Logical Foundations of (e)RDF(S): Complexity and Reasoning*

Jos de Bruijn[1] and Stijn Heymans[2]

[1] Faculty of Computer Science, Free University of Bozen-Bolzano, Italy
debruijn@inf.unibz.it
[2] Digital Enterprise Research Institute (DERI), University of Innsbruck, Austria
stijn.heymans@deri.org

Abstract. An important open question in the semantic Web is the precise relationship between the RDF(S) semantics and the semantics of standard knowledge representation formalisms such as logic programming and description logics. In this paper we address this issue by considering embeddings of RDF and RDFS in logic. Using these embeddings, combined with existing results about various fragments of logic, we establish several novel complexity results. The embeddings we consider show how techniques from deductive databases and description logics can be used for reasoning with RDF(S). Finally, we consider querying RDF graphs and establish the data complexity of conjunctive querying for the various RDF entailment regimes.

1 Introduction

The Resource Description Framework RDF [16], together with its vocabulary description language RDFS [3], constitutes the basic language for the semantic Web. More expressive semantic Web languages such as the description logic-based OWL DL [19] and future semantic Web rule languages[3] are supposed to extend it. However, certain properties of the RDF semantics posed layering problems in the definition of OWL DL [12]; it was decided to extend only a part of RDF. This has led to a situation in which the user communities of RDF and OWL DL are increasingly diverging, leading to a fragmentation of the semantic Web. There is a possibility that a similar situation will occur if a possible future rules language for the semantic Web does not adequately account for RDF(S).

Research has been done to uncover some of the formal properties of RDF (e.g. [13, 10, 4]). However, so far little research has been done into the formal relationships between RDF and the logical language paradigms of description logics and logic programming.[4] Therefore, we deem it worthwhile to investigate these relationships, in order to facilitate the RDF-compatibility of a future logic-based rules language for the semantic Web, and to see whether the RDF and description logic worlds can be brought closer

* This work was partially supported by the European Commission under the projects Knowledge Web (IST-2004-507482) and SUPER (FP6-026850).

[3] http://www.w3.org/2005/rules/

[4] A notable exception is [4].

together. Additional benefits of such an investigation include the possible use of techniques from logic programming and description logics for reasoning with RDF(S).

The RDF semantics specification [11] defines three increasingly expressive kinds of entailment, namely *simple*, *RDF* and *RDFS entailment*. Furthermore, it describes *extensional RDFS* (eRDFS) entailment as a possible extension of RDFS entailment which is more in line with description logic-like languages. We refer to these kinds of entailment as *entailment regimes*.

To investigate the relationship between RDF and logic we embed the various RDF entailment regimes in F-Logic [15], which is a syntactic extension of first-order logic with object oriented modeling constructs. It turns out that the attribute value construct in F-Logic is exactly equivalent to the triple construct in RDF, and the typing (class membership) construct in F-Logic is very close in spirit to the one in RDF. Additionally, F-Logic, like RDFS, allows to use the same identifier as a class, instance, or property identifier, while still having a standard first-order logic-based semantics.

These embeddings can be used for RDF reasoning using Datalog engines. Furthermore, they lead to several novel complexity results about RDF; see Table 3 on page 11 for an overview of the complexity results for the various entailment regimes.

We then show the embedding of a large subset of *extensional* RDFS in FOL, and we show that it can be embedded in the tractable description logic (contextual) $DL\text{-}Lite_{\mathcal{R}}$ [7].

Finally, we define a notion of conjunctive queries over RDF graphs, and establish the data complexity of query answering for the respective entailment regimes.

The structure of the remainder of the paper is as follows. In Section 2 we review F-Logic, $DL\text{-}Lite_{\mathcal{R}}$, and RDF. In Section 3 we define embeddings of RDF in F-Logic and FOL, and demonstrate the relationship with $DL\text{-}Lite_{\mathcal{R}}$. In Section 4 we use these embeddings and correspondences to obtain several novel complexity results for RDF. In Section 5 we define conjunctive query answering in RDF and exhibit its complexity. We discuss some implications of the results in this paper, and compare it with related work, in Section 6. We conclude the paper and outline future work in Section 7.

This paper is an extended version of [5], adding support for literals and treatment of RDF querying. Unlike in [5] we do not consider extensions of the embeddings of logical rules or formulas in this paper; see Section 7 for a discussion about the combination of RDF graphs with logical rules.
Proofs of all results are available in an extended version of this paper.

2 Preliminaries

F-Logic[5] extends first-order logic with constructs for object-oriented modeling (we use the object typing and attribute value construct), while staying in a first-order semantic framework.

The signature of an F-language $\mathcal{L}$ is of the form $\Sigma = \langle \mathcal{F}, \mathcal{P} \rangle$ with $\mathcal{F}$ and $\mathcal{P}$ disjoint sets of function and predicate symbols, each with an associated arity $n \geq 0$. Let $\mathcal{V}$ be a

[5] Note that F-Logic is also often used as an extension of nonmonotonic logic programming; however, we follow the original definition which is strictly first-order.

set of variable symbols disjoint from $\mathcal{F}$ and $\mathcal{P}$. Terms and atomic formulas are defined in the usual way; $\bot$ is an atomic formula. A molecule in F-Logic is one of the following: (i) an *is-a* molecule of the form $C : D$, which states that an individual C is of the type D, or (ii) a *data molecule* of the form $C[D \twoheadrightarrow E]$ which states that an individual C has an attribute D with the value E, where C, D and E terms,. A molecule is *ground* if it does not contain variable symbols.

Formulas of an F-language $\mathcal{L}$ are either atomic formulas, molecules, or compound formulas, which are constructed in the usual way from atomic formulas, molecules, the logical connectives $\neg$, $\wedge$, $\vee$ and $\supset$, the quantifiers $\exists$ and $\forall$, and the auxiliary symbols ')' and '('. An F-Logic theory $\Phi \subseteq \mathcal{L}$ is a set of formulas.

F-Logic Horn formulas are of the form $(\forall) B_1 \wedge \ldots \wedge B_n \supset H$, with $B_1, \ldots, B_n, H$ atomic formulas or molecules. F-Logic Datalog formulas are F-Logic Horn formulas with no function symbols of arity higher than 0 such that every variable in H occurs in some equality-free $B_1, \ldots, B_n$. F-Logic Horn and Datalog theories are sets of F-Logic Horn and Datalog formulas, respectively.

An *F-structure* is a tuple $\mathbf{I} = \langle U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\twoheadrightarrow} \rangle$, where U is a non-empty, countable set (the domain) and $\in_U$ is a binary relation over U. An n-ary function symbol $f \in \mathcal{F}$ is interpreted as a function over the domain U: $\mathbf{I}_F(f) : U^n \rightarrow U$. An n-ary predicate symbol $p \in \mathcal{P}$ is interpreted as a relation over the domain U: $\mathbf{I}_P(p) \subseteq U^n$. $\mathbf{I}_{\twoheadrightarrow}$ associates a binary relation over U with each $k \in U$: $\mathbf{I}_{\twoheadrightarrow}(k) \subseteq U \times U$. Variable assignments are defined as usual.

Given an F-structure $\mathbf{I}$ of an F-language $\mathcal{L}$, a variable assignment B, and a term t of $\mathcal{L}$, $t^{\mathbf{I},B}$ is defined as: $x^{\mathbf{I},B} = x^B$ for $x \in \mathcal{V}$ and $t^{\mathbf{I},B} = \mathbf{I}_F(f)(t_1^{\mathbf{I},B}, \ldots, t_n^{\mathbf{I},B})$ for t of the form $f(t_1, \ldots, t_n)$, with $f \in \mathcal{F}$ an n-ary function symbol and $t_1, \ldots, t_n$ terms.

Satisfaction of atomic formulas and molecules ϕ in $\mathbf{I}$, given the variable assignment B, denoted $(\mathbf{I}, B) \models_{\mathsf{f}} \phi$, is defined as: $(\mathbf{I}, B) \not\models_{\mathsf{f}} \bot$, $(\mathbf{I}, B) \models_{\mathsf{f}} p(t_1, \ldots, t_n)$ iff $(t_1^{\mathbf{I},B}, \ldots, t_n^{\mathbf{I},B}) \in \mathbf{I}_P(p)$, $(\mathbf{I}, B) \models_{\mathsf{f}} t_1 : t_2$ iff $t_1^{\mathbf{I},B} \in_U t_2^{\mathbf{I},B}$, $(\mathbf{I}, B) \models_{\mathsf{f}} t_1[t_2 \twoheadrightarrow t_3]$ iff $\langle t_1^{\mathbf{I},B}, t_3^{\mathbf{I},B} \rangle \in \mathbf{I}_{\twoheadrightarrow}(t_2^{\mathbf{I},B})$, and $(\mathbf{I}, B) \models_{\mathsf{f}} t_1 = t_2$ iff $t_1^{\mathbf{I},B} = t_2^{\mathbf{I},B}$. This extends to arbitrary formulas in the usual way.

The notion of a model is defined in the usual way. A theory $\Phi \subseteq \mathcal{L}$ *F-entails* a formula $\phi \in \mathcal{L}$, denoted $\Phi \models_{\mathsf{f}} \phi$, if for all F-structures $\mathbf{I}$ such that $\mathbf{I} \models_{\mathsf{f}} \Phi$, $\mathbf{I} \models_{\mathsf{f}} \phi$.

FOL and $DL\text{-}Lite_{\mathcal{R}}$ *Classical first-order logic* (FOL) is F-Logic without molecules. *Contextual first-order logic* (contextual FOL) is classical FOL where $\mathcal{F}$ and $\mathcal{P}$ are not required to be disjoint, function and predicate symbols do not have associated arities, and for every structure $\mathbf{I} = \langle U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\twoheadrightarrow} \rangle$ it is the case that $\mathbf{I}_F$ assigns a function $\mathbf{I}_F(f) : U^i \rightarrow U$ to every $f \in \mathcal{F}$ and $\mathbf{I}_P$ assigns a relation $\mathbf{I}_P(p) \subseteq U^i$ to every $p \in \mathcal{P}$, for every nonnegative integer i. We denote satisfaction and entailment in classical and contextual FOL using the symbols $\models$ and $\models_c$, respectively.

F-Logic can be straightforwardly embedded in FOL, as shown in the following proposition.

Proposition 1. *Let Φ and ϕ be an F-Logic theory and formula which do not contain the binary and ternary predicate symbols isa and $data$, respectively, and let Φ' and ϕ' be the FOL theory and formula obtained from Φ and ϕ by replacing every is-a molecule $a : b$ with the atomic formula $isa(a, b)$ and every data molecule $a[b \twoheadrightarrow c]$ with the atomic*

formula $data(a, b, c)$. *Then,*
– Φ *has a model iff* Φ' *has a model and*
– $\Phi \models_{\mathsf{f}} \phi$ *iff* $\Phi' \models \phi'$.

$DL\text{-}Lite_{\mathcal{R}}$ [7] is a description logic (DL) with certain desirable computational properties; most reasoning tasks are polynomial, and query answering has LogSpace data complexity.

A classical (resp., contextual) $DL\text{-}Lite_{\mathcal{R}}$ language consists of pairwise disjoint (resp., possibly non-disjoint) sets of concept ($\mathcal{C}$), role ($\mathcal{R}$), and individual ($\mathcal{F}$) identifiers. Concepts and roles in $DL\text{-}Lite_{\mathcal{R}}$ are defined as follows:

$$\begin{aligned} C_l &\longrightarrow A \mid \exists R \\ C_r &\longrightarrow A \mid \exists R \mid \neg A \mid \neg \exists R \\ R, R' &\longrightarrow P \mid P^- \end{aligned}$$

with $A \in \mathcal{C}$, $P \in \mathcal{R}$, C_l (resp., C_r) a left- (resp., right-)hand side concept, and $R, R' \in \mathcal{R}$ roles.

A $DL\text{-}Lite_{\mathcal{R}}$ knowledge base $\mathcal{K}$ consists of a TBox $\mathcal{T}$ which is a set of inclusion axioms of the forms $C_l \sqsubseteq C_r$ and $R \sqsubseteq R'$, and an ABox $\mathcal{A}$ which is a set of concept and role membership assertions of the forms $A(a)$ and $P(a_1, a_2)$, with $a, a_1, a_2 \in \mathcal{F}$. We define the semantics of $DL\text{-}Lite_{\mathcal{R}}$ through a translation to classical (resp., contextual) FOL, using the mapping function π, which is defined in Table 1; π extends naturally to sets of axioms and assertions.

$\pi(A, X) = A(X)$ $\pi(P, X, Y) = P(X, Y)$ $\pi(P^-, X, Y) = P(Y, X)$	$\pi(C_l \sqsubseteq C_r) = \forall x(\pi(C_l, x) \supset \pi(C_r, x))$ $\pi(R_1 \sqsubseteq R_2) = \forall x, y(\pi(R_1, x, y) \supset \pi(R_2, x, y))$
$\pi(\exists R, X) = \exists y(R(X, y))$ $\pi(\neg A, X) = \neg\pi(A, X)$ $\pi(\neg\exists R, X) = \neg\exists y(R(X, y))$	$\pi(A(a)) = A(a)$ $\pi(P(a_1, a_2)) = P(a_1, a_2)$
y is a new a variable	

Table 1. Mapping $DL\text{-}Lite_{\mathcal{R}}$ to FOL

Given a classical (resp., contextual) $DL\text{-}Lite_{\mathcal{R}}$ knowledge base $\mathcal{K}$, the classical (resp., contextual) *FOL equivalent* of $\mathcal{K}$ is the theory $\Phi = \pi(\mathcal{K}) = \pi(\mathcal{T}) \cup \pi(\mathcal{A})$.

RDF is a data language, where the central notion is the *graph*, which is a set of *triples* of the form $\langle s, p, o \rangle$; s is the *subject*, p is the *predicate*, and o is the *object* of the triple.

A *vocabulary* $V = \langle \mathcal{F}, \mathcal{PL}, \mathcal{TL}, \mathcal{B} \rangle$ consists of a set $\mathcal{F}$ of URI references, a set $\mathcal{PL}$ of plain literals (i.e. Unicode character strings with an optional language tag), a set $\mathcal{TL}$ of typed literals (i.e. pairs (s, u) of a Unicode string s and a URI u denoting a datatype), and a set $\mathcal{B}$ of blank nodes (i.e. existentially quantified variables); see [16, Sections 6.4, 6.5, 6.6] for more details about the specific form of these symbols. Terms

are URI references, plain or typed literals, or blank nodes. A *graph* S of a vocabulary V is a set of *triples* $\langle s, p, o\rangle$, with $s, p, o \in \mathcal{F} \cup \mathcal{PL} \cup \mathcal{TL} \cup \mathcal{B}$.[6] With $bl(\langle s, p, o\rangle)$ (resp., $bl(S)$) we denote the set of blank nodes in $\langle s, p, o\rangle$ (resp., S). A triple $\langle s, p, o\rangle$ (resp., graph S) is *ground* if $bl(\langle s, p, o\rangle) = \emptyset$ (resp., $bl(S) = \emptyset$).

An interpretation of a vocabulary V is a tuple $I = \langle U, U_P, U_L, I_{\mathcal{F}}, I_L, I_{ext}\rangle$, where U is a countable non-empty set, called the domain, U_P is a countable set of properties, $U_L \subseteq U$ is a countable set of literal values with $\mathcal{PL} \subseteq U_L$, $I_{\mathcal{F}}$ is an interpretation function for URI references $I_{\mathcal{F}} : \mathcal{F} \rightarrow U \cup U_P$, I_L is an interpretation function for typed literals $I_L : \mathcal{TL} \rightarrow U$, and I_{ext} is an *extension function* $I_{ext} : U_P \rightarrow 2^{(U \times U)}$.

Given an interpretation I of a vocabulary V and a subset of the blank nodes $\mathcal{B}' \subseteq \mathcal{B}$, we define a mapping $A : \mathcal{B}' \rightarrow U$ which is used to interpret blank nodes. For a term t we define $t^{I,A}$ as: (a) if $t \in \mathcal{F}$, then $t^{I,A} = I_{\mathcal{F}}(t)$, (b) if $t \in \mathcal{PL}$, then $t^{I,A} = t$, (c) if $t \in \mathcal{TL}$, then $t^{I,A} = I_L(t)$, and (d) if $t \in \mathcal{B}'$, then $t^{I,A} = A(t)$.

An interpretation I *satisfies* a triple $\langle s, p, o\rangle$ with respect to a mapping $A : \mathcal{B}' \rightarrow U$, with $bl(\langle s, p, o\rangle) \subseteq \mathcal{B}'$, denoted $(I, A) \models \langle s, p, o\rangle$, if $p^{I,A} \in U_P$ and $\langle s^{I,A}, o^{I,A}\rangle \in I_{ext}(p^{I,A})$. I satisfies a graph S with respect to a mapping $A : bl(S) \rightarrow U$, denoted $(I, A) \models S$, if $(I, A) \models \langle s, p, o\rangle$ for every $\langle s, p, o\rangle \in S$. An interpretation I is a *model* of a graph S, denoted $I \models S$, if $(I, A) \models S$ for some $A : bl(S) \rightarrow U$.

Any interpretation is an *s-interpretation* (simple interpretation). An interpretation I is an *rdf-* (resp., *rdfs-*, *erdfs-*)interpretation if it interprets the RDF (resp., RDFS) vocabulary, satisfies the RDF (resp., RDFS) axiomatic triples, and satisfies a number of conditions, as specified in [11].

An RDF graph S s-(resp., rdf-, rdfs-, or erdfs-)entails an RDF graph E if every s-(resp., rdf-, rdfs-, or erdfs-)interpretation which is a model of S is also a model of E. We refer to these kinds of entailment as *entailment regimes*, and use the symbol $\models_x$, with $x \in \{s, rdf, rdfs, erdfs\}$, to denote entailment under the respective regimes.

Intuitively, the difference between the RDFS and eRDFS entailment regimes is that for the latter, whenever an ontological relation (e.g. subclass or property domain) implicitly holds in an interpretation, the corresponding RDF statement (`subClassOf`, `domain`) must be true, whereas this is not always the case with the RDFS entailment regime. The following example illustrates this difference.

Example 1. Let S be the following graph: $S = \{\langle parent, \texttt{domain}, Person\rangle, \langle mother, \texttt{subPropertyOf}\ parent\rangle$ which says that the domain of $parent$ is $Person$, and the property $mother$ is a sub-property of $parent$. Using eRDFS entailment we can conclude from S that the domain of $mother$ is also $Person$:

$$S \models_{erdfs} \langle mother, \texttt{domain}, Person\rangle;$$

it is always the case that the subject of any $mother$ triple has the type $Person$; thus, $mother$ implicitly has the domain $Person$. We cannot draw this conclusion when using RDFS entailment; in RDFS, only explicitly asserted domain constraints can be derived.

[6] Note that we allow literals in subject (s), and literals and blank nodes in predicate (p) positions, whereas the RDF specification [16] does not. Nonetheless, our results immediately apply to standard RDF graphs as defined in [16].

3 RDF(S) Embedding

In this section we first define an embedding of the various entailment regimes in F-Logic. We then consider an embedding of eRDFS entailment in FOL and DL.

3.1 Embedding RDF in F-Logic

We embed a graph as a conjunction of data molecules; URI references and literals are treated as constant symbols, and blank nodes are treated as existentially quantified variables. In the remainder we assume that RDF graphs are finite.

Given a vocabulary $V = \langle \mathcal{F}, \mathcal{PL}, \mathcal{TL}, \mathcal{B} \rangle$, an F-language $\mathcal{L}$ *corresponds to* V if it has a signature of the form $\Sigma = \langle \mathcal{F}' \supseteq \mathcal{F} \cup \mathcal{PL} \cup \mathcal{TL}, \mathcal{P} \rangle$.[7]

Definition 1. *Let S be an RDF graph of a vocabulary V, let $\langle s, p, o \rangle \in S$ be a triple in S, and let $\mathcal{L}$ be an F-language which corresponds to V. Then,*

$$\begin{aligned} (tr(\langle s, p, o \rangle) \in \mathcal{L}) &= s[p \twoheadrightarrow o] \text{ and} \\ (tr(S) \in \mathcal{L}) &= \{\exists\, bl(S)(\bigwedge\{tr(\langle s, p, o \rangle) \mid \langle s, p, o \rangle \in S\})\}. \end{aligned}$$

If ϕ is an F-Logic formula or theory in prenex normal form with only existential quantifiers, then ϕ^{sk} denotes the *Skolemization* of ϕ, i.e. every existentially quantified variable is replaced with a globally unique new constant symbol.

We use a set of formulas $\Psi^x \subseteq \mathcal{L}$, as defined in Table 2, to axiomatize the semantics of an entailment regime $x \in \{s, rdf, rdfs, erdfs\}$.

Proposition 2. *Let S be an RDF graph of a vocabulary V. Then, $tr(S)^{sk} \cup \Psi^x$, with $x \in \{s, rdf, rdfs\}$, can be equivalently rewritten to a set of F-Logic Datalog formulas.*

Note that Ψ^{erdfs} cannot be equivalently rewritten to a set of Datalog formulas, because of the use of universal quantification in the antecedents of some of the implications in Ψ^{erdfs}.

Theorem 1. *Let S and E be RDF graphs of a vocabulary V and let $x \in \{s, rdf, rdfs, erdfs\}$ be an entailment regime. Then,*

$$\begin{aligned} S \models_x E \quad &\textit{iff} \quad tr(S) \cup \Psi^x \models_{\mathsf{f}} tr(E), \textit{ and} \\ S \textit{ is } x\textit{-satisfiable} \quad &\textit{iff} \quad tr(S) \cup \Psi^x \textit{ has a model.} \end{aligned}$$

The following corollary follows immediately from Theorem 1 and the classical results about Skolemization (see e.g. [9]). For the case of s-entailment, the result was implicitly stated in [11, Skolemization lemma].

Corollary 1. *Let S and E be RDF graphs and let $x \in \{s, rdf, rdfs, erdfs\}$ be an entailment regime. Then,*

$$S \models_x E \quad \textit{iff} \quad tr(S)^{sk} \cup \Psi^x \models_{\mathsf{f}} tr(E).$$

[7] Even though typed literals are pairs in RDF, we treat them simply as constant symbols in our embedding.

$\Psi^s = \emptyset$

$\Psi^{rdf} = \Psi^s \cup \{tr(\langle s,p,o\rangle) \mid \langle s,p,o\rangle$ is an RDF axiomatic triple$\}\cup$
$\{wellxml(t) \mid t \in \mathcal{TL}$ is a well-typed XML literal$\}\cup$
$\{illxml(t) \mid t \in \mathcal{TL}$ is an ill-typed XML literal$\}\cup$
$\{\forall x(\exists y, z(y[x \twoheadrightarrow z]) \supset x[\texttt{type} \twoheadrightarrow \texttt{Property}]),$
$\forall x(wellxml(x) \supset x[\texttt{type} \twoheadrightarrow \texttt{XMLLiteral}]),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{XMLLiteral}] \wedge illxml(x) \supset \bot)\}$

$\Psi^{rdfs} = \Psi^{rdf} \cup \{tr(\langle s,p,o\rangle) \mid \langle s,p,o\rangle$ is an RDFS axiomatic triple$\}\cup$
$\{pl(t) \mid t \in \mathcal{PL}\}\cup$
$\{\forall x,y,z(x[y \twoheadrightarrow z] \supset x[\texttt{type} \twoheadrightarrow \texttt{Resource}] \wedge z[\texttt{type} \twoheadrightarrow \texttt{Resource}]),$
$\forall u,v,x,y(x[\texttt{domain} \twoheadrightarrow y] \wedge u[x \twoheadrightarrow v] \supset u[\texttt{type} \twoheadrightarrow y]),$
$\forall u,v,x,y(x[\texttt{range} \twoheadrightarrow y] \wedge u[x \twoheadrightarrow v] \supset v[\texttt{type} \twoheadrightarrow y]),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{Property}] \supset x[\texttt{subPropertyOf} \twoheadrightarrow x]),$
$\forall x,y,z(x[\texttt{subPropertyOf} \twoheadrightarrow y] \wedge y[\texttt{subPropertyOf} \twoheadrightarrow z] \supset x[\texttt{subPropertyOf} \twoheadrightarrow z]),$
$\forall x,y(x[\texttt{subPropertyOf} \twoheadrightarrow y] \supset \forall z_1,z_2(z_1[x \twoheadrightarrow z_2] \supset z_1[y \twoheadrightarrow z_2])),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{Class}] \supset x[\texttt{subClassOf} \twoheadrightarrow \texttt{Resource}]),$
$\forall x,y(x[\texttt{subClassOf} \twoheadrightarrow y] \supset \forall z(z[\texttt{type} \twoheadrightarrow x] \supset z[\texttt{type} \twoheadrightarrow y])),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{Class}] \supset x[\texttt{subClassOf} \twoheadrightarrow x]),$
$\forall x,y,z(x[\texttt{subClassOf} \twoheadrightarrow y] \wedge y[\texttt{subClassOf} \twoheadrightarrow z] \supset x[\texttt{subClassOf} \twoheadrightarrow z]),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{ContainerMembershipProperty}] \supset x[\texttt{subPropertyOf} \twoheadrightarrow \texttt{member}]),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{Datatype}] \supset x[\texttt{subClassOf} \twoheadrightarrow \texttt{Literal}]),$
$\forall x(pl(x) \supset x[\texttt{type} \twoheadrightarrow \texttt{Literal}]),$
$\forall x(x[\texttt{type} \twoheadrightarrow \texttt{Literal}] \wedge illxml(x) \supset \bot)\}$

$\Psi^{erdfs} = \Psi^{rdfs} \cup \{\forall x,y(\forall u,v(u[x \twoheadrightarrow v] \supset u[\texttt{type} \twoheadrightarrow y]) \supset x[\texttt{domain} \twoheadrightarrow y]),$
$\forall x,y(\forall u,v(u[x \twoheadrightarrow v] \supset v[\texttt{type} \twoheadrightarrow y]) \supset x[\texttt{range} \twoheadrightarrow y]),$
$\forall x,y(x[\texttt{type} \twoheadrightarrow \texttt{Property}] \wedge y[\texttt{type} \twoheadrightarrow \texttt{Property}] \wedge \forall u,v(u[x \twoheadrightarrow v] \supset$
$u[y \twoheadrightarrow v]) \supset x[\texttt{subPropertyOf} \twoheadrightarrow y]),$
$\forall x,y(x[\texttt{type} \twoheadrightarrow \texttt{Class}] \wedge y[\texttt{type} \twoheadrightarrow \texttt{Class}] \wedge \forall u(u[\texttt{type} \twoheadrightarrow x] \supset u[\texttt{type} \twoheadrightarrow y]) \supset$
$x[\texttt{subClassOf} \twoheadrightarrow y])\}$

Table 2. Axiomatization of the RDF entailment regimes

Since, by Proposition 2, $tr(S)^{sk}$, $tr(S)^{sk} \cup \Psi^{rdf}$ and $tr(S)^{sk} \cup \Psi^{rdfs}$ are equivalent to sets of Datalog formulas, this result implies that simple, RDF, and RDFS entailment can be computed using existing F-Logic rule reasoners[8] such as FLORA-2, and Ontobroker, as well as any rule reasoners which supports Datalog (see Proposition 1). Notice that, in the corollary, $tr(E)$ can be seen as a boolean conjunctive query (i.e. a yes/no query) in which the existentially quantified variables in $tr(E)$ are the non-distinguished variables.

We now consider an alternative, direct embedding of the extensional RDFS semantics (erdfs-entailment) which eliminates part of the RDFS vocabulary from the embedded graph, yielding a set of Datalog formulas.

We first define the notion of *nonstandard use* of the RDFS vocabulary, which intuitively corresponds to using the vocabulary in locations where it has not been intended,

[8] Note that a Datalog formula with $\bot$ in the antecedent corresponds to an integrity constraint, i.e. a query which may not have an answer set.

for example in places where it redefines the semantics of RDF constructs such as in the triple $\langle$`type`, `subPropertyOf`, $a\rangle$.

We say that a term t occurs in a *property position* if it occurs as the predicate of a triple, as the subject or object of a `subPropertyOf` triple, as the subject of a `domain` or `range` triple, or as the subject of a triple $\langle t, \texttt{type}, \texttt{Property}\rangle$ or $\langle t, \texttt{type}, \texttt{ContainerMembershipProperty}\rangle$. A term t occurs in a *class position* if it occurs as the subject or object of a `subClassOf` triple, as the object of a `domain`, `range`, or `type` triple, as the subject of a triple $\langle t, \texttt{type}, \texttt{Class}\rangle$ or $\langle t, \texttt{type}, \texttt{Datatype}\rangle$. Otherwise, we say that t occurs in an *individual position*.

Definition 2. *Let S be an RDF graph. Then, S has* nonstandard use *of the RDFS vocabulary if*
– `type`, `subClassOf`, `domain`, `range` *or* `subPropertyOf` *occurs in the subject or object position of a triple in S or*
– `ContainerMembershipProperty`, `Resource`, `Class`, `Datatype` *or* `Property` *occurs in any position other than the object position of a* `type`*-triple in S.*

We conjecture that large classes of RDF graphs will not have any nonstandard use of the RDFS vocabulary. We now proceed to define a direct embedding of the extensional RDFS entailment regime in F-Logic.

Definition 3. *Let $\langle s, p, o\rangle$ be an RDF triple. Then,*

$$\begin{aligned}
tr^{erdfs}(\langle s, \texttt{type}, \texttt{Datatype}\rangle) &= \forall x(x : s \supset x : \texttt{Literal}),\\
tr^{erdfs}(\langle s, \texttt{type}, o\rangle) &= s : o,\\
tr^{erdfs}(\langle s, \texttt{subClassOf}, o\rangle) &= \forall x(x : s \supset x : o),\\
tr^{erdfs}(\langle s, \texttt{subPropertyOf}, o\rangle) &= \forall x, y(x[s \twoheadrightarrow y] \supset x[o \twoheadrightarrow y]),\\
tr^{erdfs}(\langle s, \texttt{domain}, o\rangle) &= \forall x, y(x[s \twoheadrightarrow y] \supset x : o),\\
tr^{erdfs}(\langle s, \texttt{range}, o\rangle) &= \forall x, y(x[s \twoheadrightarrow y] \supset y : o), \textit{ and}\\
tr^{erdfs}(\langle s, p, o\rangle) &= s[p \twoheadrightarrow o], \textit{ otherwise.}
\end{aligned}$$

Let S be an RDF graph of a vocabulary $V = \langle \mathcal{F}, \mathcal{PL}, \mathcal{TL}, \mathcal{B}\rangle$. Then,

$$\begin{aligned}
tr^{erdfs}(S) &= \{\exists\, bl(S)(\textstyle\bigwedge\{tr^{erdfs}(\langle s, p, o\rangle) \mid \langle s, p, o\rangle \in S\})\}, \textit{ and}\\
\Psi^{erdfs-V} &= \{tr^{erdfs}(\langle s, p, o\rangle) \mid \langle s, p, o\rangle \textit{ is an RDF(S) axiomatic triple with no nonstandard use of the RDF(S) vocabulary}\} \cup \{t : \texttt{XMLLiteral} \mid t \in \mathcal{TL} \textit{ is a well-typed XML literal}\} \cup \{illxml(t) \mid t \in \mathcal{TL} \textit{ is an ill-typed XML literal}\} \cup\\
&\quad \{t : \texttt{Literal} \mid t \in \mathcal{PL}\} \cup \{\forall x(x : \texttt{Literal} \wedge illxml(x) \supset \bot)\})
\end{aligned}$$

The *property* (resp., *class*) *vocabulary* of an RDF graph S consists of all the symbols occurring in property (resp., class) positions in S and the RDF(S) axiomatic triples with no nonstandard use of the RDF(S) vocabulary.

Given two RDF graphs S and E, we write $E \trianglelefteq S$ if the property and class vocabularies of E are subsets of the property and class vocabularies of S (modulo blank node renaming and instantiation, i.e. replacement of blank nodes with URI references or literals) and `Resource`, `ContainerMembershipProperty`, `Class`, `Property` and `Datatype` do not occur in E

Theorem 2. *Let S, E be RDF graphs with no nonstandard use of the RDFS vocabulary such that* `Resource`*,* `Class`*,* `Property`*,* `ContainerMembershipProperty` *and* `Datatype` *do not occur in E. Then,*

- *whenever $E \trianglelefteq S$,*

$$S \models_{erdfs} E \text{ iff } tr^{erdfs}(S) \cup \Psi^{erdfs-V} \models_{\mathsf{f}} tr^{erdfs}(E);$$

- *$(tr^{erdfs}(S))^{sk}$ is a conjunction of F-Logic Datalog formulas, and whenever E does not contain the terms* `subClassOf`*,* `domain`*,* `range`*, and* `subPropertyOf`*, $tr^{erdfs}(E)$ is a conjunction of atomic molecules prefixed by an existential quantifier and*

$$S \models_{erdfs} E \text{ iff } (tr^{erdfs}(S))^{sk} \cup \Psi^{erdfs-V} \models_{\mathsf{f}} tr^{erdfs}(E).$$

Since $(tr^{erdfs}(S))^{sk} \cup \Psi^{erdfs-V}$ is a set of Datalog formulas we have that, if the RDF graphs fulfill certain conditions, query answering techniques from the area of deductive databases can be used for checking extensional RDFS entailment.

3.2 Embedding Extensional RDFS in First-Order Logic

We now consider an embedding of extensional RDFS entailment in FOL, based on the direct embedding of extensional RDFS in F-Logic considered above (Definition 3).

An F-Logic theory or formula is *translatable* to contextual FOL if for molecules of the forms $t_1[t_2 \twoheadrightarrow t_3]$ and $t_1 : t_2$ holds that t_2 is a constant symbol (i.e. 0-ary function symbol).

Let Φ (resp., ϕ) be an F-Logic theory (resp., formula) which is translatable to contextual FOL, then $(\Phi)^{FO}$ (resp, $(\phi)^{FO}$) is the contextual FOL theory obtained from Φ (resp., ϕ) by:
– replacing every data molecule $t_1[t_2 \twoheadrightarrow t_3]$ with $t_2(t_1, t_3)$, and
– replacing every is-a molecule $t_1 : t_2$ with $t_2(t_1)$.
The following proposition follows immediately from a result in [6].

Proposition 3. *Let Φ, ϕ be an equality-free F-Logic theory and formula which are translatable to contextual FOL. Then,*

$$\Phi \models_{\mathsf{f}} \phi \text{ iff } (\Phi)^{FO} \models_{c} (\phi)^{FO}.$$

An RDF graph S is a *non-higher-order* RDF graph if S does not contain blank nodes in class and property positions, and does not contain nonstandard use of the RDFS vocabulary. A non-higher-order RDF graph S is a *classical* RDF graph if the sets of URIs occurring in individual, class and property positions in S and its context (e.g. entailing or entailed graph) are mutually disjoint. Notice that every ground RDF graph which does not contain nonstandard use of the RDFS vocabulary is a non-higher-order RDF graph. One can also verify that every OWL DL graph, as defined in [19], is a classical RDF graph, but there are classical RDF graphs which are not OWL DL graphs.

The following theorem identifies subsets of extensional RDFS which have a natural correspondence to contextual and classical FOL. Observe that if S is a non-higher-order RDF graph, then $tr^{erdfs}(S)$ is translatable to contextual FOL.

Theorem 3. *Let S and E be non-higher-order RDF graphs such that $E \trianglelefteq S$. Then,*

$$S \models_{erdfs} E \text{ iff } (tr^{erdfs}(S))^{FO} \models_c (tr^{erdfs}(E))^{FO}.$$

If, additionally, S, E are classical graphs, then $(tr^{erdfs}(S))^{FO}$ and $(tr^{erdfs}(E))^{FO}$ are theories of classical first-order logic, and

$$S \models_{erdfs} E \text{ iff } (tr^{erdfs}(S))^{FO} \models (tr^{erdfs}(E))^{FO}.$$

Proposition 4. *Let S be a ground non-higher-order graph*[9]*. Then, $(tr^{erdfs}(S))^{FO}$ can be equivalently rewritten to the FOL equivalent Φ of a contextual $DL\text{-}Lite_{\mathcal{R}}$ knowledge base $\mathcal{K}$.*

If S is a classical RDF graph, then $(tr^{erdfs}(S))^{FO}$ can be equivalently rewritten to the FOL equivalent Φ of a classical $DL\text{-}Lite_{\mathcal{R}}$ knowledge base.

4 Complexity

In this section we review the complexity of the various forms of entailment in RDF and present several novel results, based on the embeddings presented in the previous section.

The complexity of simple entailment and RDFS entailment is well known, and the complexity of RDF entailment follows immediately. Note that, although the set of axiomatic triples is infinite, only a finite subset, linear in the size of the graphs, needs to be taken into account when checking entailment.

Proposition 5 ([10, 13, 4]). *Let S and E be graphs. Then, the problem of checking $S \models_s E$, $S \models_{rdf} E$, or $S \models_{rdfs} E$ is NP-complete in the combined size of the graphs, and polynomial in the size of S. If E is ground, then the respective problems are polynomial in the combined size of the graphs.*
Additionally, the problem of checking $S \models_{erdfs} E$ is NP-hard in the size of the graphs.

The membership proofs in [10, 13, 4] rely on the fact that the set of all (relevant) entailed triples of a given graph can be computed in polynomial time using the RDFS entailment rules [13]; the problem can then be reduced to subgraph homeomorphism. Using Corollary 1 and the fact that the problem of checking ground entailment in Datalog [8] is polynomial in the size of the data ($tr(S)$) gives to a novel argument for membership.

NP-hardness can be shown through a reduction from a known NP-hard problem (e.g. 3SAT).

From the embedding in F-Logic, together with the complexity of nonrecursive Datalog [8], we obtain the following novel characterization of the complexity of simple and RDF entailment.

[9] Note that, when considering a variant of $DL\text{-}Lite_{\mathcal{R}}$ which allows existentials in the ABox – also allowed in OWL DL – this restriction could be relaxed to S being a non-higher-order RDF graph with no blank nodes outside of individual positions.

Theorem 4. *Let S and E be RDF graphs. Then, the problems $S \models_s E$ and $S \models_{rdf} E$ are in LogSpace in the size of S, and in the combined size of the graphs if E is ground.*

Using the correspondence of Proposition 4, the results on the complexity of reasoning in $DL\text{-}Lite_{\mathcal{R}}$ [7], and the classical results on skolemization [9] we obtain the following result for extensional RDFS entailment.

Theorem 5. *Let S and E be RDF graphs with no nonstandard use of the RDFS vocabulary such that $E \trianglelefteq S$. Then, the problem of deciding $S \models_{erdfs} E$ is NP-complete in the size of the graphs, and polynomial if E is ground.*

Entailment	Restrictions on S	Restrictions on E	Complexity
$\models_s$,$\models_{rdf}$,$\models_{rdfs}$	none	none	NP-complete
$\models_s$,$\models_{rdf}$	none	ground	LogSpace
$\models_{rdfs}$	none	ground	P
$\models_{erdfs}$	none	none	NP-hard
$\models_{erdfs}$	no nonst. RDFS	no nonst. RDFS	NP-complete
$\models_{erdfs}$	no nonst. RDFS	ground, no nonst. RDFS	P

Table 3. Complexity of Entailment $S \models_x E$ in RDF, measured in the size of S, E

Table 3 summarizes the complexity of reasoning with the entailment regimes of RDF; "No nonst. RDFS" stands for "no nonstandard use of the RDFS vocabulary; S and E are such that the property and class vocabularies of E are subsets of the property and class vocabularies of S (modulo blank node renaming and instantiation); and `Resource`, `Class`, `Property`, `ContainerMembershipProperty` and `Datatype` do not occur in E". The results in the first and third line of the table were obtained in [10, 4, 13], and the fourth line follows immediately. To the best of our knowledge, the other results are novel.

5 Querying

In this section we consider conjunctive queries over RDF graphs using the RDF entailment regimes we considered throughout this paper.

Given a countable set $\mathcal{V}$ of variable symbols, disjoint from the symbols in V, *a generalized RDF triple* is a tuple of the form $\langle s, p, o\rangle$, with s, p and o terms or variable symbols. A *conjunctive query* $q(\boldsymbol{x})$ over an RDF graph S is a set of generalized RDF triples $q(\boldsymbol{x})$ such that $\boldsymbol{x}$ is a vector of variables occurring in q, also called the *distinguished variables* of q; the blank nodes occurring in q, $bl(q)$, are the *non-distinguished variables* of q.

Given an RDF graph S and conjunctive query $q(\boldsymbol{x})$, then a tuple of terms $\boldsymbol{a}$ is an *answer* of a query under x-entailment if $S \models_x q(\boldsymbol{a})$, with $x \in \{s, rdf, rdfs, erdfs\}$ an entailment regime. The complexity of query answering is related to the complexity of the corresponding recognition problem: the recognition problem associated with a

query $q(\boldsymbol{x})$ is the decision problem of checking whether, given an RDF graph S and a tuple $\boldsymbol{a}$ of terms, the entailment $S \models_x q(\boldsymbol{a})$ holds. The *data complexity* of query answering under the x entailment regime corresponds to the complexity of the corresponding recognition problem, in the size of S.

From the preceding results on the complexity of the various entailment regimes we obtain the following characterization of the complexity of query answering.

Theorem 6. *Let S be an RDF graph, let $x \in \{s, rdf, rdfs, erdfs\}$ be an entailment regime, and let $q(\boldsymbol{x})$ be a conjunctive query. Then, the data complexity of query answering under the x entailment regime is*

- *in LogSpace, if $x \in \{s, rdf\}$ and*
- *polynomial, if $x \in \{rdfs\}$.*

6 Discussion and Related Work

In this section we discuss implications of the results in this paper, and place it in the context of related work. We distinguish between work done on RDF and on RDF querying.

RDF There have been several investigations [10, 4, 13] into the semantics of RDF. The investigation in [10] reconstructs the semantics from a graph database perspective, and the one in [4] reconstructs the semantics from a logical language perspective. The investigation of the RDF semantics in [13] stays very close to the RDF specification. Additionally, [13] shows that the entailment rules presented in the original specification [11] are not complete with respect to the semantics. These reconstructions have led to a number of complexity results for RDF entailment. In this paper, we built upon these results and complemented them with several novel results for simple, RDF, and extensional RDFS entailment.

The investigation in [4] is close in spirit to our investigation, albeit that [4] bases its logical reconstruction on (contextual) first-order logic, rather than F-Logic.

RDFS(FA) [18] defines a new (extensional) semantics for RDFS which is in line with the semantics of OWL DL, as well as a number of syntactic restrictions to achieve a layered meta-modeling architecture. It is currently not known what the precise relationship is between RDFS(FA) and the RDFS semantics defined in the standard [11].

Finally, we mention [17], in which the authors identify a syntactic subset of RDFS which allows for efficient reasoning ($O(n \log n)$), while still being expressive enough to capture large classes of ontologies.

RDF Querying SPARQL [22] is a query language for RDF, currently under development at W3C. Of all the RDF entailment regimes, SPARQL currently only considers simple entailment. However, the use of other regimes is considered a possible future extension.

The queries we considered in Section 5 are conjunctive queries and correspond to what are called "[SPARLQ] graph pattern expressions constructed by using only AND" in [20]. Therefore, not surprisingly, data complexity of conjunctive query answering

when considering simple entailment corresponds to the data complexity of evaluating such graph pattern expressions; they are both in LogSpace (cf. Theorem 6 and [20, Theorem 4]).

Finally we mention [21], in which a translation from SPARQL queries to Datalog is described. The combination of such a translation with an embedding of the RDF or RDFS semantics, as described in Theorem 1, could be used for evaluating SPARQL queries using the respective entailment regimes.

7 Conclusions and Future Work

We have presented embeddings of the different RDF entailment regimes in F-Logic, and we have shown how deductive database and description logic technology can be used for reasoning with RDF. An implementation of answering conjunctive queries over RDF graphs under the RDF and RDFS entailment regimes, and restricted RDF graphs under the eRDFS entailment regime, based on the Datalog reasoner IRIS[10], can be found at: `http://tools.deri.org/rdfs-reasoner`. It is planned to extend this reasoner with support for more expressive query languages, such as SPARQL, considering the embedding in Datalog presented in [21].

In the course of our investigation we have presented several novel complexity results. To the best of our knowledge, ours is the first comprehensive investigation of the extensional RDFS entailment regime. These results could be used for, for example, rule-based extensions of RDF, or increasing the alignment between RDF and logic-based semantic Web languages (e.g. OWL DL).

Our future work includes the extension of the considered embeddings with more extensive treatment of datatypes, in the form of *D*-entailment [11], and *D**-entailment [13], as well as more expressive query languages such as SPARQL.

Several proposals have been made for rule extensions of RDF graphs (e.g. [2, 14, 1]), and several rule-based systems which deal with RDF exist (e.g. Jena, CWM). In an earlier version of the present paper [5] we considered extensions of RDF graphs with logical rules and axioms, based on the embeddings we have presented. However, such extensions are not entirely faithful with respect to the model-theoretic semantics of RDF. Therefore, our future work includes an investigation of combinations of logical rules and RDF based on a notion of common models, i.e. an interpretation is a model of a combination if it is a model of both the logical theory and the RDF graph.

Finally, we plan to investigate the precise relationship between eRDFS and OWL DL entailment, taking the subset compatible with $DL\text{-}Lite_{\mathcal{R}}$ (see Proposition 4) as a starting point.

References

1. Analyti, A., Antoniou, G., Damásio, C.V., Wagner, G.: Stable model theory for extended RDF ontologies. In: Proceedings of the 4th International Semantic Web Conference (ISWC 2005). (2005) 21–36

[10] `http://sourceforge.net/projects/iris-reasoner`

2. Berners-Lee, T., Connolly, D., Kagal, L., Scharfand, Y., Hendler, J.: N3Logic: A logic for the web. Journal of Theory and Practice of Logic Programming (TPLP), Special Issue on Logic Programming and the Web (2007)
3. Brickley, D., Guha, R.V.: RDF vocabulary description language 1.0: RDF schema. Recommendation 10 February 2004, W3C (2004)
4. Bruijn, J. de, Franconi, E., Tessaris, S.: Logical reconstruction of normative RDF. In: Proceedings of the Workshop OWL: Experiences and Directions (OWLED-2005). (2005)
5. Bruijn, J. de, Heymans, S.: RDF and logic: Reasoning and extension. In: Proceedings of the 6th International Workshop on Web Semantics (WebS 2007), in conjunction with the 18th International Conference on Database and Expert Systems Applications (DEXA 2007). (2007)
6. Bruijn, J. de, Heymans, S.: On the relationship between description logic-based and f-logic-based ontologies. Fundamenta Informaticae (2008) Accepted for publication.
7. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. In: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR2006). (2006) 260–270
8. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Computing Surveys (CSUR) **33**(3) (2001) 374–425
9. Fitting, M.: First Order Logic and Automated Theorem Proving (second edition). Springer Verlag (1996)
10. Gutierrez, C., Hurtado, C., Mendelzon, A.O.: Foundations of semantic web databases. In: Proceedings of the 23rd ACM Symposium on Principles of Database Systems (PODS2004). (2004) 95–106
11. Hayes, P.: RDF semantics. Recommendation 10 February 2004, W3C (2004)
12. Horrocks, I., Patel-Schneider, P.F., Harmelen, F. van: From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics **1**(1) (2003) 7–26
13. Horst, H.J. ter: Completeness, decidability and complexity of entailment for RDF schema and a semantic extension involving the OWL vocabulary. Journal of Web Semantics **3**(2–3) (2005) 79–115
14. Horst, H.J. ter: Combining RDF and part of OWL with rules: Semantics, decidability, complexity. In: Proceedings of the 4th International Semantic Web Conference (ISWC 2005). (2005)
15. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. Journal of the ACM **42**(4) (1995) 741–843
16. Klyne, G., Carroll, J.J.: Resource description framework (RDF): Concepts and abstract syntax. Recommendation 10 February 2004, W3C (2004)
17. Muñoz, S., Pérez, J., Gutierrez, C.: Minimal deductive systems for RDF. In: Proceedings of the 4th European Semantic Web Conference (ESWC2007). (2007)
18. Pan, J.Z., Horrocks, I.: RDFS(FA): Connecting RDF(S) and OWL DL. **19**(2) (2007) 192–206
19. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax. Recommendation 10 February 2004, W3C (2004)
20. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. In: Proceedings of the 5th International Semantic Web Conference (ISWC 2006). (2006) 30–43
21. Polleres, A.: From SPARQL to rules (and back). In: Proceedings of the 16th International World Wide Web Conference (WWW2007). (2007) 787–796
22. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. Working Draft 26 March 2007, W3C (2007)