RDF Entailment as a Graph Homomorphism

Jean-François Baget

RDF Entailment as a Graph Homomorphism

Jean-François Baget

INRIA Rhône-Alpes
655 avenue de l'Europe
38334 Saint Ismier
France
jean-francois.baget@inrialpes.fr

Abstract. Semantic consequence (entailment) in RDF is ususally computed using Pat Hayes Interpolation Lemma. In this paper, we reformulate this mechanism as a graph homomorphism known as projection in the conceptual graphs community.
Though most of the paper is devoted to a detailed proof of this result, we discuss the immediate benefits of this reformulation: it is now easy to translate results from different communities (*e.g.* conceptual graphs, constraint programming, ...) to obtain new polynomial cases for the NP-complete RDF ENTAILMENT problem, as well as numerous algorithmic optimizations.

1 Introduction

Simple RDF is the knowledge representation language on which RDF (Resource Description Framework) and its extension RDFS are built. As a logic, it is provided with a syntax (its abstract syntax will be used here), and model theoretic semantics [1]. These semantics are used to define entailments: an RDF graph G entails an RDF graph H iff H is true whenever G is. However, since an infinity of interpretations must be evaluated according to this definition, an operational inference mechanism, sound and complete w.r.t. entailment, is needed. This is the purpose of the interpolation lemma [1]: a finite procedure characterizing entailments. It has been extended to take into account more expressive languages (RDF, RDFS [1], and other languages *e.g.* [2]). All these extensions rely on a polynomial-time initial treatment of the graphs, the hard kernel remaining the basic simple entailment, which is a NP-hard problem.

In this paper, we intend to contribute to the study of this fundamental simple entailment by reformulating it as a graph homomorphism, extensively studied both in mathematics and in graph theory. This will allow the RDF community to import numerous results from related problems: colored homomorphisms [3], conceptual graphs projection [4], or constraint satisfaction problems [5]. The experience acquired during the last 20 years in these different communities can help us to quickly develop efficient algorithms for RDF entailment, as well as understand what are the polynomial cases for this problem.

However, the bulk of the paper presented here is devoted to the reformulation as a graph homomorphism itself. All necessary proofs have been included,

independently from the proof of the interpolation lemma [1]. Indeed, we believe that our proof framework can be used as a basis to apply our reformulation to many extensions of simple RDF: in that case, an in-depth understanding of that proof is required.

Section 2 is devoted to the basic definitions and results of [1]. In section 3, we reformulate the interpolation lemma as a directed, multigraph homomorphism. Section 4 provides a standalone proof of this result, via a reformulation of entailment as a directed hypergraph homomorphism. In section 5, we provide a list of results that can be translated to simple RDF entailments. Finally, in section 6, we discuss the advantages and limitations of this approach.

2 Simple RDF: Syntax, Semantics, and Inferences

This section presents *simple RDF*, the basic logic on which RDF and RDFS are built: we recall here definitions and results presented in [1]. We first present the abstract syntax of RDF: note that we distinguish here an RDF *tripleset* (a set of triples) from its associated graph (that will be presented in the next section). The semantics of RDF triplesets allows to formally define the notion of entailment, that is characterized by the interpolation lemma. Note also that though we use here the terms of interpretations, entailment, it refers here without ambiguity to what is called simple interpretation or simple entailment in [1]. These definitions precise our notations, and the examples given introduce the running example used all along this paper. The reader should refer to [1] for more explanations.

2.1 RDF Abstract Syntax

We consider a set of *terms* $\mathcal{V}$ partitioned in three pairwise disjoint sets: a set $\mathcal{U}$ of *URI references* (or *urirefs*), a set $\mathcal{B}$ of *blanks*, and a set $\mathcal{L}$ of literals (itself partitioned into two disjoint sets, the set $\mathcal{L}_P$ of plain literals and the set $\mathcal{L}_T$ of typed literals). Let $V \subseteq \mathcal{V}$ be a subset of $\mathcal{V}$, then we denote by $\mathcal{U}(V)$ (resp. $\mathcal{B}(V)$, $\mathcal{L}_P(V)$, $\mathcal{L}(V)$, $\mathcal{L}_T(V)$) the set of urirefs of V (resp. of blanks of V, of literals of V, of plain literals of V, of typed literals of V). Without loss of generality, and for the sake of simplicity, we have not taken language tags into account here.

Definition 1 (RDF tripleset). *An* RDF tripleset *is a subset of* $(\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times \mathcal{V}$. *Its elements are called* RDF triples.

An RDF triple $\langle s, p, o \rangle$ can be read "there is a relation of sort p whose subject is the entity s an whose object is the entity o". Let G be an RDF tripleset. We denote by $\mathcal{V}(G)$ the terms of $\mathcal{V}$ that appear in any triple of G, *i.e.* $\mathcal{V}(G) = \{v \in \mathcal{V} \mid \exists \langle s, p, o \rangle \in G,\ x = s \text{ or } x = p \text{ or } x = o\}$.

Example 1. Let $V = \{u_1, u_2, b_1, b_2, l\}$ be a set of terms where u_1 and u_2 are urirefs, b_1 and b_2 are blanks, and l is a plain literal. Let us now consider the two following RDF triplesets, used as a running example along this paper:

- $H = \{\langle u_1, u_1, b_1 \rangle\ \langle u_1, u_1, b_2 \rangle, \langle b_2, u_2, l \rangle, \langle b_1, u_1, b_2 \rangle\}$
- $G = \{\langle u_1, u_1, b_1 \rangle\ \langle b_1, u_1, b_1 \rangle, \langle b_1, u_2, l \rangle, \langle u_1, u_2, u_2 \rangle\ \}$

2.2 Interpretations

Definition 2 (Simple Interpretations). *Let V be a set of terms. An* interpretation *of V is a 5-tuple $\langle I_R, I_P, \iota_{ext}, \iota_s, \iota_l \rangle$ where I_R is a set of* resources *containing $\mathcal{L}_P(V)$[1], I_P is a set of* properties, *$\iota_{ext} : I_P \to 2^{I_R \times I_R}$ maps each property to a set of pairs of resources (the* extension *of the property), $\iota_s : \mathcal{U}(V) \to I_R \cup I_P$ maps each uriref to a resource or a property, and $\iota_l : \mathcal{L}_T(V) \to I_R$ maps each typed literal to a resource.*

Example 2. Let V be the set of terms defined in Ex. 1. We consider the following interpretation $I = \langle I_R, I_P, \iota_{ext}, \iota_s, \iota_l \rangle$ of V defined by:

- $I_R = \{\clubsuit, \heartsuit, l\}$;
- $I_P = \{\clubsuit, \heartsuit\}$;
- $\iota_{ext}(\clubsuit) = \{\langle \clubsuit, \heartsuit \rangle, \langle \heartsuit, \heartsuit \rangle\}$ and $\iota_{ext}(\heartsuit) = \{\langle \heartsuit, l \rangle\}$;
- $\iota_s(u_1) = \clubsuit$ and $\iota_s(u_2) = \heartsuit$.

For the sake of clarity, it has been proposed in [1] to give a graphical representation of an interpretation as shown in Fig. 1.

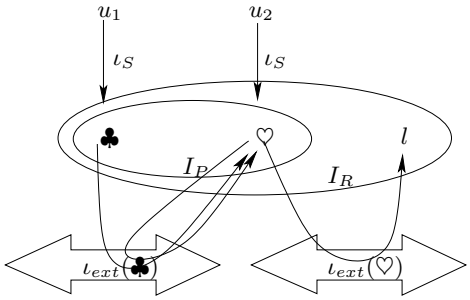

Fig. 1. A Graphical Representation of the Interpretation I.

Definition 3 (Models). *Let G be an RDF tripleset, and V be a set of terms that contains the set of terms of G, i.e. such that $(\mathcal{U}(\mathcal{V}(G)) \cup \mathcal{L}(\mathcal{V}(G))) \subseteq V$. An interpretation $\langle I_R, I_P, \iota_{ext}, \iota_s, \iota_l \rangle$ of V is a* model *of G iff there exists a mapping $\iota : \mathcal{V}(G) \to I_R \cup I_P$ such that:*

1. *for each plain literal $l \in \mathcal{L}_P(\mathcal{V}(G))$, $\iota(l) = l$;*
2. *for each typed literal $l \in \mathcal{L}_T(\mathcal{V}(G))$, $\iota(l) = \iota_l(l)$;*
3. *for each uriref $u \in \mathcal{U}(\mathcal{V}(G))$, $\iota(u) = \iota_s(u)$;*
4. *for each blank $b \in \mathcal{B}(\mathcal{V}(G))$, $\iota(b) \in I_R$;*

[1] This inclusion allows to avoid, for the sake of simplicity, the set LV of literal values.

5. *for each triple* $\langle s, p, o\rangle \in G$, $\langle \iota(s), \iota(o)\rangle \in \iota_{ext}(\iota(p))$.

Example 3. Let us show that the interpretation I in Ex. 2 is a model for the RDF tripleset H in Fig. 1. We have $\iota(u_1) = \clubsuit$, $\iota(u_2) = \heartsuit$, and $\iota(l) = l$ (all these values are constrained by the interpretation I). Our only choice is with the blanks: we chose $\iota(b_1) = \iota(b_2) = \heartsuit \in I_R$. It remains now to check that for each triple $\langle s, p, o\rangle$ of H, $\langle \iota(s), \iota(o)\rangle \in \iota_{ext}(\iota(p))$. We will only check the triple $\langle b_2, u_2, l\rangle$: $\iota_{ext}(\iota(u_2)) = \iota_{ext}(\heartsuit) = \{\langle \heartsuit, l\rangle\} \supseteq \langle \iota(b_2), \iota(l)\rangle$. The condition is also verified for the 4 other triples. It follows that I is a model of H. A tenacious reader can now check that I is not a model of G (he has to prove that no mapping ι respects these conditions).

2.3 The Interpolation Lemma

Definition 4 (Satisfiability, Entailment). *Let G and H be two RDF triplesets. We say that G is* satisfiable *if there exists an interpretation that is a model of G. We say that H is a* semantic consequence *of G (we also say that G* entails *H, and note $G \models H$) if every model of G is also a model of H.*

Example 4. The RDF tripleset H of Ex. 1 is satisfiable since the interpretation I of Ex. 2 is a model of H. Since I is not a model of G (Ex. 1), we can conclude that H does not entail G.

Definition 5 (Instance). *Let G be an RDF tripleset, and let V be a set of terms that contains the set of terms of G. Let us consider an* instance mapping *$\alpha : \mathcal{V}(G) \to V$ mapping each blank of G to a term of V, and each uriref or literal to itself. The RDF tripleset $G_\alpha = \{\langle \alpha(s), p, \alpha(o)\rangle \mid \langle s, p, o\rangle \in G\}$ is called an* instance *of G.*

Example 5. Let us consider the set of terms V and the RDF tripleset G of Ex. 1. We define an instance mapping α as follows: $\alpha(b_1) = b_1$, $\alpha(b_2) = b_1$ (every other element of V is mapped to itself). The instance H_α is the RDF tripleset defined by: $H_\alpha = \{\langle u_1, u_1, b_1\rangle, \langle b_1, u_2, l\rangle, \langle b_1, u_1, b_1\rangle\}$ (notice that a second occurence of $\langle u_1, u_1, b_1\rangle$ has been removed from the set).

Theorem 1 (Interpolation Lemma). *Let G and H be two RDF triplesets. Then $G \models H$ iff there exists an instance H' of H such that $H \subseteq G$.*

Example 6. Since the RDF tripleset H_α of Ex. 5 is a subset of the RDF tripleset G of Ex. 1, then $G \models H$.

3 RDF Triplesets as Directed, Labelled Multigraphs

RDF triplesets are given a standard graphical representation: the drawing of the graph (as a mathematical structure) associated with the tripleset (hence the usual name of RDF graphs). It is generally assumed that most people are more comfortable with this representation than with triples, at least when the graphs

involved are not to big. The graphs whose drawings correspond to this representation are directed, labelled multigraphs (there can be many arcs between two nodes, a requirement since two arcs can have different labels). In this section, we reformulate the usual characterization of entailment (the interpolation lemma of [1], expressed on the RDF tripleset) as a graph homomorphism: graphs are no longer only used for a representation purpose, but also for reasonings.

3.1 Standard Graphical Representation of an RDF Tripleset

Definition 6 (Directed, Labelled Multigraphs). *A* directed labelled multigraph *(or* M-graph*) over a set of terms V is a 4-tuple $G = \langle N, A, \gamma, \epsilon \rangle$ where N is a finite set of* nodes*, A is a finite set of* arcs*, $\gamma : A \to N \times N$ maps each arc to a pair of nodes called its* ends *(the first being the* origin *and the second the* destination*), and $\epsilon : N \cup A \to V$ maps each node and arc to a term.*

Let G be an RDF tripleset. We call *entities* of G the subset of $\mathcal{V}(G)$ that contains the terms appearing either as subject or object in a triple of G, *i.e.* $ent(G) = \{x \in \mathcal{V}(G) \mid \exists \langle s, p, o \rangle \in G,\ x = s \text{ or } x = o\}$ (it is called the *nodeset* in [2]). The M-graph $\mathcal{M}(G) = \langle N, A, \gamma, \epsilon \rangle$ associated with the RDF tripleset G is built as follows:

1. To each term $e \in ent(G)$ we associate a distinct node $m(e)$. Then $N = \{m(e) \mid e \in ent(G)\}$. Each node is labelled by the element of the set of terms associated to it: $\epsilon(m(e)) = e$.
2. To each triple $t = \langle s, p, o \rangle \in G$ we associate a distinct arc $m(t)$. Then $A = \{m(t) \mid t \in G\}$. The ends of the arc $m(t)$ are the nodes associated with the subject and the object of the triple t: $\gamma(m(t)) = \langle m(s), m(o) \rangle$. The label of the arc $m(t)$ is the property of the triple t: $\epsilon(m(t)) = p$.

Example 7. The M-graph $\mathcal{M}(H) = \langle N, A, \gamma, \epsilon \rangle$ obtained from the graph H of Ex. 1 is defined by: $N = \{1, 2, 3, 4\}$, $A = \{a, b, c, d\}$, $\gamma(a) = \langle 1, 2 \rangle$, $\gamma(b) = \langle 2, 3 \rangle$, $\gamma(c) = \langle 1, 3 \rangle$, $\gamma(d) = \langle 3, 4 \rangle$, $\epsilon(1) = u_1$, $\epsilon(2) = b_1$, $\epsilon(3) = b_2$, $\epsilon(4) = l$, $\epsilon(a) = u_1$, $\epsilon(b) = u_1$, $\epsilon(c) = u_1$, and $\epsilon(d) = u_2$.

The M-graph $\mathcal{M}(G)$ associated with an RDF tripleset G can be drawn as follows: each node labelled by a uriref or a blank is represented by an oval, and each node labelled by a literal is represented by a rectangle. The label of the node is written inside the oval or rectangle associated to it (it is not mandatory to write the label when it is a blank). Each arc a with $\gamma(a) = \langle x, y \rangle$ is represented by an arrow from the figure associated with x to the figure associated with y. The label $\epsilon(a)$ is written next to this arrow.

Example 8. FIG. 2 represents the drawing of the M-graph $\mathcal{M}(H)$ of Ex. 7 (usually conflated with the RDF tripleset H itself).

Note that the complexity in both time and space of the transformation $\mathcal{M}$ is linear in the size of the tripleset if the graph is encoded by an adjacence list, and is quadratic if it is encoded by an incidence matrix.

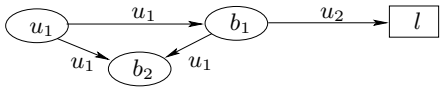

Fig. 2. Drawing of the M-graph associated with an RDF tripleset.

3.2 Simple Entailment as a Multigraph Homomorphism

Here we caracterize RDF entailment as a M-graph homomorphism. Graphs homomorphisms have been extensively studied in mathematics as well as in computer science (*e.g.* [3]), though the generalization we use here is more akin to the projection used to caracterize entailment of conceptual graphs (CGs) [4]. Though we have decided not to present our results via a translation to CGs, the reader can refer to [6] or [7] for precise relationships between RDF and CGs.

Definition 7 (Directed, Labelled Multigraph Homomorphism). *Let $G = \langle N, A, \gamma, \epsilon\rangle$ and $G' = \langle N', A', \gamma', \epsilon'\rangle$ be M-graphs over a set of terms V. Let $\leq$ be a preorder over V. A* directed, labelled multigraph homomorphism *according to $\leq$ (or* $\leq$-M-morphism*) from G into G' is a mapping $\pi : N \to N'$ that preserves the preorder on labels as well as incidence of arcs, i.e.:*

1. *for each $n \in N$, $\epsilon'(\pi(n)) \leq \epsilon(n)$;*
2. *for each $a \in A$ with $\gamma(a) = \langle s, o\rangle$, $\exists a' \in A'$ such that $\gamma'(a') = \langle \pi(s), \pi(o)\rangle$ and $\epsilon'(a') \leq \epsilon(a)$.*

Example 9. Let us consider the M-graphs associated with the triplesets G and H of Ex. 1. Now we define $\leq_1$ as the smallest preorder defined on the set of terms V fulfilling these conditions:

- for each two blanks b_1 and b_2, $b_1 \leq_1 b_2$;
- for each blank b and each uriref or literal c, $c \leq_1 b$.

It implies that urirefs and literals are pairwise non comparable. Then there exists a $\leq_1$-M-morphism from H into G, illustrated by the dashed arrows in FIG. 3.

It remains now to prove that such a $\leq_1$-M-morphism caracterizes simple RDF entailment.

Theorem 2. *Let G and H be two RDF triplesets defined over a set of terms V. Let $\leq_1$ be the partial order on V defined in Ex. 9. Then $G \models H$ if and only if there is a $\leq_1$-M-morphism from $\mathcal{M}(H)$ into $\mathcal{M}(G)$.*

As proven below, this is a mere reformulation of the interpolation lemma. Next section provides a standalone proof (that can be considered as an another proof for the interpolation lemma). Further sections will be devoted to the advantages of this reformulation (complexity and algorithms).

Proof. We use the interpolation lemma to prove both directions of the equivalence:

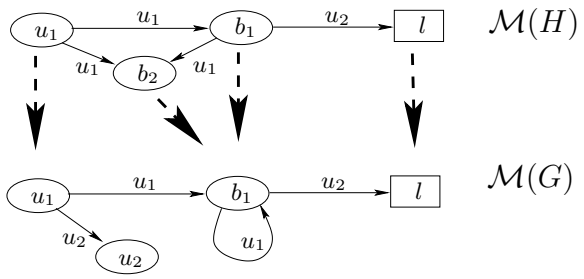

Fig. 3. $\leq_1$-M-morphism from the M-graph $\mathcal{M}(H)$ into the M-graph $\mathcal{M}(G)$.

($\Rightarrow$) Suppose $G \models_s H$. Then there exists an instance mapping α such that $H_\alpha \subseteq G$. Let us consider the mapping π from the nodes of $\mathcal{M}(H)$ into the nodes of $\mathcal{M}(G)$ defined as follows: for each node $m(x)$ of $\mathcal{M}(H)$ (*i.e.* associated with the term x in $\mathcal{V}(H)$), $\pi(m(x)) = m(\alpha(x))$ (where $m(\alpha(x))$ is the node of $\mathcal{M}(G)$ associated with the term $\alpha(x)$). Let us now prove that π is a $\leq_1$-M-morphism from $\mathcal{M}(H)$ into $\mathcal{M}(G)$.

1. We first prove that π preserves the preorder on nodes labels. Let $m(x)$ be an arbitrary node of $\mathcal{M}(H)$. The label of $m(x)$ is x. The label of $\pi(m(x))$ is the label of $m(\alpha(x))$, *i.e.* $\alpha(x)$. It remains to show that $\alpha(x) \leq_1 x$. If x is a blank, then it is greater than anything else (def. of $\leq_1$). Otherwise, $\alpha(x) = x$. In both cases, $\alpha(x) \leq_1 x$.
2. Finally, we prove that π preserves the incidence of arcs and the preorder on their labels. Let a be an arc of H with $\gamma(a) = \langle m(s), m(o)\rangle$ and $\epsilon(a) = p$. By construction of $\mathcal{M}(H)$, $\langle s, p, o\rangle$ is a triple of H. The interpolation lemma asserts that $\langle \alpha(s), p, \alpha(o)\rangle$ is a triple of G. By construction of $\mathcal{M}(G)$, it contains an arc a' such that $\gamma(a') = \langle m(\alpha(s)), m(\alpha(o))\rangle$ and $\epsilon(a') = p$. We have $\epsilon(a') = p \leq_1 \epsilon(a) = p$ and, by definition of π, $\gamma(a') = \langle \pi(m(s)), \pi(m(o))\rangle$. □

($\Leftarrow$) Suppose a $\leq_1$-M-morphism from $\mathcal{M}(H)$ into $\mathcal{M}(G)$. Let us consider the mapping $\alpha : \mathcal{V}(H) \to V$ defined as follows: for every node $m(b)$ in $\mathcal{M}(H)$ labelled by a blank, $\alpha(b) = \epsilon(\pi(m(b)))$, for every node $m(x) \in M(H)$ labelled by an uriref or a literal, $\alpha(x) = x$. The mapping α is an instance mapping. It remains to prove that $H_\alpha \subseteq G$. Let us consider an arbitrary triple $\langle \alpha(s), p, \alpha(o)\rangle \in H_\alpha$. We have to prove that this triple is an element of G. By construction of $\mathcal{M}(H)$, there exists an arc a of $\mathcal{M}(H)$ such that $\gamma(a) = \langle m(\alpha(s)), m(\alpha(o))\rangle$ and $\epsilon(a) = p$. By definition of an homomorphism, there exists an arc $a' \in \mathcal{M}(G)$ with $\gamma(a') = \langle \pi(m(\alpha(s))), \pi(m(\alpha(o)))\rangle$ and $\epsilon(a') \leq_1 \epsilon(a)$. Since it is an uriref, $\epsilon(a) = \epsilon(a') = p$. See that for every entity $e \in H$, $\pi(m(\alpha(e))) = m(\alpha(e))$. If e is a uriref or a literal, $\alpha(e) = e$, and we have to prove that $\pi(m(e)) = m(e)$. It is true because there is a unique node in $\mathcal{M}(G)$ labelled by e, and the node labelled by e in $\mathcal{M}(H)$ must be mapped into it.[2] If e is a blank, then $\alpha(e) = \epsilon(\pi(m(e)))$ (by definition of α). Then $m(\alpha(e)) = m(\epsilon(\pi(m(e)))) = \pi(m(e))$. It follows that $\gamma(a') = \langle m(\alpha(s)), m(\alpha(o))\rangle$ and finally, that the triple $\langle \alpha(s), p, \alpha(o)\rangle$ (used to obtain a') is in G. □

[2] The reader familiar with conceptual graphs will recognize here the requirement for a normality condition.

A first interest of this reformulation is in a representational point of view: in FIG. 3, not only data is graphically represented, but also inferences (the drawing of the morphism that caracterizes entailment). Experiences in the CG community (*e.g.* [8]) show that these "graphical inferences" are very easy to understand for non-specialists in logics or computer science. We will also show (in Sect. 5) that this reformulation offer great benefits for computational purposes.

4 RDF Triplesets as Directed, Labelled Hypergraphs

But before that, we will focus on another encoding of RDF triplesets (as hypergraphs). A different representation of RDF triplesets (as bipartite graphs) has been proposed in [9]. Its main advantage is its proximity to the tripleset's semantics. Here we use this representation (these bipartite graphs are the incidence bipartites associated with our hypergraphs, so they can be considered as the same mathematical objects) also for a reasoning purpose. Indeed we use a transformation of RDF triplesets into hypergraphs (as in [9]) as well as a transformation of interpretations into the same hypergraphs.

A first result (Lemma 2) shows that an interpretation I is a model for a tripleset G iff there is a morphism from the hypergraph associated with G into the one associated with H. The immediate interests are twofold: it provides us with a clear graphical representation of interpretations (extending the representation in [9] to interpretations), and, in the same way as in the previous section, it is a graphical representation of the proof that an interpretation is a model of a tripleset.

The same morphism, this time between two graphs associated with triplesets, is used to caracterize simple RDF entailment (Theorem 3). It shows how the graphs in [9] can be used for reasonings. Finally, we show the equivalence between this caracterization and the one used in Theorem 2, effectively providing another proof for the Interpolation Lemma.

Let us now discuss about the proof of Theorem 3 itself. It is grounded on a very simple framework. Let us consider a logic $\mathcal{L}$ (here simple RDF). Let us consider a set $\mathcal{E}$ (here the hypergraphs), and a transitive relation $\sqsubseteq$ (here the existence of a morphism) on $\mathcal{E}$. Let us now introduce a transformation $\mathcal{H}$ associating an element of $\mathcal{E}$ to each formula and each interpretation of $\mathcal{L}$. This transformation must satisfy the following criteria:

1. i is a model of f if and only if $\mathcal{H}(f) \sqsubseteq \mathcal{H}(i)$;
2. for every satisfiable formula f of $\mathcal{L}$, there exists a model i of f such that $\mathcal{H}(i) = \mathcal{H}(f)$.

These two criteria are then sufficient to prove that $\sqsubseteq$ is sound and complete w.r.t. entailment of $\mathcal{L}$, *i.e.* that $f \models f'$ iff $f' \sqsubseteq f$. Lemma 2 expresses the first condition, and lemma 3 the second. Theorem 3 reformulates this soundness and completeness result in the case of RDF triplesets.

Note that this framework has been successfully applied for conceptual graphs [10], and remains valid for many extensions of simple RDF: we show in the next

section how it can be extended to RDF/RDFS, but it could also be used for the extensions presented in [2]. A Master's thesis is actually devoted in our team, using this framework, to extend RDF entailment to path queries.

4.1 Preliminary Definitions

Definition 8 (Directed, Labelled Hypergraph). *A* directed labelled hypergraph *(or* H-graph*) over a set of terms V is a triple $G = \langle N, H, \epsilon \rangle$ where N is a finite set of* nodes*, $H \subseteq N^+$ is a finite set of* hyperarcs*, and $\epsilon : N \cup H \to V$ maps each node and hyperarc to an element of the set of terms.*

An H-graph can be represented as follows: a node is represented by a rectangle in which we write its label. An hyperarc $\langle x-1, ..., x_p \rangle$ is represented by an oval in which we write its label. For $1 \leq i \leq p$, we draw a line between the oval and the rectangle associated with the node x_i, and write the number i next to this line to indicate the ordering of this tuple. We have chosen this representation by analogy with conceptual graphs (indeed, the CG semantically equivalent to a tripleset has the same representation as this hypergraph).

We must now update our morphisms to this new structure. The following lemma handles the required transitivity of the binary relation associated with the existence of a morphism.

Definition 9 (Directed, Labelled Hypergraph Homomorphism). *Let $G = \langle N, H, \epsilon \rangle$ and $G' = \langle N', H', \epsilon' \rangle$ be two H-graphs over a set of terms V. Let $\leq$ be a preorder over V. A* directed, labelled hypergraph homomorphism *according to $\leq$ (or $\leq$-*H-morphism*) from G into G' is a mapping $\pi : N \to N'$ that preserves the preorder on labels as well as incidence of hyperarcs, i.e.:*

1. *for each $n \in N$, $\epsilon'(\pi(n)) \leq \epsilon(n)$;*
2. *for each $h = \langle n_1, \ldots, n_k \rangle \in H$, $\exists a' = \langle \pi(n_1), \ldots, \pi(n_k) \rangle \in H'$ such that $\epsilon'(a') \leq \epsilon(a)$.*

Lemma 1 (Composition). *The composition of two $\leq$-H-morphisms is a $\leq$-H-morphism.*

Proof. Let G_1, G_2, G_3 be three H-graphs over a set of terms V. Let $\leq$ be a preorder on V. Let π_1 (resp. π_2) be a $\leq$-H-morphism according to $\leq$ from G_1 into G_2 (resp. from G_2 into G_3). We prove that $\pi_2 \circ \pi_1$ is a $\leq$-H-morphism from G_1 into G_3.

1. Let n be a node of G_1. We have $\epsilon(\pi_1(n)) \leq \epsilon(n)$ and $\epsilon(\pi_2(\pi_1(n))) \leq \epsilon(\pi_1(n))$ (def. of H-morphism). Since a preorder is transitive, $\epsilon(\pi_2(\pi_1(n))) \leq \epsilon(n)$.
2. Let $h_1 = \langle n_1, \ldots, n_p$ be an hyperarc of G_1. Then there exists an hyperarc $h_2 = \langle \pi_1(n_1), \ldots, \pi_1(n_p) \rangle$ of G_2 with $\epsilon(h_2) \leq \epsilon(h_1)$ (def. of H-morphism). Similarly, there exists an hyperarc $h_3 = \langle \pi_2(\pi_1(n_1)), \ldots, \pi_2(\pi_1(n_p)) \rangle$ with $\epsilon(h_3) \leq \epsilon(h_2)$. We also conclude thanks to the transitivity of $\leq$. □

4.2 Hypergraph Representation of a Simple Interpretation

Let $I = \langle I_R, I_P, \iota_{ext}, \iota_s, \iota_l \rangle$ be an interpretation of a set of terms V. We associate to this interpretation an H-graph $\mathcal{H}(I) = \langle N, H, \epsilon \rangle$ built as follows:

1. To each resource $r \in I_R \cup I_P$ we associate a distinct node $h(r)$. Then $N = \{h(r) \mid r \in I_R \cup I_P\}$. Each of these nodes will be labelled by a subset of V. Intuitively, $\epsilon(h(x)) = \{v_1, \ldots, v_k\}$ means that $v_1, \ldots, v_k$ are all the terms of V interpreted by the resource or property x in I. Let us now formally build this labelling: each node is initially labelled by the emptyset $\{\}$. Then for each element x of V:
   - if x is a plain literal in $\mathcal{L}_P(V)$, $\epsilon(h(x)) = \epsilon(h(x)) \cup \{x\}$;
   - if x is a typed literal in $\mathcal{L}_T(V)$, $\epsilon(h(\iota_l(x))) = \epsilon(h(\iota_l(x))) \cup \{x\}$;
   - if x is an uriref in $\mathcal{U}(V)$, $\epsilon(h(\iota_s(x))) = \epsilon(h(\iota_s(x))) \cup \{x\}$;
   - otherwise, if x is a blank in $\mathcal{B}(V)$, do nothing.
2. For each element $p \in I_P$, for each pair $\langle x, y \rangle \in \iota_{ext}(p)$, there exists an hyperarc $\langle h(x), h(p), h(y) \rangle$ in H labelled by $\{iext\}$.

Example 10. Fig. 4 shows the representation of the H-graph associated with the interpretation of Ex. 2. This representation is simpler than the usual one (Fig. 1), and highlights the structure of the interpretation. However we have lost information on the set I_P, since a node that is not the second argument of an hyperarc may belong to I_P or not (though this information is never needed).

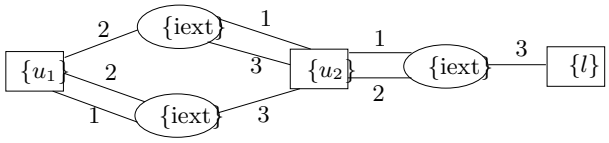

Fig. 4. The H-graph associated with an interpretation.

4.3 Hypergraph Representation of an RDF Tripleset

Let G be an RDF tripleset. The directed, labelled hypergraph $\mathcal{H}(G) = \langle N, H, \epsilon \rangle$ associated with G is built as follows:

1. To each element $e \in \mathcal{V}(G)$ we associate a distinct node $h(e)$. Then $N = \{h(e) \mid e \in \mathcal{V}(G)\}$. As for the hypergraph associated with an interpretation, each node h(e) is labelled by a set. This set is the emptyset if e is a blank and the singleton $\{e\}$ otherwise.
2. To each triple $t = \langle s, p, o \rangle \in G$ we associate a distinct hyperarc $h(t) = \langle h(s), h(p), h(o) \rangle$. Then $H = \{\langle h(s), h(p), h(o) \rangle \mid \langle s, p, o \rangle \in G\}$. The label of the arc $h(t)$ is $\{iext\}$.

Example 11. Fig. 5 shows the H-graph associated with the RDF tripleset H of Ex. 1.

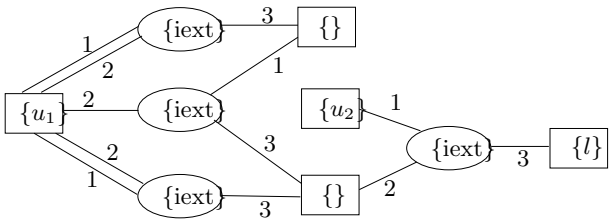

Fig. 5. The H-graph associated with an RDF tripleset.

4.4 Simple Entailment as an Hypergraph Homomorphism

Lemma 2. *Let G be an RDF tripleset, and V be a set of terms that contains the set of terms of G. Let $\leq_2$ be the partial order defined by $e \leq_2 e' \Leftrightarrow e' \subseteq e$. An interpretation I of V is a model of G iff there exists a $\leq_2$-H-morphism from $\mathcal{H}(G)$ into $\mathcal{H}(I)$.*

Proof. We successively prove both directions of the equivalence.

($\Rightarrow$) Let us consider a model $I = \langle I_R, I_P, \iota_{ext}, \iota_s, \iota_l \rangle$ of G. We have to show that there exists a $\leq_2$-H-morphism from $\mathcal{H}(G)$ into $\mathcal{H}(I)$. Since I is a model of G, there exists a mapping $\iota : \mathcal{V}(G) \rightarrow I_R \cup I_P$ that respects the 5 conditions listed in Def. 3. We build the mapping π from the nodes of $\mathcal{H}(G)$ into the nodes of $\mathcal{H}(I)$ as follows: if $h(x)$ is a node of G, then $\pi(h(x)) = h(\iota(x))$. It remains to show that π is a $\leq_2$-H-morphism.

1. Let $h(x)$ be a node of x. Let us show that $\epsilon(\pi(h(x))) \leq_2 \epsilon(h(x))$, *i.e.* $\epsilon(h(x)) \subseteq \epsilon(\pi(h(x)))$.
   - if x is a blank, then by construction $\epsilon(h(x)) = \emptyset$ and is thus a subset of any other set;
   - otherwise, by construction, $\epsilon(h(x)) = \{x\}$. It remains only to check that x is an element of $\epsilon(\pi(h(x))) = \epsilon(h(\iota(x)))$. If x is a plain literal, then $\iota(x) = x$ and the label of $h(\iota(x))$ contains x. If x is an uriref (resp. a typed literal), $\iota(x) = \iota_s(x)$ (resp. $\iota_l(x)$). Then the label of $h(\iota(x))$ also contains x.
2. Let us now prove that for every hyperarc $a = \langle h(s), h(p), h(o) \rangle \in \mathcal{H}(G)$, there exists an hyperarc $a' = \langle \pi(h(s)), \pi(h(p)), \pi(h(o)) \rangle \in \mathcal{H}(I)$. If such an hyperarc exists, it will be easy to check that $\epsilon(a') \leq_2 \epsilon(a)$: all hyperarcs are labelled by $\{iext\}$. Since $a = \langle h(s), h(p), h(o) \rangle \in \mathcal{H}(G)$, then by construction there must be a triple $\langle s, p, o \rangle$ in G. Thus (Def. 3) $\langle \iota(s), \iota(o) \rangle \in \iota_{ext}(\iota(p))$. By construction of $\mathcal{H}(I)$, it contains an hyperarc $\langle h(\iota(s)), h(\iota(p)), h(\iota(o)) \rangle$. And by construction of π, this hyperarc is exactly $\langle \pi(h(s)), \pi(h(p)), \pi(h(o)) \rangle$. □

($\Leftarrow$) Now let us consider a $\leq_2$-H-morphism π from $\mathcal{H}(G)$ into $\mathcal{H}(I)$. We build a mapping $\iota : \mathcal{V}(G) \rightarrow I_R \cup I_P$ as follows: for each node $h(x)$ of $\mathcal{H}(G)$, consider the node $h(y)$ of $\mathcal{H}(I)$ such that $h(y) = \pi(h(x))$. Then $\iota(x) = y$. It remains now to prove that ι satisfies the 5 conditions of Def. 3. For each node $h(x)$ of $\mathcal{H}(G)$, we consider the node $h(y)$ of $\mathcal{H}(I)$ such that $h(y) = \pi(h(x))$.

- If x is a plain literal, we must prove that $\iota(x) = x$. By construction, $\iota(x) = y$. We know that $h(x)$ is labelled by $\{x\}$. Since π maps $h(x)$ into $h(y)$, $x \in \epsilon(h(y))$ and thus, by construction of $\mathcal{H}(I)$, $x = y = \iota(x)$.

- If x is a typed literal, we must prove that $\iota(x) = \iota_l(x)$. By construction, $\iota(x) = y$. As before, $x \in \epsilon(h(y))$. By construction of $\mathcal{H}(I)$, $\iota_l(x) = y = \iota(x)$.
- If x is an uriref, proceed as for typed literals (replacing ι_l with ι_s).
- If x is a blank, then x is the subject or object of a triple of G. Thus $h(x)$ is the first or third argument of an hyperarc of $\mathcal{H}(G)$. Since π is a H-morphism, $\pi(h(x) = h(y)$ is the first or third argument of an hyperarc of $\mathcal{H}(I)$. By definition of ι_{ext}, $y = \iota(x) \in I_R$.
- Let us now prove that, for every triple $\langle s, p, o\rangle \in G$, $\langle \iota(s), \iota(o)\rangle \in \iota_{ext}(\iota(p))$. If $\langle x_s, x_p, x_o\rangle \in G$, then, by construction of $\mathcal{H}(G)$, there exists an hyperarc $\langle h(x_s), h(x_p), h(x_o)\rangle$ of $\mathcal{H}(G)$. Since π is a H-morphism, the hyperarc $\langle \pi(h(x_s)), \pi(h(x_p)), \pi(h(x_o))\rangle = \langle h(y_s), h(y_p), h(y_o)\rangle$ is an hyperarc of $\mathcal{H}(I)$. By construction of $\mathcal{H}(I)$, $\langle y_s, y_o\rangle \in \iota_{ext}(y_p)$, *i.e.*: $\langle \iota(x_s), \iota(x_o)\rangle \in \iota_{ext}(\iota(x_p))$.

□

Lemma 3 (Isomorphic Interpretation). *Let G be an RDF tripleset. Then there exists an interpretation I of G such that $\mathcal{H}(I) = \mathcal{H}(G)$.*

Proof. By "reverse engineering" the transformation $\mathcal{H}$, we can build from $\mathcal{H}(G)$ an interpretation I such that $\mathcal{H}(I) = \mathcal{H}(G)$. To each node in $\mathcal{H}(G)$ we associate a resource of I_R (note that we impose $I_P \subseteq I_R$). For each node x, for each term $e \in \epsilon(x)$, we impose the term e to be interpreted (via ι_s or ι_l) by the resource associated with x. Finally, for each hyperarc $\langle s, p, o\rangle$ in $\mathcal{H}(G)$, we add the pair of resources associated with s and o to the extension of the resource asssociated with p. It is immediate to chack that, by applying $\mathcal{H}$ to that interpretation, we obtain the H-graph $\mathcal{H}(G)$ (or more precisely, the H-graph isomorphic to it).

Corollary 1. *Each RDF tripleset is satisfiable.*

Proof. Since there always exists a H-morphism from a graph into itself, we conclude thanks to Lem. 2 that the isomorphic interpretation of any RDF tripleset G is a model of G. Thus G is satisfiable.

Theorem 3. *Let G and H be two RDF triplesets defined over a set of terms V. Then $G \models H$ if and only if there is a $\leq_2$-H-morphism from $\mathcal{H}(H)$ into $\mathcal{H}(G)$.*

Proof. We prove both directions of the equivalence.

($\Rightarrow$) Let us suppose that $G \models H$. It means that every model of G is also a model of H. In particular, the isomorphic interpretation I of G (see Lem. 3), being a model of G, is also a model of H. Thanks to Lem. 2, it means that there exists a $\leq_2$-H-morphism from $\mathcal{H}(H)$ into $\mathcal{H}(I) = \mathcal{H}(G)$. □

($\Leftarrow$) Let us suppose that there exists a $\leq_2$-H-morphism π from $\mathcal{H}(H)$ into $\mathcal{H}(G)$. We have to prove that every model of G is also a model of H. Let us consider an arbitrary model M of G. Thanks to Lem. 2, there exists a $\leq_2$-H-morphism π' from $\mathcal{H}(G)$ into $\mathcal{H}(M)$. We use Lem. 1 to show that $\pi' \circ \pi$ is a $\leq_2$-H-morphism from $\mathcal{H}(H)$ into $H(G)$. Finally, we conclude (Lem. 2) that M is also a model of H. □

4.5 Relationships with Multigraphs

This section finishes with this last theorem, asserting the equivalence of M-morphisms and H-morphisms for RDF simple entailment. Since the proof is

immediate, it is left out. It means that we can use indifferently M-graphs or H-graphs for computing entailments, or for checking if an interpretation is a model for a tripleset. It is also the final step providing another proof for the interpolation lemma.

Theorem 4. *Let G and H be two RDF triplesets defined over a set of terms V. Then there is a directed, labelled hypergraph homomorphism from $\mathcal{H}(H)$ into $\mathcal{H}(G)$ according to $\leq_2$ if and only if there is a directed, labelled multigraph homomorphism from $\mathcal{M}(H)$ into $\mathcal{M}(G)$ according to $\leq_1$.*

5 Complexity and Algorithms

It is now well known that SIMPLE RDF ENTAILMENT (deciding whether or not an RDF tripleset simply entails another one) is a NP-complete problem. It has been proven via the equivalence with conceptual graphs [6, 7] or via a reduction to graph colouring [2]. Thus checking if an interpretation is a model for an RDF tripleset is also an NP-complete problem (we have shown here that they were the same problem). The latter author also provides us with a polynomial case for SIMPLE RDF ENTAILMENT: when there is no blank node in the entailee H.

We present here links and guidelines allowing to quickly translate results obtained in other knowledge representation communities (namely conceptual graphs and constraint programming), thanks to our reformulation of entailment as a graph homomorphism.

5.1 Constraint Networks and Polynomial Cases

The relationships between homomorphisms, conceptual graphs projection and constraint satisfaction problems allow to obtain much more interesting polynomial cases. Let us consider here the following equivalences:

1. the RDF tripleset G simply entails the RDF tripleset H
2. there is a $\leq_1$-M-morphism from $\mathcal{M}(H)$ into $\mathcal{M}(G)$
3. there is projection from the conceptual graph $\mathcal{C}(\mathcal{M}(H))$ into $\mathcal{C}(\mathcal{M}(G))$
4. the constraint network $\mathcal{N}(\mathcal{C}(\mathcal{M}(H)), \mathcal{C}(\mathcal{M}(G)))$ is satisfiable.

We do not have the place here to explicit the transformations involved, though it should be done in an extended version of this paper. 1) $\equiv$ 2) is proven in this paper, 2) $\equiv$ 3) is proven in [7], and 3) $\equiv$ 4) is proven in [11]. The interesting point is that these transformations are polynomial, and that the graphs $\mathcal{M}(H)$, $\mathcal{C}(\mathcal{M}(H))$ and $\mathcal{N}(\mathcal{C}(\mathcal{M}(H)), \mathcal{C}(\mathcal{M}(G)))$ have exactly the *same structure*. So every polynomial case based upon the structure of a constraint network or upon the structure of the projected conceptual graph immediately translates into a polynomial case based upon the structure of the entailee in simple RDF.

Both conceptual graphs projection [12] and constraint network satisfiability [13] have been proven polynomial when the graphs are trees. It follows naturally that SIMPLE RDF ENTAILMENT is polynomial when the entailee M-graph is a

tree. A lot of work has been produced in the constraint satisfaction community to generalize this result: more general cases (using hypertree decompositions) are listed in [14], all can be directly translated to SIMPLE RDF ENTAILMENT.

5.2 Algorithms

Since the Backtrack algorithm used to solve constraint satisfaction problems rely on the structure of the associated graph, the same algorithm optimizations can be used for the SIMPLE RDF ENTAILMENT. Some of these optimizations have been selected in [10] (in the conceptual graph formalism). The main point is that these optimizations do not require any overhead cost. These algorithms are considered as very efficient outside the phase transition.

6 Conclusion and Perspectives

We have presented here a reformulation of simple RDF entailment as a graph homomorphism. The standalone proof of soundness and correctness is used as a new proof of the interpolation lemma.This proof can be used as a framework to study reasoning engines for extensions of simple RDF. Though we have shown that a benefit of our reformulation was to offer the end-user with a graphical illustration of reasonings, our main interest resides in using the graph structure for an optimization purpose. The links we establish between RDF entailment, graph homomorphism, conceptual graphs projection and constraint satisfaction problems are an important step in that direction.

However, RDF is a language developped for the web. And the specific problem that will be encountered is the huge size of the data. The RDF WEB ENTAILMENT problem should be presented as follows: given a RDF tripleset (a query) Q, is there a set of RDF triplesets $G_1, ..., G_2$ available on the (semantic) web such that they entail Q? Though there is no theoretical problem (we have just to compute whether the merge of $G_1, ..., G_2$ entails Q), it is doubtful that it will be possible to compute the merge of all triplesets available on the web. [15] provides us with an algorithm that remains sound and complete without merging the graphs (in conceptual graphs terms, when the target is not in nortmal form). Moreover, this algorithm is less efficient than the standard backtrack, and do not benefit effectively from the above mentioned optimizations. This example, among other, shows that, though RDF ENTAILMENT can benefit from results obtained in similar formalisms, its new feature (a language designed for the web) leads to particular problems that we should take into account.

References

[1] Hayes, P.: RDF Semantics. W3C Recommendation (2004) `http://www.w3.org/TR/2004/REC-rdf-mt-20040210/`.

[2] ter Horst, H.J.: Extending the RDFS Entailment Lemma. In: Proceedings of the Third International Semantic Web Conference, ISWC'04. Volume 3298 of LNCS., Springer (2004) 77–91

[3] Hahn, G., Tardif, C.: Graph homomorphisms: structure and symmetry. In: Graph Symmetry. Number 497 in NATO Adv. Sci. Inst. Ser. C. Math. Phys. Sci. (1997) 107–166
[4] Chein, M., Mugnier, M.L.: Conceptual Graphs: fundamental notions. Revue d'Intelligence Artificielle **6** (1992) 365–406
[5] Montanari, U.: Networks of Constraints: Fundamental Notions and Application to Picture Processing. Information Sciences **7** (1974) 95–132
[6] Corby, O., Dieng, R., Hebert, C.: A Conceptual Graph Model for W3C Resource Description Framework. In: International Conference on Conceptual Structures. (2000) 468–482
[7] Baget, J.F.: Homomorphismes d'hypergraphes pour la subsomption en RDF/RDFS. In: 10e conférence sur langages et modèles à objets (LMO). Volume 10. (2004) 203–216
[8] Genest, D.: Extensions du modèle des graphes conceptuels pour la recherche d'informations. PhD thesis, Université de Montpellier II (2000)
[9] Hayes, J., Guttiérrez, C.: Bipartite Graphs as Intermediate Model for RDF. In: Proceedings of the Third International Semantic Web Conference, ISWC'04. Volume 3298 of LNCS., Springer (2004) 47–61
[10] Baget, J.F.: Simple conceptual graphs revisited: hypergraphs and conjunctive types for efficient projection algorithms. In: 11th international conference on conceptual sructures (ICCS). Number 2870 in LNCS, Springer (2003) 195–208
[11] Mugnier, M.L.: Knowledge Representation and Reasonings Based on Graph Homomorphism. In: 8th International Conference on Conceptual Structures (ICCS'00). Volume 1867 of LNCS., Springer (2000) 172–192
[12] Mugnier, M.L., Chein, M.: Polynomial algorithms for projection and matching. In: Selected Papers from AWCG'92. Volume 754 of LNAI., Springer (1993)
[13] Freuder, E.: A sufficient condition for backtrack-free search. Journal of the ACM **29** (1982) 24–32
[14] Gottlob, G., Leone, N., Scarcello, F.: A comparison of structural decomposition methods. (1999)
[15] Guinaldo, O., Haemmerlé, O.: Knowledge querying in the conceptual graphs model: the RAP module. In: Proc. of ICCS. 98. LNCS, (Springer) 287–294