition of fuzzy relations.

A fuzzy set R is defined as a function $R : X \rightarrow [0, 1]$, where X is a countable crisp set X. The degree of subsumption between two fuzzy sets A and B is defined as

$$A \sqsubseteq B = \inf_{x \in X} A(x) \Rightarrow B(x),$$

where $\Rightarrow$ is an r-implication function.

A (binary) fuzzy relation R is defined as a function $R : X \times Y \rightarrow [0, 1]$ over two countable crisp sets X and Y. The composition of two fuzzy relations $R_1 : X \times Y \rightarrow [0, 1]$ and $R_2 : Y \times Z \rightarrow [0, 1]$ is defined as $(R_1 \circ R_2)(x, z) = \sup_{y \in Y} R_1(x, y) \otimes R_2(y, z)$. We say a fuzzy relation R is transitive if and only if $R(x, z) \geq (R \circ R)(x, z)$, and reflexive if and only if $R(x, x) = 1$.

IV. FUZZY RDF GRAPH AND FUZZY pD^* SEMANTICS

In this section, we first extend the definition of generalized RDF graph in [3] to define fuzzy RDF graph in Section IV-A. Then we provide fuzzy pD^* semantics in Section IV-B.

A. Syntax of fuzzy RDF graph

An *fuzzy triple* is an expression $t[n]$ where t is a triple and $v \in (0, 1]$ is called the *fuzzy degree*. A *fuzzy RDF graph* is defined as a set of fuzzy triples. Here we require that the fuzzy degree $n > 0$, since $n = 0$ means that this fuzzy triple is not in the graph.

A *subgraph* G' of a fuzzy graph G is a fuzzy graph if for every $(s, p, o)[n] \in G'$ there is a fuzzy triple $(s, p, o)[m] \in G$ such that $n \leq m$.

We call the triple set $crisp(G) = \{(s, p, o) : (s, p, o)[n] \in G\}$ the crisp projection of G. Furthermore, we define the set of RDF terms of G as $T(G) = T(crisp(G))$, the set of blank nodes of G as $bl(G) = bl(crisp(G))$, and the vocabulary of G as $V(G) = V(crisp(G))$. If a fuzzy graph has no blank node, we say it is *ground*.

Example 2: The following fuzzy RDF graph describes the vague information given in Example 1. For example, triple (example:pic1, rdf:type, example:BrightPicture)[0.7] represents that the pic1 is a bright picture to a degree of 0.7. The last triple does not correspond to a statement mentioned in E.g. 1, but according to our domain knowledge, we know the property "contain" should be treated as a transitive property.

(pic1, type, BrightPicture)[0.7]
(pic1, sameAs, pic2)[0.9]
(pic2, contain, pic3)[0.6]
(pic3, contain, pic4)[0.5]
(pic2, contain pic4)[0.1]
(pic1, contain pic4)[0.1]
(contain, type, TransitiveProperty)[1]

B. Fuzzy pD^* semantics

We define the fuzzy pD^* semantics. We use $\Rightarrow$ as an r-implication relative to a t-norm $\otimes$.

A *fuzzy interpretation* I of a vocabulary V is a tuple $I = \langle R_I, P_I, E_I, S_I, L_I, LV_I \rangle$, where R_I, P_I, E_I, S_I, L_I and LV_I are exactly as for the crisp case (see III-A), while E_I is the extension function for which item 4 is changed as follows:

4' E_I maps every property $p \in P_I$ to a total function $E_I(p) : R_I \times R_I \to [0, 1]$, which assigns a fuzzy degree to each pair of resources, denoting the degree that the pair is an instance of the property p;

Unlike a simple interpretation in the crisp RDF semantics, the extension function E_I of a fuzzy interpretation maps a property p into a function assigning each resource pair with a fuzzy degree.

Similar to the crisp case, a fuzzy interpretation I of a vocabulary V can also be treated as a function with domain V as in Eq. (1). Moreover, given a partial function $A : \mathbf{B} \rightharpoonup R_I$, the function I_A is defined in the same way as in Eq. (2). We say I satisfies a fuzzy RDF graph G if there is some function $A : bl(G) \to R_I$, such that for each triple $(s, p, o)[n] \in G$, we have $I_A(p) \in P_I$ and $E_I(I_A(p))(I_A(s), I_A(o)) \geq n$.

Now we are ready to define the fuzzy pD^* interpretation. Without loss of clarity, we omit the namespace rdf, rdfs and owl in the following.

We first use the extension function to define the class set and class extension function. Let V be a vocabulary, and I a fuzzy interpretation of $V \cup \{\texttt{type}, \texttt{Class}\}$ such that $I(\texttt{type}) \in P_I$. The class set C_I in I is defined as follows

$$C_I = \{a \in R_I : E_I(I(\texttt{type}))(a, I(\texttt{Class})) > 0\}$$

and the class extension function CE_I maps each class $a \in C_I$ to a total function $CE_I(a) : R_I \to [0, 1]$ where

$$CE_I(a)(b) = E_I(I(\texttt{type}))(b, a)).$$

Suppose D is a datatype map, a fuzzy pD^* interpretation over a vocabulary V is a fuzzy interpretation I over $V \cup$ *rdfV* $\cup$ *rdfsV* $\cup$ $\mathbf{dom}(D) \cup pV$ that satisfies all *fuzzy pD^* constraints*. For the space limitation, Table IV lists parts of these constraints. The full list of fuzzy pD^* constraints can be found in [16].

We explain constraint 7 for `domain` here by considering Table I, while others can be explained similarly. In the crisp case, the constraint for `domain` is stated as follows: If $(a, b) \in E_I(I(\texttt{domain})$, then $\forall e, f \in R_I, E_I(a)(e, f) \Rightarrow CE_I(b)(e)$. In the fuzzy case, $\forall$ corresponds to inf, and the statement "If ϕ then ψ" is translated as $I(\phi) \leq I(\psi)$. Then we get constraint 7 immediately.

We define the fuzzy pD^* entailment relation by using the notion of a pD^* interpretation. We say that a fuzzy graph S fuzzily pD^* entails a fuzzy graph G, denoted as $S \models_{fp} G$, if each pD^* interpretation I that satisfies S also satisfies G.

V. Fuzzy pD^* entailment

In this section, we first give the pD^* entailment rules. Then, we define the Best Degree Bound(BDB) of a triple, and the BDB problem that will be caused by forward chain reasoning under fuzzy semantics. We also define the partial and full fuzzy pD^* closure and show that the partial fuzzy pD^* closure can be calculated in polynomial time. Finally, we discuss the consistency checking and entailment problems and provide the complexity results under fuzzy pD^* semantics.

A. Fuzzy pD^* entailment rules

Given a datatype map D, some of fuzzy pD^*-entailment rules corresponding to Table IV are listed in Table V. The full fuzzy pD^*-entailment rule set can be found in [16]. Formally speaking, given a fuzzy RDF graph G, a fuzzy pD^* interpretation I that satisfies G, if we can apply a pD^* entailment rule over G to derive some new triple $(s, p, o)[n]$, then I also satisfies $G \cup \{(s, p, o)[n]\}$.

We explain rule f-rdfs2, other rules can be explained similarly. As we have discussed in the last section, the constraint for `domain` is:

$$E_I(I(\texttt{domain}))(a, b) \leq \inf_{(e,f) \in R_I \times R_I} E_I(a)(e, f) \Rightarrow CE_I(b)(e)$$

Therefore, if a fuzzy interpretation I satisfies $(p, \texttt{domain}, u)[n]$ and $(v, p, w)[m]$ in G, then we know there is some function $A : bl(G) \to R_I$, such that $E_I(I(\texttt{domain}))(I_A(p), I_A(u)) \geq n$ and $E_I(I_A(p))(I_A(v), I_A(w)) \geq m$. Then according to the `domain` constraint we have $E_I(I_A(p))(I_A(v), I_A(w)) \Rightarrow CE_I(I_A(u))(I_A(v))$

Table IV
FUZZY pD^* CONSTRAINT (PART)

1	I satisfies all fuzzy RDF, RDFS, D and P axiomatic triples mentioned in [3] with fuzzy degree 1
2	$a \in P_I$ if and only if $E_I(I(\texttt{type}))(a, I(\texttt{Property})) = 1$
7	If $a \in P_I$, $b \in C_I$, then $E_I(I(\texttt{domain}))(a, b) \leq \inf_{(e,f)\in R_I \times R_I} E_I(a)(e, f) \Rightarrow CE_I(b)(e)$
16	For each pair $(a, d) \in D$, $I(a) = d$
17	If $l = (s, a) \in V$, $(a, d) \in D$ and $s \in L(d)$, then $L_I(l) = L2V(d)(s) \in \text{LV}_I$ and $E_I(I(\texttt{type}))(L_I(l), d)) = 1$
18	If $l = (s, a) \in V$, $(a, d) \in D$ and $s \notin L(d)$ then $L_I(l) \notin \text{LV}_I$ and $E_I(I(\texttt{type}))(L_I(l), d) = 0$
22	If $CE_I(I(\texttt{TransitiveProperty}))(p) > 0$, then $P \in P_I$, and $CE_I(I(\texttt{TransitiveProperty}))(p) \leq \inf_{(a,b,c)\in R_I\times R_I\times R_I} (E_I(p)(a, b) \otimes E_I(p)(b, c)) \Rightarrow E_I(p)(a, c)$
23	$E_I(I(\texttt{sameAs}))$ is an equivalence relation on R_I
24	If $a \in C_I$ then $E_I(I(\texttt{sameAs}))(a, b) \leq E_I(I(\texttt{subClassOf}))(a, b)$
25	If $p \in P_I$ then $E_I(I(\texttt{sameAs}))(p, q) \leq E_I(I(\texttt{subPropertyOf}))(p, q)$
26	If $p \in P_I$ then $E_I(p)(a, b) \leq \inf_{(a',b')\in R_I\times R_I} (E_I(I(\texttt{sameAs}))(a, a') \otimes E_I(I(\texttt{sameAs}))(b, b')) \Rightarrow E_I(p)(a', b')$
27	If $E_I(I(\texttt{inverseOf}))(p, q) > 0$ then $p \in P_I$ and $q \in P_I$ and $E_I(I(\texttt{inverseOf}))(p, q) \leq \inf_{(a,b)\in R_I\times R_I} \min(E_I(p)(a, b) \Rightarrow E_I(q)(b, a), E_I(q)(b, a) \Rightarrow E_I(p)(a, b))$

$\geq E_I(I(\texttt{domain}))(I_A(p), I_A(u)) \geq n$.
Therefore we have

$$\begin{aligned} & CE_I(I_A(u))(I_A(v)) \\ \geq\ & E_I(I_A(p))(I_A(v), I_A(w)) \otimes \\ & \quad (E_I(I_A(p))(I_A(v), I_A(w)) \Rightarrow CE_I(I_A(u))(I_A(v))) \\ \geq\ & n \otimes m \end{aligned}$$

Thus we know I must satisfy $G \cup \{(v, \texttt{type}, u)[n \otimes m]\}$.

For each rule, we call the set of triples in the condition part whose fuzzy degrees appear in the conclusion part *entailment part*. The other triples in the condition part constitute the *existence part* of the rule. Then the fuzzy degree of the conclusion part is the t-norm conjunction over the fuzzy degrees in the entailment part. For example, for rule f-rdfs2, the two fuzzy triples $(p, \texttt{domain}, u)[n]$ and $(v, p, w)[m]$ together constitute the entailment part, because the fuzzy degree of the conclusion part is $n \otimes m$. The existence part of rule f-rdfs2 is empty. In rule f-rdf1, the entailment part is empty, and the existence part contains one triple $(v, p, w)[n]$. By convention, the t-norm conjunction over the empty set is 1. As we will see later, because of the existence part of a rule, it is non-trivial to show the computational complexity of the partial fuzzy pD^* closure.

Example 3: (Example 2 continued) We illustrate how fuzzy triples (pic2, type, BrightPicture)[0.63] and (pic2, contain, pic4)[0.3] can be entailed by considering the Product t-norm. First, (BrightPicture, sameAs, BrightPicture)[1] can be entailed by applying rule f-rdfp5b over (pic1, type, BrightPicture)[0.7]. By applying rule f-rdfp11 to triples (BrightPicture, sameAs, BrightPicture)[1], (pic1, sameAs, pic2)[0.9] and (pic1, type, BrightPicture)[0.7], we obtain triple (pic2, type, BrightPicture)[0.63]. Then, by applying f-rdfp4 to triples (pic2, contain, pic3)[0.6], (pic3, contain, pic4)[0.5] and (contain, type, TransitiveProperty)[1], we can get (pic2, contain, pic4)[0.3].

B. Partial and full fuzzy pD^* closure

We first define the notion of a derivation of a fuzzy triple, then we give the definition of partial and full fuzzy pD^* closure.

Definition 1: (Derivation) Given a fuzzy RDF graph G, a fuzzy triple E, a *derivation* of E from G is a finite sequence of fuzzy triples $E_1, ..., E_l$, such that $E_l = E$ and each $i \leq l$, either $E_i \in G$ or E_i is obtained applying a fuzzy pD^* entailment rule to one or more fuzzy triples E_j such that $j < i$. If there is a derivation of E, we say E can be derived from G.

Example 4: Consider the fuzzy graph mentioned in Example 2, a sequence of four fuzzy triples (pic2, type, pic3)[0.6], (pic3, type, pic4)[0.5], (contain, type, TransitiveProperty)[1], (pic2, contain, pic4)[0.3] is a derivation of (pic2, contain, pic4)[0.3], since the first three fuzzy triples are in the graph, and the last can be derived from the first one using rule f-rdfp4 as explained in Example 3. Furthermore the sequence consisting of one fuzzy triple (pic2, contain, pic4)[0.1] is also a derivation of (pic2, contain, pic4)[0.1], since the only fuzzy triple is originally in the graph.

In a derivation of E, if there is a fuzzy triple appearing more than once, we can safely remove the later appearance from the derivation. So in the following we only consider the derivation contains no duplicated fuzzy triples

We can see that a triple (s, p, o) may be derived from a fuzzy RDF graph with different fuzzy degrees. We define the derived degree set $N(s, p, o) = \{n : (s, p, o)[n]$ can be derived from $G\}$. We show that $N(s, p, o)$ can be infinite. Suppose we use the Product t-norm $a \otimes b = a \cdot b$, and we consider the following two triples $(a, \texttt{sameAs}, a')[n]$ and $(a, p, b)[m]$. By applying rule f-rdfp5b over these two triples, we get $(b, \texttt{sameAs}, b)[1]$. We then apply rule f-rdfp11 over these three triples and obtain $(a', p, b)[nm]$. After that, by applying rule f-rdfp11 over $(a, \texttt{sameAs}, a')[n]$ and the last two fuzzy triples, we can derive $(a, \texttt{sameAs}, b)[n^2m]$. Similarly, we can derive $(a', p, b)[n^{2k+1}m]$ and $(a, p, b)[n^{2k}m]$ for any non-negative integer k. Therefore $\{n^{2k}m : k = 0, 1, ...\} \subseteq N(a, p, b)$, and if $n < 1$, then $N(a, p, b)$ is infinite.

However, for the two fuzzy triples sharing the same triple part $(s, p, o)[n]$ and $(s, p, o)[m]$ where $m < n$, it is easy

Table V
FUZZY pD^*-ENTAILMENT RULES (PART)

	Condition	Constraint	Conclusion
f-rdf1	$(v,p,w)[n]$		$(p,\mathtt{type},\mathtt{Property})[1]$
f-rdfs2	$(p,\mathtt{domain},u)[n]$ $(v,p,w)[m]$		$(v,\mathtt{type},u)[n\otimes m]$
f-rdfp4	$(p,\mathtt{type},\mathtt{TransitiveProperty})[n]$		
	$(u,p,v)[m]$ $(v,p,w)[l]$		$(u,p,w)[n\otimes m\otimes l]$
f-rdfp5a	$(v,p,w)[n]$		$(v,\mathtt{sameAs},v)[1]$
f-rdfp5b	$(v,p,w)[n]$		$(w,\mathtt{sameAs},w)[1]$
f-rdfp6	$(v,\mathtt{sameAs},w)[n]$	$w\in U\cup B$	$(w,\mathtt{sameAs},v)[n]$
f-rdfp7	$(u,\mathtt{sameAs},v)[n]$ $(v,\mathtt{sameAs},w)[m]$		$(u,\mathtt{sameAs},w)[n\otimes m]$
f-rdfp11	$(u,p,v)[n]$ $(u,\mathtt{sameAs},u')[m]$ $(v,\mathtt{sameAs},v')[l]$	$u'\in U\cup B$	$(u',p,v')[n\otimes m\otimes l]$

to see if a fuzzy interpretation satisfies $(s,p,o)[n]$, then it also satisfies both of them. So we can safely remove the one with lower fuzzy degree. This motivates us to define the Best Degree Bound of a triple as follows.

Definition 2: (Best Degree Bound(BDB)) Given a fuzzy RDF graph G, the *Best Degree Bound* $n\in[0,1]$ of a triple (s,p,o) derived from G is $n>0$ such that $(s,p,o)[n]$ can be derived from G and any fuzzy triple $(s,p,o)[m]$ that can be derived from G satisfies $m\leq n$; or $n=0$ if none of such derivation exists.

Example 5: Given the fuzzy RDF graph mentioned in Example 2. Let us consider the BDB of triple (pic2, contain, pic4). As we discussed in Example 4, both (pic2, contain, pic4)[0.3] and (pic2, contain, pic4)[0.1] can be derived from the graph. However we can see $n=0.3$ is the largest n that (pic2, contain, pic4)[n] can be derived. So the BDB of triple (pic2, contain, pic4) is 0.3.

Now there is a tricky question. Consider a triple (s,p,o), its derived degree set $N(s,p,o)$ may be infinite. If $\sup N(s,p,o)\notin N(s,p,o)$, which means for any $n\in N(s,p,o)$, we can always find some $m\in N(s,p,o)$ such that $m>n$, then when we use forward chain reasoning method to apply fuzzy pD^* rules to calculate the closure iteratively, this process will never terminate. In other words, if there is a triple whose BDB does not exists, then we cannot use forward chain reasoning method to solve the entailment problem. We can define the BDB problem as follows.

Problem 1: (BDB problem) Given a fuzzy RDF graph G and an arbitrary triple (s,p,o), is there always a BDB for (s,p,o) derived from G?

Luckily, we can prove the following lemma, which makes forward chain reasoning possible for fuzzy pD^* entailment problem. Although the proof is non-trivial, we do not provide it here for the space limitation. We refer the readers to our technical report [16].

Lemma 1: (BDB existence lemma) Given a fuzzy RDF graph G, for any triple (s,p,o), the BDB of (s,p,o) derived from G always exists.

Now we can define the partial and full fuzzy pD^* closure.

Definition 3: (Partial and full fuzzy pD^* closures) Suppose G is a fuzzy RDF graph, D a datatype map, and K a non-empty subset of the positive integers $\{1,2,...\}$ satisfying $i\in K$ for each $\mathtt{rdf}:\_i\in V(G)$. The *partial fuzzy pD^* closure* $G_{fp,K}$ is defined in the following way: In the first step, all RDF, RDFS, D and P axiomatic fuzzy triples are added to G except for those containing URI references $\mathtt{rdf:}\_i$ where $i\notin K$. Then we apply rule f-lg to every triple to replace the literal l with a blank node b_l so that distinct well-typed D-literals with the same value is replaced with the same blank node. Finally arbitrary derivations are made using applications of the fuzzy pD^* entailment rules or fuzzy P entailment rules until the graph is unchanged. In this process, if there are two fuzzy triples $(s,p,o)[n]$ and $(s,p,o)[m]$ in the closure, then eliminate one with lower fuzzy degree. In addition, the full fuzzy pD^* closure G_{fp} of G is defined by taking $G_{fp}=G_{fp,K}$, where $K=\{1,2,...\}$.

From now on we call a fuzzy triple $(s,p,o)[n]$ a BDB fuzzy triple if n is the BDB of (s,p,o). According to the BDB existence lemma, we know the partial fuzzy pD^* closure $G_{fp,K}$ of a graph G always exists and contains only BDB fuzzy triples. Since for each triple (s,p,o) there is at most one BDB fuzzy triple $(s,p,o)[n]$ in $G_{fp,K}$, we know the size of $G_{fp,K}$ is polynomial to $|G|$, $|K|$ and $|D|$.

Next, we consider the computational complexity of calculating $G_{fp,K}$. In the crisp case, since each triple can be added to the partial pD^* closure $G_{p,K}$ at most once, the number of successful rule applications to obtain new triples will be at most $|G_{p,K}|$. Furthermore, for each rule, whether a successful rule application exists can be determined within polynomial time in $|G_{p,K}|$, and $|G_{p,K}|$ can be bounded by a polynomial function of $|G|$, $|D|$ and $|K|$. So the partial closure for the crisp RDF graph can be computed within polynomial time. We can see that in the crisp case, the key point is that every triple will be added at most once. But this is not true in the fuzzy case due to the existence part of some rules. For example, if $G=\{(s,\mathtt{type},\mathtt{Property})[n]\}$ where $n<1$, then in the derivation of $(s,\mathtt{type},\mathtt{Property})[1]$, the fuzzy triple $(s,\mathtt{type},\mathtt{Property})[n]$ must appear. So, given a generalized RDF graph, to derive a BDB fuzzy triple $t[n]$, another fuzzy triple $t[m]$ $(m\neq n)$ may be derived. On the other hand, we notice that, in our previous example, triple $(s,\mathtt{type},\mathtt{Property})$ only appears twice in the derivation of the BDB fuzzy triple $(s,\mathtt{type},\mathtt{Property})[1]$: one is $(s,\mathtt{type},\mathtt{Property})[n]$ which is in graph G, and the other is $(s,\mathtt{type},\mathtt{Property})[1]$ which is the BDB fuzzy triple. This observation inspires us to consider such kind of derivation in which every fuzzy triple is either originally in the graph or a BDB fuzzy triple. We can see that in such a derivation, each triple appears at most twice, and thus the length of such a derivation can be bounded by a polynomial function of $|G|$, $|D|$ and $|K|$ since the number of different triples can be bounded by a polynomial function of $|G|$, $|D|$ and $|K|$. However, for those BDB fuzzy triples whose triple parts do not appear in the original graph, such derivation may not exists. For example, given a graph $G = \{(p, \mathtt{type}, P)[n], (P, \mathtt{subClassOf}, \mathtt{Property})[m]\}$, in any derivation of the BDB fuzzy triple $(p, \mathtt{type}, \mathtt{Property})[1]$, fuzzy triple $(p, \mathtt{type}, \mathtt{Property})[n \otimes m]$ which is neither in the original graph nor a BDB fuzzy triple must appear. The solve this problem, we consider separating the partial closure computation into two parts, calculating the potential BDB fuzzy degrees and deriving new triples. By iteratively applying these two kinds of computation we finally get the partial closure. Based on this idea, we define the notions of *local BDB* as the potential BDB, and the *local BDB derivation* so that every triple in such a derivation is either in the original graph or a fuzzy triple with local BDB fuzzy degree. In the following we first formally define the notions of *local BDB* and *local BDB derivation*.

Definition 4: (Local BDB) Given a fuzzy triple set G, for each triple $t = (s, p, o) \in crisp(G)$, the *local BDB* of t is the maximal value $n > 0$ that there is a derivation $E_1 ... E_k$ of $t[n]$ from G that satisfies that $E_i = (s_i, p_i, o_i) \in crisp(G)$ for $i = 1, 2, ..., k$. Since $t \in crisp(G)$, we know the local BDB of t cannot be 0.

Similar to the proof of Lemma 1, we can prove that the local BDB of each triple always exists. From now on we say that a fuzzy triple $t[n]$ is a local BDB fuzzy triple in G if n is the local BDB of t derived from G.

Definition 5: (Local BDB derivation) Given a fuzzy triple set G, for each triple $t = (s, p, o) \in crisp(G)$, a *local BDB derivation* of t from G is a derivation $E_1 ... E_k$ of $t[n]$ from G so that

- n is the local BDB of t from G;
- The triple part of E_i is in $crisp(G)$ for $i = 1, 2, ..., k$;
- For each E_i, either $i = k$ or E_i is used to derive some E_j where $j > i$ by applying some rule $r \in \Gamma$;
- Each E_i is either a local BDB fuzzy triple from G or never be used as the entailment part of any rule to derive later fuzzy triples and
- For those E_i never used as the entailment part, we further restrict $E_i \in G$.

Example 6: Given the fuzzy graph G in Example 2. Let us consider the local BDB derivation of triple t =(pic1, contain, pic4). We know the derivation L of $t[0.1]$, which consisting of $t[0.1]$, is a derivation from G satisfying all constraints of a local BDB derivation. Notice a sequence L' of seven fuzzy triples (pic2, type, pic3)[0.6], (pic3, type, pic4)[0.5], (contain, type, TransitiveProperty)[1], (pic2, contain, pic4)[0.3], (pic1, sameAs, pic2)[0.9], (pic4, sameAs, pic4)[1], (pic1, sameAs, pic4)[0.27] is a derivation of the fuzzy triple (pic1, sameAs, pic4)[0.27]. Although $0.27 > 0.1$, we can see the triple part $(pic4, sameAs, pic4)$ of the 6th fuzzy triple in L' is not in G, thus L' is not a local BDB derivation. So the local BDB of t is 0.1, and L is a local BDB derivation of t.

The first three constraints are directly derived from the definitions of local BDB and derivation. The fourth and fifth constraints seems less straightforward. However, from these two constraints, we can directly conclude that each fuzzy triple in a local BDB derivation of t from G is either a fuzzy triple contained in G or a local BDB fuzzy triple from G. Furthermore, in constraint four, we restrict that those fuzzy triples which are not local BDB fuzzy triples must not be used as the entailment part. Based on these constraints, we can prove the following two lemmas.

Lemma 2: (Local BDB derivation length) Given a fuzzy triple set G, for each triple $t = (s, p, o) \in crisp(G)$, the length of a local BDB derivation of t from G is at most $2|G|$.

Lemma 3: (Local BDB derivation existence) Given a fuzzy triple set G, for each triple $t = (s, p, o) \in crisp(G)$, there is a local BDB derivation of t from G.

The non-trivial proofs of these two lemmas can be found in [16]. Now we are ready to show that the partial fuzzy pD^* closure can be computed within polynomial time.

Lemma 4: Let D be a datatype map, G a fuzzy triple set, and K a finite non-empty subset of $\{1, 2, ...\}$, the partial fuzzy pD^* closure $G_{fp,K}$ can be computed within polynomial time in $|G|$, $|K|$ and $|D|$.

Since this lemma is a vital lemma to show the complexity result, we sketch the basic idea of its proof here. We iteratively do the following two calculations: (1) calculate the local BDB fuzzy triples from G and (2) check if we can derive a fuzzy triple whose triple part is not in $crisp(G)$ and add it into G. Since the size of G is increased by 1 in each iteration, the number of the iterations is at most $|G_{fp,K}|$. In each iteration, according to our previous discussion (see the paragraph following Definition 3), the calculation (2) can be done within polynomial time. To perform calculation (1), we employ an iterative subroutine. In each iteration, the subroutine tries all possible rule applications over G to update the fuzzy degrees of the fuzzy triples in G. Then each iteration can be done within polynomial time. The key fact is that if there is a derivation with length l of a local BDB fuzzy triple, then this local BDB fuzzy triples will be contained after l iterations G. As we have discussed, every local BDB fuzzy triple has a local BDB derivation with length $2|G|$. Thus, we can calculate all local BDB fuzzy triples from G within $2|G|$ iterations. As a consequence, calculation (1) can be done within polynomial time. Therefore, $G_{fp,K}$ can be computed within polynomial time in $|G|$, $|K|$ and $|D|$. The full proof can be found in [16].

C. Consistency, soundness, completeness and complexity

We define fuzzy D-clash and fuzzy P-clash as follows.

Definition 6: A *fuzzy D-clash* is a fuzzy triple $(b, \texttt{type}, \texttt{Literal})[1]$, where b is a blank node allocated by rule lg to an ill-typed D-literal. A *fuzzy P-clash* is either a combination of two triples of the form $(v, \texttt{differentFrom}, w)[n]$ and $(v, \texttt{sameAs}, w)[m]$ where $n, m > 0$, or a combination of three triples of the form $(v, \texttt{disjointWith}, w)[n]$, $(u, \texttt{type}, v)[m_1]$ and $(u, \texttt{type}, w)[m_2]$ where $n, m_1, m_2 > 0$.

Now we can give the consistency, soundness and completeness results. For the space limitation, we do not give the proofs for the following theorems, and refer the reader to our technical report [16].

Theorem 1: (Consistency) Let G be a fuzzy RDF graph, D a datatype map, and K a non-empty subset of $\{1, 2, ...\}$ containing at least those i that $\texttt{rdf:\_}i \in V(G)$. If the partial fuzzy pD^* closure $G_{fp,K}$ of G does not contain a fuzzy P-clash or a fuzzy D-clash, then there is a fuzzy pD^* interpretation that satisfies G. Otherwise there is no fuzzy pD^* interpretation satisfying G.

Theorem 2: (Soundness and Completeness) Let D be a datatype map, S and G two fuzzy RDF graphs. Let H be a partial fuzzy pD^* closure $S_{fp,K}$ of S where $i \in K$ for each $\texttt{rdf :\_}i \in V(G)$. Then, $S \models_{fp} G$ if and only if H contains an instance of G as a subset or H contains a fuzzy D-clash or a fuzzy P-clash.

Finally, the complexity results follow from Lemma 4, Theorem 1 and 2.

Theorem 3: (Complexity) Let D be a finite datatype map. The consistency checking problem is in P. The fuzzy pD^* entailment relation $S \models_{fp} G$ between two fuzzy RDF graphs S and G is decidable. This problem is $NP\text{-}complete$, and in P if G is ground.

VI. Conclusion and Future Work

In this paper, we proposed fuzzy pD^* semantics based on an arbitrary t-norm function to generalize pD^* semantics. We provided the fuzzy pD^* entailment rules and defined the Best Degree Bound (BDB) of a triple derived from a fuzzy RDF graph. We showed that the BDB of an arbitrary triple from an arbitrary RDF graph always exists. We defined the partial and full fuzzy pD^* closures and showed that the partial fuzzy pD^* closure exists for any fuzzy RDF graph. We then introduced the local BDB derivation of a triple from a fuzzy RDF graph and showed its existence. Based on this result, we proved that the partial fuzzy pD^* closure can be computed within polynomial time. We proved soundness and completeness results for the entailment relation. Finally, we proved that the consistency checking is in P, the entailment is *NP-complete*, and in P if the target fuzzy RDF graph is ground. As a future work, we will develop efficient reasoning algorithms to tackle the large-scale reasoning problem under fuzzy pD^* semantics. We also plan to apply our theoretical work to some interesting applications, such as multimedia.

Acknowledgement

Guilin Qi is partially supported by NSFC (61003157) and National Research Foundation for the Doctoral Program of Higher Education of China (20100092120029).

References

[1] RDF, *http://www.w3.org/RDF/*.

[2] RDFS, *http://www.w3.org/TR/rdf-schema/*.

[3] H. J. Horst, "Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary," *Journal of Web Semantics*, vol. 3, no. 2-3, pp. 79 – 115, 2005.

[4] U. Straccia, "A Minimal Deductive System for General Fuzzy RDF," in *RR'09*, 2009.

[5] M. Mazzieri, A. F. Dragoni, and U. P. D. Marche, "A fuzzy semantics for semantic web languages," in *Proc. URSW 2005*, 2005, pp. 12–22.

[6] M. Mazzieri and A. F. Dragoni, "A Fuzzy Semantics for the Resource Description Framework," in *Proc. URSW 2008*, 2008, pp. 244–261.

[7] O. Udrea, D. Recupero, and V. Subrahmanian, "Annotated RDF," in *Proc. of ESWC 2006*, 2006, pp. 487–501.

[8] U. Straccia, N. Lopes, G. Lukacsy, and A. Polleres, "A General Framework for Representing and Reasoning with Annotated Semantic Web Data," in *Proc. of AAAI 10*, 2010, pp. 1437–1442.

[9] U. S. Nuno Lopes, Axel Polleres and A. Zimmermann, "AnQL: SPARQLing Up Annotated RDF," in *Proc. of ISWC 10*, no. 6496, 2010, pp. 518–533.

[10] G. Stoilos, G. B. Stamou, J. Z. Pan, V. Tzouvaras, and I. Horrocks, "Reasoning with Very Expressive Fuzzy Description Logics," *JAIR*, pp. 273–320, 2007.

[11] U. Straccia, "Reasoning within Fuzzy Description Logics," *JAIR*, pp. 137–166, 2001.

[12] ——, "Fuzzy ALC with Fuzzy Concrete Domains," in *Proc. of DL 05*, 2005.

[13] J. Z. Pan, G. B. Stamou, G. Stoilos, S. Taylor, and E. Thomas, "Scalable querying services over fuzzy ontologies," in *Proc. of WWW 08*, 2008, pp. 575–584.

[14] S. Muñoz, J. Pérez, and C. Gutierrez, "Minimal Deductive Systems for RDF," in *Proc. ESWC 2007*, 2007, pp. 53–67.

[15] P. Hajek, *Metamathematics of Fuzzy Logic*. Kluwer Academic Publishers, 1998.

[16] C. Liu, G. Qi, H. Wang, and Y. Yu, "Fuzzy reasoning over RDF data using OWL vocabulary," Shanghai Jiao Tong University, Tech. Rep., Dececmber 2010, available at http://apex.sjtu.edu.cn/apex_wiki/fuzzypd.