Probabilistic RDF

Octavian Udrea[1,2], V.S. Subrahmanian[1,2], and Zoran Majkić[2]

[1] University of Maryland Computer Science Dept., College Park MD 20742
[2] University of Maryland Institute for Advanced Computer Studies
{udrea,vs,zoran}@cs.umd.edu

Abstract. RDF is rapidly being adopted around the world as a paradigm for knowledge representation. However, we are not aware of any version of RDF that can express probabilistic knowledge. In this paper, we develop a framework called *Probabilistic RDF* (pRDF for short) - we provide a syntax as well as a model theoretic and fixpoint semantics for acyclic pRDF theories. We then provide algorithms to efficiently answer queries posed to pRDF ontologies. We have developed a prototype implementation that shows that our algorithms work very well in practice.

1 Introduction

Over the last few years, the use of RDF as a paradigm for representing knowledge has grown dramatically. RDF and OWL ontologies exist on a wide variety of topics ranging from genetics to visual sensor data fusion. These are clearly domains that are chock full of uncertainty - image processing programs based on Bayesian analysis often return probabilistic identifications of objects, while relationships between a symptom or disease and the genetic markers a person may also be probabilistic.

In order to express such information, we introduce *Probabilistic RDF* (pRDF) for short. We define the concept of a pRDF schema and a pRDF instance in Section 2. A pRDF instance extends RDF triples - it is a set of certain types of quadruples. The rest of the paper focuses on "acyclic" theories. Section 3 provides a formal model theoretic semantics for pRDF which also contains a result saying that pRDF theories that are "large enough" (more precisely have at least 9 quadruples in the pRDF instance) are guaranteed to be consistent. Section 4 shows that we can associate a monotonic function with any pRDF theory — this function has a least fixpoint that compactly represents the set of all quadruples entailed by the pRDF theory. However, using the fixpoint to answer queries is not always desirable because the size of the fixpoint can be enormous. Section 5 provides algorithms to efficiently answer atomic queries, while Section 6 provides algorithms to compute conjunctive queries. Section 7 describes our prototype implementation together with experiments showing that queries can be answered in very small amounts of time (a few seconds) for pRDF instances as large as 100,000 quadruples.

2 pRDF syntax

We now develop a formal syntax for pRDF. We assume the existence of three basic sets: $\mathcal{U}$ is the set of URI references, $\mathcal{L}$ is the set of literals (primitive data values) and $\mathcal{B}$ is the set of blank nodes such that $\mathcal{U} \cap \mathcal{B} = \emptyset$. We also assume the following are arbitrary but fixed:

(i) $\mathcal{C} \subseteq \mathcal{U}$ is the set of classes.
(ii) $\mathcal{P} \subseteq \mathcal{U}$ is the set of properties. Our approach distinguishes between transitive and non-transitive properties. We assume that a set $\mathcal{P}^t \subseteq \mathcal{P}$ of *transitive* properties is specified, where $\mathcal{P}^t \cup \mathcal{P}^{nt} = \mathcal{P}$ and $\mathcal{P}^t \cap \mathcal{P}^{nt} = \emptyset$.
(iii) $\mathcal{I} \subseteq \mathcal{U} \cup \mathcal{B}$ is a set of instances (individuals).

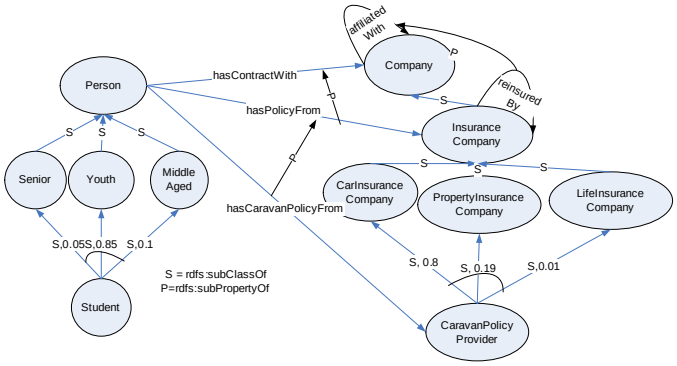


Fig. 1. pRDF schema example

We now define a pRDF schema.

Definition 1 (pRDF schema). *A* probabilistic RDF schema *is a finite set S of elements in one of the following forms:*

(S1) Probabilistic quadruples *are of the form $(s, rdfs : subClassOf, O, \delta)$, where $s \in \mathcal{C}, O \subseteq \mathcal{C}$ and $\delta : O \to (0, 1]$ a probability distribution over O. We require that:*
  (S1.1) $\sum_{v \in O} \delta(v) = 1$.
  (S1.2) $\forall\, ((s, rdfs : subClassOf, O_1, \delta_1), (s, rdfs : subClassOf, O_2, \delta_2) \in S$ *such that* $O_1 \neq O_2)$ *then* $O_1 \cap O_2 = \emptyset$.
(S2) Non-probabilistic triples *are in one of the following forms:*
  (S2.1) $(p_1, rdfs : subPropertyOf, p_2)$, *where* $p_1, p_2 \in \mathcal{P}$.
  (S2.2) $(p, rdfs : range, c)$, *where* $p \in \mathcal{P}, c \in \mathcal{C}$.
  (S2.2) $(p, rdfs : domain, c)$, *where* $p \in \mathcal{P}, c \in \mathcal{C}$.

Example 1. Figure 1 contains a graph representation of an pRDF schema. *(Student, rdfs:subClassOf,* $\{Senior, Youth, MiddleAged\}$*,* $\{.05, .85, .1\}$*)* is a probabilistic schema quadruple stating that any individual belonging to class *Student* belongs to class *Senior* with .05 probability or to class $Youth$ with .85 probability or to class $MiddleAged$ with .1 probability. *(hasPolicyFrom, rdfs:subPropertyOf, hasContractWith)* is a non-probabilistic triple. The rest of the paper will mostly focus on pRDF instances, however we should point out that schema queries are a straightforward extension of pRDF instance queries.

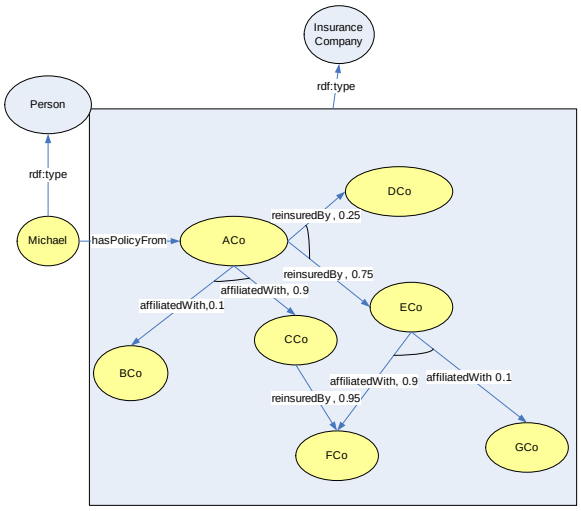


Fig. 2. pRDF instance example

Definition 2 (pRDF instance). *A* probabilistic RDF instance R *is a finite set of probabilistic quadruples in one of the following forms:*

(I1) (s, p, V, δ)*, where* $s \in \mathcal{I}$*,* $p \in \mathcal{P}$*,* $V \subseteq \mathcal{I} \cup \mathcal{L}$*,* $\delta : V \rightarrow (0, 1]$ *such that:*

(I1.1) $\sum_{v \in V} \delta(v) \leq 1$.

(I1.2) $\forall\ (s, p, V_1, \delta_1), (s, p, V_2, \delta_2) \in R$ *such that* $V_1 \neq V_2$*,* $V_1 \cap V_2 = \emptyset$.

(I2) $(s, rdf : type, V, \delta)$*, where* $s \in \mathcal{I}$*,* $V \subseteq \mathcal{C}$ *such that:*

(I2.1) $\sum_{v \in V} \delta(v) \leq 1$.

(I2.2) $(s, rdf : type, V_1, \delta_1), (s, rdf : type, V_2, \delta_2) \in R \Rightarrow (V_1 = V_2) \wedge (\delta_1 = \delta_2)$.

Example 2. Consider the scenario depicted in Figure 2. Michael wants to choose an insurance policy that would put his mind at ease. The market is not entirely stable: knowledge about which company is backed by whom is important in making his choice — however, he has only incomplete information about the

relationships and affiliations between insurance companies. His current policy is with ACo. The graph depicted in Figure 2 is a pRDF instance. For example, the quadruple *(ECo, affiliatedWith, {FCo, GCo},{.9,.1})* states that ECo is affiliated with FCo with 90% probability and to GCo with 10% probability.

We will often abuse notation and talk about quadruples in a pRDF instance having the form (s, p, V, δ) where p is either a property or $rdf : type$, with the restrictions imposed. This allows us to talk about both kinds of quadruples in a pRDF instance in a unified way. We also use $rdfs : subPropertyOf^*$ to denote the transitive closure of the $rdfs : subPropertyOf$ relation.

Definition 3 (pRDF theory). *A pRDF theory is a pair (S, R), where S is a pRDF schema and R is a pRDF instance.*

Example 3. The pRDF schema in Figure 1 and the pRDF instance in Figure 2 form a pRDF theory.

We now define the important concept of a p-path.

Definition 4 (p-path). *Let (S, R) be a pRDF theory and let $p \in \mathcal{P}^t$. Let P be a sequence of quadruples $(s_i, p_i, v_i, \gamma_i)$, for $i = 1, \ldots, n$, where:*

(i) $\forall\ i \in [1, n], \exists\ (s_i, p_i, V, \delta) \in R$ *s.t.* $v_i \in V \wedge \delta(v_i) = \gamma_i$.
(ii) $\forall\ i \in [1, n], p_i\ rdfs : subPropertyOf^*\ p$.
(iii) $\forall\ i \in [1, n-1], v_i = s_{i+1}$.

*In this case, P is a p-*path *of length n with origin s_1 and destination v_n: we denote P by $\langle s_i, p_i, v_i, \gamma_i \rangle_{i \in [1,n]}$. We use $org(P)$ and $dest(P)$ to denote s_1 and v_n respectively.*

Example 4. Consider the pRDF instance in Figure 2. There are two distinct $affiliatedWith$-paths between ACo and FCo, the first one containing the node ECo and the second containing CCo. Note that these are $affiliatedWith$-paths because of the schema triple *(reinsuredBy, rdfs:subPropertyOf, affiliatedWith)* depicted in Figure 1.

A pRDF instance is *acyclic* iff for all properties p, there are no cyclic p-paths in it.

Assumption. Throughout this paper, we assume that all pRDF theories are acyclic. [3]

We recall the notion of a triangular norm [2] that is often used to compute the probability of a conjunction.

Definition 5 (t-norm). *A* triangular norm *(t-norm for short) is any binary function $\otimes$ from $[0, 1] \times [0, 1]$ to $[0, 1]$ such that:*

[3] This is a reasonable assumption - in [1], we show that RDF theories with cycles involving properties have major semantic problems. Space constraints prevent us from going through these details.

(i) $\otimes$ *is associative and commutative;*
(ii) $\forall\ x, y, z, w \in [0,1]$ *s.t.* $x \leq y$ *and* $z \leq w$, $x \otimes z \leq y \otimes w$.
(iii) $0 \otimes x = 0$;
(iv) $1 \otimes x = x$.

Intuitively, suppose we know that the probability of events e_1, e_2 are P_1, P_2 — depending upon our knowledge of the relationships between e_1, e_2, we could choose a t-norm $\otimes$ to compute a value for $e_1 \wedge e_2$. For a t-norm $\otimes$, we will also denote by $min_{\otimes}(p, p_t) = min(\{q | p \otimes q \geq p_t\})$. As a convention, if $\{q | p \otimes q \geq p_t\} = \emptyset$, then we write $min_{\otimes} = 1$. The following result is straightforward.

Proposition 1. $\forall\ p, \forall\ p_t\ p_t \leq min_{\otimes}(p, p_t)$.

Example 5. A few famous examples of t-norms include:

(i) Product: $x \otimes y = xy$;
(ii) Weak Lukasiewicz t-norm: $x \otimes y = min(x, y)$;
(iii) Strong Lukasiewicz t-norm: $x \otimes y = max(x + y - 1, 0)$.

We should point out that for most t-norms, $min_{\otimes}$ is easy to compute. For instance, in case (i),

$$min_{\otimes}(p, p_t) = \begin{cases} 1 \text{ if } p \leq p_t \\ \frac{p_t}{p} \text{ otherwise} \end{cases}$$

Similarly for case (iii)

$$min_{\otimes}(p, p_t) = \begin{cases} 0 \text{ if } p_t = 0 \\ 1 + p_t - p \text{ otherwise} \end{cases}$$

3 pRDF semantics

We now introduce a declarative model theoretic semantics for pRDF. Given a pRDF quadruple (s, p, V, δ), we know that each RDF-triple of the form (s, p, v) where $v \in V$ has a probability $\delta(v)$ of being true. For any set of such triples (or "world") obtained in this manner from a pRDF theory, we would like to associate a probability. We now define these terms.

Definition 6 (World). *A world W is a set of triples (s, p, v), such that $(s \in \mathcal{I} \wedge p \in \mathcal{P} \wedge v \in \mathcal{I} \cup \mathcal{L}) \vee (s \in \mathcal{I} \wedge p = rdf : type \wedge v \in \mathcal{C})$. We denote the set of possible worlds by $\mathcal{W}$.*

Definition 7 (pRDF interpretation). *A pRDF interpretation is a mapping $I : \mathcal{W} \rightarrow [0,1]$, such that $\sum_{W \in \mathcal{W}} I(W) = 1$.*

A pRDF interpretation postulates that exactly one possible world is the "real" world — uncertainty arises because we don't know which of the possible worlds is the real one. The model of possible worlds in relation to uncertainty is due to Halpern et. al[3].

Example 6. Consider the pRDF theory in Figures 1 and 2. Any subgraph of the graph in Figure 2 is a possible world. These are not all the possible worlds; W={*(DCo, affiliatedWith, GCo)*} is also a world. Intuitively, we would assign a very low probability to such a world, since there is no evidence for the information in W.

Definition 8 (pRDF satisfaction). *An interpretation I* satisfies *a pRDF quadruple* (s, p, V, δ) *iff* $\forall\ v \in V, \delta(v) \leq \sum_{(s,p,v)\in W} I(W)$. *I satisfies a pRDF theory* (S, R) *w.r.t. t-norm* $\otimes$ *iff:*

(i) I satisfies every quadruple in R.
(ii) $\forall\ p-paths\ \langle s_i, p_i, v_i, \gamma_i\rangle_{i\in[1,n]}$ *in* (S, R), $\bigotimes_i \gamma_i \leq \sum_{(s_1,p,v_n)\in W} I(W)$.

We say that (S, R) *is* consistent *iff it has a satisfying interpretation.*

The second condition above uses $\otimes$ to compute the probability of a given p-path[4]. Entailment is defined in the usual way.

Definition 9 (Entailment). *Let* $\otimes$ *be an arbitrary but fixed t-norm.* (S, R) *entails* (s, p, V, δ), *written* $(S, R) \models_\otimes (s, p, V, \delta)$ *iff every satisfying interpretation of* (S, R) *also satisfies* (s, p, V, δ). *Furthermore,* $(S_1, R_1) \models_\otimes (S_2, R_2)$ *iff every satisfying interpretation of* (S_1, R_1) *is a satisfying interpretation of* (S_2, R_2). (S_1, R_1) *is equivalent to* (S_2, R_2), *written* $(S_1, R_1) \equiv_\otimes (S_2, R_2)$ *iff* $(S_1, R_1) \models_\otimes (S_2, R_2)$ *and* $(S_2, R_2) \models_\otimes (S_1, R_1)$.

Example 7. The pRDF theory in Figures 1 and 2 entails *(ACo, affiliatedWith, {GCo},{.075})*. It also trivially entails *(ECo, affiliatedWith, {FCo,GCo},{0.8,0.05})*.

The following important (and rather bizarre) theorem states that as long as the pRDF instance has at least 9 items in it, it is guaranteed to be consistent. Intuitively, this is due to the fact that there is an exponential number of worlds and a polinomial number of values $I(W)$ to be determined for a satisfying interpretation of a pRDF theory (S, R).

Theorem 1. *Every pRDF theory* (S, R) *such that* $|R| \geq 9$ *is consistent w.r.t. any t-norm* $\otimes$.

Proof sketch. Let $\mathcal{W}_R$ be the set of worlds that only contain elements found in some quadruple in R. Let $z = |\{(s, p, v)|\exists\ (s, p, V, \delta) \in R\ s.t.\ v \in V\}|$. Clearly $|\mathcal{W}_R| = 2^z$. Let $I(W) = 0\ \forall\ W \in \mathcal{W} - \mathcal{W}_R$. We want to find a satisfying interpretation for (S, R); we denote $x_W = I(W) \forall W \in \mathcal{W}_R$. We write the following system of equations:

[4] We note that analogous concepts of interpretation and satisfaction can be defined for pRDF schemas (in the same way as RDF model theory defines RDF and RDFS interpretations). However, due to space constraints, we primarily focus on pRDF instances.

(i) $\sum_{W \in \mathcal{W}_R} x_W = 1$;

(ii) $\forall\ s, p, v$ s.t. $\exists (s, p, V, \delta) \in R$ where $v \in V$, $\sum_{(s,p,v) \in W} x_W = \delta(v)$.

(iii) $\forall\ s, p, v, \forall\ p\text{-}paths\ P^j = \langle s_i^j, p_i^j, v_i^j, \gamma_i^j \rangle_{i \in [1, n_j]}$ with $n_j \geq 2$ such that $\forall j, s_1^j = s \wedge v_n^j = v$, let $\lambda(s, p, v) = \max_j(\bigotimes_i \gamma_i^j)$. Then $\sum_{(s,p,v) \in W} x_W = \lambda(s, p, v)$.

Clearly, the system above meets the conditions required by Definition 8. The system of equations has 2^z unknowns. Item (i) contains one equation; item (ii) contains at most z equations; item (iii) contains at most $4z^2$ equations (since there are at most $2z$ individuals and/or data values referenced by R). Therefore, the system has at most $4z^2 + z + 1$ equations and exactly 2^z unknowns. The rank of the matrix of coefficients is identical to the number of equations since:

1. No two equations from item (ii) can have the same coefficients since we required that $\forall\ (s, p, V_1, \delta_1), (s, p, V_2, \delta_2) \in R$, $V_1 = V_2$.
2. No two equations from item (iii) can have the same coefficients from the way we choose $\lambda(s, p, v)$;
3. No equation from item (ii) can have the same coefficients with an equation from item (iii) — the worlds involved in equations of type (iii) refer to paths of length at least 2, whereas those of type (ii) are required to contain triples (s, p, v).

Therefore, $|R| \geq 9 \Rightarrow z \geq 9 \Rightarrow 4z^2 + z + 1 < 2^z$, then the system has an infinite number of solutions, thus there exist an infinite number of satisfying interpretations for (S, R).

Assumption. As most realistic pRDF theories will meet the condition set out in Theorem 1, throughout the rest of this paper, we assume we are only dealing with consistent pRDF theories.

4 pRDF: Fixpoint Semantics

In this section, we build upon Theorem 1 and show how to associate an operator with each pRDF theory that maps pRDF theories with pRDF theories. This operator has a least fixpoint that compactly represents all quadruples entailed by the theory.

Suppose (S, R) is a pRDF theory and n is a non-negative integer. Let $\mathcal{T}_n(s, p, v)$ be the set of p-paths between s and v of length n or less.

Definition 10 (Δ Operator). *Let (S, R) be a pRDF theory. Let:*

$$\epsilon(S, R) = \{(s, p', V, \delta) | \exists (s, p, V, \delta) \in R \wedge (p, rdfs : subProptertyOf, p') \in S\}.$$

$$\mu(S, R) = \{(s, p, \{v\}, \delta) | (|T_2(s, p, v)| \geq 2) \wedge \delta(v) = \max_{P_j \in T_2(s,p,v)} (\bigotimes_i \gamma_i^j)\}.$$

Let $\Delta(S, R) = (S, R \cup \epsilon(S, R) \cup \mu(S, R))$.

Similarly to relational databases, with a fixed pRDF schema, we show that Δ is monotonic w.r.t. the pRDF instance.

Proposition 2. *Δ is monotonic in its second argument, i.e. if $R_1 \subseteq R_2$, then $\Delta(S, R_1) \subseteq \Delta(S, R_2)$. Hence, for a given (S, R), Δ has a least fixpoint (S, R') with $R \subseteq R'$. This least fixpoint is denoted lfp(S,R) and is also called the* closure *of (S, R).*

Example 8. Consider the pRDF theory in Figures 1 and 2 and let $x \otimes y = xy$. The closure of this theory includes *(ACo, affiliatedWith, {FCo},{0.855})*; the probability was computed as the maximum on the two paths between ACo and FCo Another triple in the closure is *(ACo, affiliatedWith,{GCo},{.075})*.

Clearly, (S, R) entails $\Delta(S, R)$ according to Definitions 8 and 9, thus $(S, R) \models_{\otimes}$ lfp(S,R). Soundness of the inference system defined by the closure is thus guaranteed. The following proposition establishes the completeness of the closure, showing that all quadruples entailed by (S, R) are either in lfp(S,R) or trivially entailed by a single quadruple in it.

Theorem 2. *Let (S, R) be a pRDF theory and let $(S, R_{fp}) = lfp(S,R)$. Then for any (s, p, V, δ) s.t. $(S, R) \models_{\otimes} (s, p, V, \delta)$, $\exists\ (s, p, V', \delta') \in R_{fp}$ such that $V \subseteq V' \wedge \forall\ v \in V, \delta(v) \leq \delta'(v)$.*

Justification. Let us assume that $(S, R) \models_{\otimes} (s, p, V, \delta)$ and the proposition does not hold. Then there are two possible cases:

1. $\forall (s, p, V', \delta') \in R_{fp}, \exists\ v \in V - V'$. It is then straightforward that $\nexists (s, p, V'', \delta'') \in R$ s.t. $v \in V''$. Hence, from the justification of Proposition 1, the worlds containing quadruples with s, p, v are not in $\mathcal{W}_R$, thus there exists a satisfying interpretation such that $\sum_{(s,p,v) \in W} I(W) = 0$. Since $\delta(v) > 0$, I cannot satisfy (s, p, V, δ), leading to contradiction.
2. $\forall (s, p, V', \delta') \in R_{fp}$ such that $V \subseteq V'$, $\exists v \in V$ s.t. $\delta(v) > \delta'(v')$. We can easily see by induction, by the construction of $\mu(S, R)$ that the probability value assigned to each p-path is the same as the value used in the system of equations in the justification of Proposition 1. Since we have shown for any pRDF theory an infinite number of satisfying interpretations that meet (stricter) equality conditions than the inequalities required by Definition 8, we can easily construct a satisfying interpretation that does not satisfy (s, p, V, δ).

5 Atomic pRDF queries

An atomic pRDF query has the form (s, p, v, λ) where at most one of the members of the quadruple is allowed to be a variable. We prefix variables with a question mark. In this section, we show how to answer queries in the cases when the subject s, the property p and the probability λ are variables (the case when the value v is a variable is analogous to the case for s and hence is not repeated).

Theorem 2 provides a simple algorithm for answering atomic queries. The algorithm would: (1) compute lfp(S,R); (2) Compute A, the set of potential answers by performing a linear search for possible substitutions of q to quadruples in lfp(S,R); (3) Remove from A any redundant quadruples (w.r.t. entailment).

The simple method shown above is inefficient since: (i) it computes the entire lfp(S,R), which is space and time consuming; (ii) does not take advantage of the implicit probabilistic pruning required when λ is constant. We will now present efficient algorithms for answering atomic queries. The methods shown here can be easily extended to algorithms for answering simple queries with more than one variable[5].

Algorithm pRDF_Subject$(S, R, \otimes, q = (?s, p_q, v_q, \lambda_q))$
Input: pRDF theory (S, R), t-norm $\otimes$, query q.
Output: $Ans_q(S, R)$.
Notation: $SP(p) = \{p' \in \mathcal{P} | (p', rdfs : subPropertyOf^*, p)\}$.

1. $R' \leftarrow \{(s, p', V, \delta) \in R | p' \in SP(p_q)\}$;
2. **if** $p_q \in \mathcal{P}^t$ **then**
3. $\quad Q \leftarrow \{(s, \lambda_s) | \exists (s, p, V, \delta) \in R' \ s.t. \ (v_q \in V) \wedge (\lambda_s = \delta(v) \geq \lambda_q)\}$;
4. $\quad Q' \leftarrow Q$;
5. $\quad$ **while** $Q' \neq \emptyset$ **do**
6. $\quad\quad (s, \lambda_s) \leftarrow$ remove from Q' element with highest λ_s;
7. $\quad\quad Q'' \leftarrow \{(s', \lambda_{s'} \otimes \lambda_s) | \exists (s, p, V, \delta) \in R' \ s.t. \ (s \in V) \wedge$
$\quad\quad\quad (\lambda_{s'} = \delta(s) \geq min_\otimes(\lambda_s, \lambda_q))\}$;
8. $\quad\quad Q' \leftarrow Q' \cup Q''$;
9. $\quad\quad Q \leftarrow Q \cup Q''$;
10. $\quad\quad Q, Q' \leftarrow$ replace pairs $(s, \lambda_s), (s, \lambda'_s) \in Q, Q'$ with $(s, max(\lambda_s, \lambda'_s))$;
11. $\quad$ **end**
12. $\quad Ans \leftarrow \{(s, p_q, v_q, \lambda) | \exists (s, \lambda) \in Q\}$;
13. **else** $Ans \leftarrow \{(s, p_q, v_q, \lambda) | (\exists (s, p, V, \delta) \in R \ s.t. \ (v_q \in V) \wedge (\delta(v_q) = \lambda \geq \lambda_q)) \wedge$
$\quad\quad (\nexists (s, p_q, v_q, \lambda') \in Ans \ s.t. \lambda' > \lambda)\}$;
14. **end**
15. **return** Ans;

Fig. 3. Algorithm pRDF_*Subject* for answering queries $(?s, p, v, \lambda)$

Definition 11 (Answer). *Let (S, R) be a pRDF theory and let $q = (s, p, v, \lambda)$ be a simple query. The answer to q is $Ans_q(S, R) = \{(s', p', v', \lambda') | ((S, R) \models_\otimes (s', p', v', \lambda') \wedge (\exists \ \theta \ substitution \ s.t. \ (s\theta = s') \wedge (p\theta = p') \wedge (v\theta = v') \wedge (p\theta \leq p')) \wedge (\nexists (s', p', v', \lambda'') \in Ans_q(S, R) \ s.t. \ \lambda'' > \lambda'))\}$.*

Intuitively, the answer to (s, p, v, λ) consists of all instances of this query that are entailed by (S, R) subject to the restriction that if (S, R) entails (s', p', v', λ') and $(s'', p'', v'', \lambda'')$ and $\lambda'' < \lambda'$ then $(s'', p'', v'', \lambda'')$ is not included as it is clearly a redundant quadruple that adds nothing.

Example 9. Consider the pRDF theory in Figures 1 and 2. The following are simple queries and their answers:

1. Who is affiliated with FCo with probability above .91? $q = (?s, affiliatedWith, BCo, .91)$. The answer is {*(CCo, affiliatedWith, FCo, .95}*.

[5] However, we focus on atomic queries and conjunctions of atomic queries, as we expect this would be the predominant type of queries in a pRDF implementation.

2. Who is ACo affiliated with with probability above .8? $q = (ACo, affiliatedWith, ?v, .8)$. The answer is $\{(ACo,\ affiliatedWith,\ FCo,\ .855\}$.
3. What is the relation between ACo and FCo with probability above .7? $q = (ACo, ?p, FCo, .7)$. The answer is the same as the previous one.
4. What is the probability that ACo is affiliated with FCo? $q = (ACo, affiliatedWith, FCo, ?\lambda)$. The answer is the same as the previous one.

Algorithm pRDF_Property$(S, R, \otimes, q = (s_q, ?p, v_q, \lambda_q))$
Input: pRDF theory (S, R), t-norm $\otimes$, query q.
Output: $Ans_q(S, R)$.

1. $Q \leftarrow \{(v, p, \lambda_v) | \exists (s_q, p, V, \delta) \in R\ s.t.\ (v \in V) \wedge (\lambda_v = \delta(v) \geq \lambda_q) \wedge (p \in \mathcal{P}^t)\}$;
2. $Ans \leftarrow \emptyset$;
3. **while** $Q \neq \emptyset$ **do**
4. $\quad (v, p, \lambda_v) \leftarrow$ remove from Q element with highest λ_v;
5. $\quad Q' \leftarrow \{(v', p', \lambda_{v'} \otimes \lambda_v) | \exists (v, p', V, \delta) \in R\ s.t.\ (v' \in V) \wedge$
$\quad\quad ((p, rdfs : subPropertyOf, p') \in S) \wedge (\lambda_{v'} = \delta(v') \geq min_{\otimes}(\lambda_v, \lambda_q))\}$;
6. $\quad Q' \leftarrow Q' \cup \{(v', p, \lambda_{v'} \otimes \lambda_v) | \exists (v, p', V, \delta) \in R\ s.t.\ (v' \in V) \wedge$
$\quad\quad ((p', rdfs : subPropertyOf, p) \in S) \wedge (\lambda_{v'} = \delta(v') \geq min_{\otimes}(\lambda_v, \lambda_q))\}$;
7. $\quad Ans \leftarrow Ans \cup \{(s_q, p, v_q, \lambda) | \exists (v_q, p, \lambda) \in Q'\}$;
8. $\quad Q' \leftarrow$ remove elements containing v_q from Q';
9. $\quad Q \leftarrow Q \cup Q'$;
10. $\quad Q \leftarrow$ replace pairs $(v, p, \lambda_v), (v, p, \lambda'_v) \in Q$ with $(v, p, max(\lambda_v, \lambda'_v))$;
11. **end**
12. $Ans \leftarrow Ans \cup \{(s_q, p, v_q, \lambda) | \exists\ (s, p, V, \delta) \in R\ s.t.\ (p \in \mathcal{P}^{nt}) \wedge (v_q \in V) \wedge$
$\quad\quad (\delta(v) = \lambda \geq \lambda_q)\}$;
13. $Ans \leftarrow$ remove redundant quadruples from Ans;
14. **return** Ans;

Fig. 4. Algorithm pRDF_*Property* for answering queries $(s, ?p, v, \lambda)$

Algorithm pRDF_Probability$(S, R, \otimes, q = (s_q, p_q, v_q, ?\lambda))$
Input: pRDF theory (S, R), t-norm $\otimes$, query q.
Output: $Ans_q(S, R)$.
Notation: $SP(p) = \{p' \in \mathcal{P} | (p', rdfs : subPropertyOf^*, p)\}$.

1. $R' \leftarrow \{(s, p', V, \delta) \in R | p' \in SP(p_q)\}$;
2. **if** $p_q \in \mathcal{P}^t$ **then**
3. $\quad Q \leftarrow \{(v, \delta(v)) | \exists (s_q, p, V, \delta) \in R'\ s.t.\ (v \in V)\}$;
4. $\quad Ans \leftarrow \emptyset$;
5. $\quad$ **while** $Q \neq \emptyset$ **do**
6. $\quad\quad (v, \lambda_v) \leftarrow$ remove from Q element with highest λ_v;
7. $\quad\quad Q' \leftarrow \{(v', \delta(v') \otimes \lambda_v) | \exists (v, p', V, \delta) \in R'\ s.t.\ (v' \in V)\}$;
8. $\quad\quad Ans \leftarrow Ans \cup \{\lambda | \exists (v_q, \lambda) \in Q'\}$;
9. $\quad\quad Q' \leftarrow$ remove elements containing v_q from Q';
10. $\quad\quad Q \leftarrow Q \cup Q'$;
11. $\quad\quad Q \leftarrow$ replace pairs $(v, \lambda_v), (v, \lambda'_v) \in Q$ with $(v, max(\lambda_v, \lambda'_v))$;
12. $\quad$ **end**
13. **else** $Ans \leftarrow \{\delta(v) | \exists (s_q, p_q, V, \delta) \in R\ s.t.\ v_q \in V\}$;
14. **return** $\{(s_q, p_q, v_q, \max_{\lambda \in Ans}\ )\}$;

Fig. 5. Algorithm pRDF_*Probability* for answering queries $(s, p, v, ?\lambda)$

Algorithm *pRDF_Subject* shown in Figure 3 computes the answer to atomic queries with an unknown subject. The correctness of the algorithm follows directly from: (i) for non-transitive properties, the search performed on line 13 has the same effect as searching lfp(S,R); (ii) for transitive properties, we perform inference steps on lines 5—11 starting backwards from the path destination

v_q, corresponding to $\mu(S, R)$ in Definition 10. The algorithm prunes the search space at every inference step by excluding quadruples that are below the query probability λ_q on line 7; due to the properties of $min_{\otimes}$, the pruned quadruples clearly cannot be in the answer. Furthermore, the probability assigned to a triple (s, p, v) (such that there exist several p-paths between s and v) in Definition 10 is accounted for by line 10. Note that the main difference between Q and Q' is that elements from Q' are removed as the iteration advances through the pRDF theory; instead, Q gathers all values that could be substituted for s'. *pRDF_Subject* also performs an initial pruning step on line 1, by limiting the search to the portion of the pRDF theory that is related to p_q. For reasons of space, we omit the algorithm for answering atomic queries with unknown value due to its similarity to *pRDF_Subject*.

The complexity of *pRDF_Subject* depends on both $|R|$ and the *width* of the pRDF theory. The width of a pRDF theory (S, R) is a value $c_R = \max_{(s,p,V,\delta)\in R} |V|$. The algorithm traverses each quadruple $(s, p, V, \delta) \in R$ at most once[6] and iterates through V for each such quadruple. Thus, assuming $min_{\otimes}$ is computable in constant time, the maximum number of steps is $\mathcal{O}(|R| \cdot c_R)$, which coincides with the space bound for $Ans_q(S, R)$.

Algorithm *pRDF_Property* given in Figure 4 computes the answer to atomic queries with unknown property. The algorithm differs from *pRDF_Subject* in that it requires us to keep track of the properties when iterating through each quadruple, hence the structure of the sets Q, Q'. The algorithm starts with p-paths originating at s_q (line 1) and moves forward through the pRDF theory until v_q is reached (lines 3—11); once v_q is reached (line 7), a new quadruple is added to the answer. When all paths have been followed to their destination (or pruned by $min_{\otimes}$), the quadruples obtained from non-transitive properties are added to the answer (line 12). Finally, even though we guarantee through line 10 that Q does not contain elements for intermediary p-paths leading up to the same element, this does not necessarily hold for the destination v_q. Therefore, we eliminate quadruples trivially entailed by the answer from Ans on line 13 and return. The same complexity analysis holds as in the case of *pRDF_Subject*.

Algorithm *pRDF_Probability* shown in Figure 5 computes the answer to atomic queries with unknown probability. The answer to such a query is always a single quadruple, according to Definition 11. The algorithm computes the $\bigotimes$ value for each p_q-path from s_q to v_q and then the maximum is taken and returned.

Theorem 3. *Algorithms pRDF_Subj, pRDF_Property and pRDF_Probability are all correct, i.e. they correctly return $Ans_q(S, R)$ for each query q to a pRDF theory (S, R).*

[6] We remind the reader that the pRDF theory is free of cycles on transitive properties.

6 pRDF conjunctive queries

In this section, we give an algorithm for answering conjunctions of simple queries for which the probability is known. The method is based on (i) computing lfp(S,R) taking into account the proper probabilistic pruning (since the probabilities for all queries are known); (ii) both the pRDF theory and the query can be represented as graphs and hence inexact graph matching algorithms [4] can be used to determine candidate answers. The method described in this section can be used for general conjunctive queries, assuming a lower bound on the probability threshold of the individual queries is known; this is generally a reasonable assumption, since in a probabilistic setting a user would specify a minimum level of confidence required from the answer.

Formally, we write a conjunctive query $Q = q_1 \wedge \ldots \wedge q_n$ We define the answer to a conjunctive query $Ans_Q(S, R) = \{Q\theta \mid Q\theta$ is ground (i.e. variable free) and for all $1 \leq i \leq n$, $q_i\theta \in Ans_{q_i}(S, R)\}$.

Algorithm pRDF_Conjunctive($S, R, \otimes, Q = \{q_i = (s_i, p_i, v_i, \lambda_i) | i \in [1, n]\}$))
Input: pRDF theory (S, R), t-norm $\otimes$, set of simple queries Q.
Output: $Ans_q(S, R)$.
Notation: $SP(p) = \{p' \in \mathcal{P} | (p', rdfs : subPropertyOf^*, p)\}$.

1. $\lambda_t \leftarrow \min_{i \in [1,n]} (\lambda_i)$;
2. **if** all p_i are known **then** $R \leftarrow \{(s, p', V, \delta) \in R | p' \in \bigcup_i SP(p_i)\}$;
3. **do**
4. $\quad R \leftarrow R'$:
5. $\quad R' \leftarrow R \cup \epsilon(S, R)$;
6. $\quad R' \leftarrow R' \cup \{(s, p, \{v\}, \delta) | (|T_2(s, p, v) \geq 2) \wedge \delta(v) = \max_{P_j \in T_2(s,p,v)} (\bigotimes_i \gamma_i^j) \wedge \delta(v) \geq \lambda_t\}$;
7. **while** $R' = R$;
8. **for** all matches between Q and a subgraph R_A of (S, R) **do**
9. $\quad ok \leftarrow true$;
10. $\quad$ **for** $i \in [1, n]$ **do**
11. $\quad\quad$ **let** (s, p, v, λ) be the match for q_i;
12. $\quad\quad$ **if** $\lambda < \lambda_i$ **then**
13. $\quad\quad\quad ok \leftarrow false$;
14. $\quad\quad\quad$ **break**;
15. $\quad\quad$ **end**
16. $\quad$ **end**
17. $\quad$ **if** ok **then** $Ans \leftarrow Ans \cup \{R_A\}$;
18. **end**
19. **return** Ans;

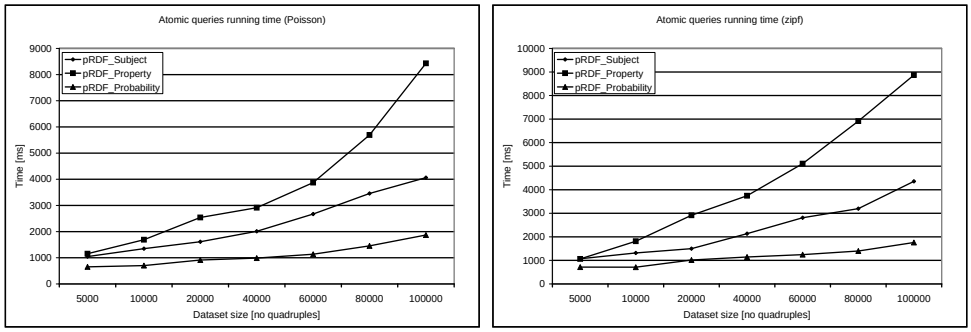
Fig. 6. Algorithm pRDF_*Conjunctive* for answering conjunctions of simple queries

A sketch of the pRDF_*Conjunctive* algorithm is shown in Figure 6. The algorithm starts by computing part of lfp(S,R) by pruning on the property labels if all queries have known properties (line 2) and furthermore on the lower bound of the probability thresholds of the simple queries (line 6). After the portion of lfp(S,R) relevant to the conjunctive query has been computed, any inexact graph matching technique can be used to generate candidate answers, which are then verified on lines 8—18. Further pruning is also possible when computing lfp(S,R) if we look at the set of possible queries $\{q_j\}$ that can be matched by a particular quadruple $(s, p, \{v\}, \delta)$ on line 6 and prune the set of such quadruples at $min_j(p_j) \leq \lambda_t$. Whether such a step is efficient clearly depends on the width

of the graph compared to the number of queries in the conjunction. The complexity of computing the closure is at the worst case exponential in $|R|$, but our experimental results in Section 7 show that the algorithms scales well on large datasets.

7 Experimental results

We have developed an experimental prototype of the pRDF system consisting of about 1700 lines of Java code. Experiments were conducted on a Pentium 4 3.2 GHz machine with 512 MB or RAM running SuSE Linux 9.1. The experiments were performed on synthetically generated datasets. The parameters and method of random generation will be described for each set of experiments.



(a) Poisson distribution (b) Zipf distribution

Fig. 7. Atomic query running time

In the first set of experiments, $|\mathcal{P}| = 50$ and $|\mathcal{P}^t| = 15$. The average width of the pRDF theory was 15. We varied $|R|$ between 5,000 and 100,000 quadruples and measured the running time, including any disk I/O overhead. The experiment was performed by using the Poisson and Zipf distribution respectively when generating quadruples. The results averaged over 30 runs are shown in Figure 7(a) and (b). Ss expected, the $pRDF\_Probability$ is the fastest algorithm; this is due to (i) knowing s_q, p_q, v_q and (ii) the reduction of the space needed to store intermediate values of Ans. $pRDF\_Subject$ has a linear trend, whereas $pRDF\_Property$ is the least efficient due to the large search space. In a separate experiment we have determined that by using he naive method (computing lfp(S,R) and performing a linear search), the running time increases by a factor ranging between 6 and 15. Clearly, the probability distribution used for the quadruples has very little influence on the running time of the algorithms.

In the second set of experiments, we measured the efficiency of $pRDF\_Conjunctive$. The conditions were the same as in the first set of experiments. We averaged the running time over 30 runs for 5, 10, 20 and 30 atomic queries in the conjunction. The results are shown in Figure 8.

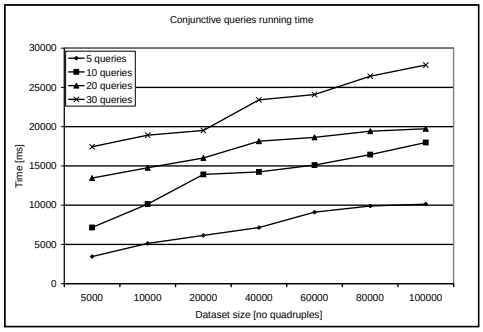
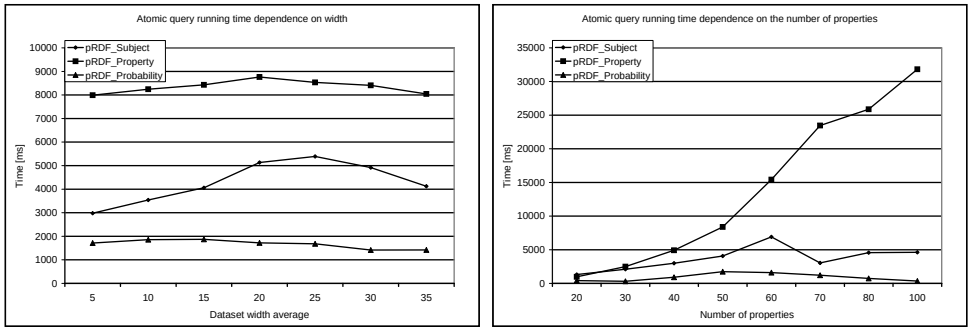


Fig. 8. Conjunctive query running time



(a) Dependence on pRDF theory width (b) Dependence on number of properties

Fig. 9. Atomic query running time

In a third set of experiments, with $|R|$=100,000, $|\mathcal{P}| = 50$ and $|\mathcal{P}^t| = 15$, we varied the average width of the pRDF theory from 5 to 35, with a standard deviation of 1.5. The results averaged over 30 runs are shown in Figure 9(a). At the beginning, the number of iterations in the algorithms increases linearly with the width. As the width reaches 20, the probabilities in the quadruples decrease enough so more and more paths are pruned. Figure 9(b) shows the dependence of the running time on the number of properties, with an average width of 15; we maintained a 20% ratio of transitive properties. We can see that $pRDF\_Property$ has a high dependence on the number of properties, since more properties mean more answer candidates. For the other two algorithms, two phenomena affect the running time: on the one hand, increasing the number of properties means more paths to consider during the search; one the other hand, since $|R|$ is fixed, it also means paths are shorter, hence the depth of the search is reduced.

8 Related work and conclusion

Giugno and Lukasiewicz [5] were perhaps the first to consider incorporating uncertainty into semantic web models. They developed a framework to account for probabilities in SHOQ(D). Udrea et. al. [6] developed a framework called PARQ to represent probabilistic ISA-hierarchies and improve the recall of queries to relational databases using these probabilistic ISA-hierarchies. [7, 8] in a workshop held in France earlier this year developed some ideas for probabilistic ontologies.

Our approach also relates to the vast amount of work in Bayesian networks [9]. However, Bayesian networks are a different type of labeled graphs than pRDF theories; secondly, our query answering algorithms also differ from Bayesian approaches.

In this paper, we have developed a *Probabilistic RDF* framework within which users can express probabilistic information about the relationships expressed in RDF. We have developed results on the consistency of pRDF theories, together with a fixpoint semantics and algorithms to answer queries to pRDF theories. Our prototype implementation of pRDF is, to our knowledge, the first of its kind.

References

1. Udrea, O., Deng, Y., Ruckhaus, E., Subrahmanian, V.S.: A graph theoretical foundation for integrating rdf ontologies. In: Proccedings of the Twentieth National Conference on Artificial Intelligence. (2005) 1442–1447
2. Fagin, R.: Combining fuzzy information from multiple systems. J. Computer and Systems Sciences (**58**) 83–99
3. Halpern, J.: The relationship between knowledge, belief, and certainty. In: Proceedings of the 5th Annual Conference on Uncertainty in Artificial Intelligence (UAI-90), New York, NY, Elsevier Science Publishing Company, Inc. (1990) 142–151
4. Hlaoui, A., Wang, S.: A new algorithm for inexact graph matching. In: ICPR (4). (2002) 180–183
5. Giugno, R., Lukasiewicz, T.: P-shoq(d): A probabilistic extension of shoq(d) for probabilistic ontologies in the semantic web. In: JELIA '02: Proceedings of the European Conference on Logics in Artificial Intelligence, London, UK, Springer-Verlag (2002) 86–97
6. Udrea, O., Deng, Y., Hung, E., Subrahmanian, V.: Probabilistic ontologies and relational databases. Lecture Notes in Computer Science **3760** (2005) 1–17
7. Dubois, D., Mengin, J., Prade., H.: Possibilistic uncertainty and fuzzy features in description logic: a preliminary discussion. In: Proc. Workshop on Fuzzy Logic and the Semantic Web (ed. E. Sanchez). (2005) 5–7
8. Straccia, U.: Towards a fuzzy description logic for the semantic web. In: Proc. Workshop on Fuzzy Logic and the Semantic Web (ed. E. Sanchez). (2005) 3–3
9. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)