RDF as Graph-Based, Diagrammatic Logic

Frithjof Dau

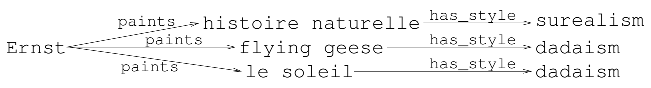
Dept. of Mathematics, Dresden Technical University, Germany

Abstract. The Resource Description Framework (RDF) is the basic standard for representing information in the Semantic Web. It is mainly designed to be machine-readable and -processable. This paper takes the opposite side of view: RDF is investigated as a logic system designed for the needs of humans. RDF is developed as a logic system based on mathematical graphs, i.e., as *diagrammatic reasoning system*. As such, is has humanly-readable, diagrammatic representations. Moreover, a sound and complete calculus is provided. Its rules are suited to act on the diagrammatic representations. Finally, some normalforms for the graphs are introduced, and the calculus is modified to suit them.

1 Introduction: RDF and Graphs

The Resource Description Framework (RDF) is the basic standard for representing information in the Semantic Web. The information is encoded in statements, which are subject-predicate-object triples of information. RDF can be seen from two perspectives: computers and humans. For computers, it is important that RDF is a machine processable data-format. Thus, much effort is spent on specifications which suit this purpose (e.g. RDF/XML). An approach to make RDF accessible for humans can already be found in the RDF-primer [8], where the set of triples is converted into a mathematical graph: The subjects and objects of all statements are mapped to vertices, and each statement is mapped to a directed edge between its subject and object, labeled by its predicate. Consider the following set of triples $\mathcal{T}_1$ and its representation as graph:

```
(Ernst,paints,hist. naturelle),(Ernst,paints,flying geese),
(Ernst,paints,le soleil),(hist. naturelle,has style,surrealism),
(flying geese,has style,dadaism),(le soleil,has style,dadaism)
```



The translation of triple set to these graphs is appealing, but is has two problems.

RDF allows a statement predicate to appear as the subject of another statement. E.g., {`(type type prop)`, `(subject type prop)`} is a well-formed set of triples, but it cannot be transformed to a graph with the conventions of [8]. The first goal of the paper is closes the gap that not all triple sets can be represented as graphs. For this, we adopt the approach of [1, 2] instead: RDF triple sets are translated to hyper graphs, where subjects, objects, *and predicates* are mapped to vertices, and each statement (s, p, o) is mapped to an *3-ary hyperedge* linking these vertices.

RDF is used for information-*processing*. In [3] an entailment relation between triple sets is introduced. But this relation does not the diagrammatic representation of RDF. It is well accepted that diagrams are often easier to comprehend than symbolic notations ([7, 6]). The second goal of the paper is, after adopting the semantics of [3], to provide an adequate *diagrammatic* calculus.

In this understanding, RDF is developped as diagrammatic reasoning system (DRS), in a humanly readable and mathematically precise manner. In the next sections, an overview of the syntax, semantics, a sound and complete calculus, and some normalforms of RDF Graphs for RDF as DRS is provided. Due to space limitations, some formal definitions an all proofs are omitted. They can be found in a technical report on `www.dr-dau.net`.

2 Syntax

In this section we define the syntax of the system, starting with the vocabulary.

Definition 1. *Let* $\mathsf{Blanks} := \{\_1, \_2, \_3, \ldots, \}$ *be a set of so-called* BLANKS. *A triple* $\mathsf{Voc} := (\mathsf{URIs}, \mathsf{TypedLit}, \mathsf{PlainLit})$ *is called* VOCABULARY. *The elements of these sets are called* UNIVERSAL RESOURCE IDENTIFICATORS, TYPED LITERALS, *and* PLAIN LITERALS, *resp. We set* $\mathsf{Lit} := \mathsf{TypedLit} \cup \mathsf{PlainLit}$. *The elements of* Lit *are called* LITERALS. *The elements of* $\mathsf{URIs} \cup \mathsf{TypedLit} \cup \mathsf{PlainLit}$ *are called* NAMES. *We assume that* Blanks, URIs, $\mathsf{TypedLit}$, $\mathsf{PlainLit}$ *are pairwise disjoint.*

Next we define the well-known triple notation of RDF as well as the corresponding mathematical graphs. To avoid confusion, we use the term *RDF triple set* instead of the common term *RDF graph* for the triple notation.

Definition 2. *An* RDF TRIPLE SET OVER Voc *is a set of triples* (s, p, o) *with* $s \in \mathsf{URIs} \cup \mathsf{Blanks}$, $p \in \mathsf{URIs}$, $o \in \mathsf{URIs} \cup \mathsf{Blanks} \cup \mathsf{Lit}$. *We call* s *the* SUBJECT, p *the* PROPERTY *and* o *the* OBJECT *of the triple.*

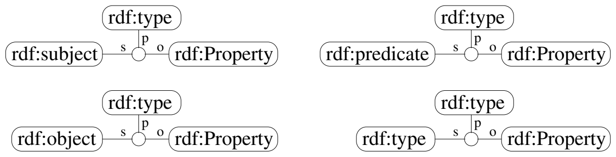
A structure (V, E, ν, κ) *is an* RDF GRAPH OVER Voc *iff* V *and* E *are finite sets of* VERTICES *and* EDGES, $\nu : E \to V^3$ *is a mapping such that each vertex is incident with as least one edge, and* $\kappa : V \to \mathsf{Voc}$ *is a mapping such that for each* $e \in E$, *we have* $\kappa(e\big|_1) \in (\mathsf{URIs} \cup \mathsf{Blanks})$, $\kappa(e\big|_2) \in \mathsf{URIs}$, *and* $\kappa(e\big|_3) \in (\mathsf{URIs} \cup \mathsf{Blanks} \cup \mathsf{Lit})$. *For* $e \in E$ *with* $\nu(e) = (v_1, v_2, v_3)$ *we will often write* $e = (v_1, v_2, v_3)$, *and each* v_i, $i = 1, 2, 3$ *is denoted by* $e\big|_i$. *For* $v \in V$ *we set* $E_v := \{e \in E \mid \exists i.\nu(e)\big|_i = v\}$, *and for* $e \in E$ *we set* $V_e := \{v \in V \mid \exists i.\nu(e)\big|_i = v\}$. *The set* $\{bl \in \mathsf{Blanks} \mid \exists v \in V : \kappa_b(v) = bl\}$ *is called the* SET OF BLANKS OF $\mathcal{G}$.

Compared to RDF triple sets, RDF graphs provide a richer means to express in different ways some given amount of information, and predicates which appear as the subject of other statements do not cause problems. This shall be exemplified with the first four axiomatic triples from the RDF-semantics (see [3]). They are:

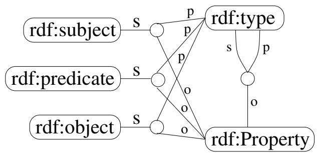
```
(rdf:type rdf:type rdf:Property)      (rdf:subject rdf:type rdf:Property)
(rdf:predicate rdf:type rdf:Property) (rdf:object rdf:type rdf:Property)
```

Note that `type` occurs both as subject and as predicate in these triples, so these four triples cannot be displayed due to the graph-drawing conventions of [5]. But with the herein defined RDF graphs, there are now different possibilities to transform this set of triples to a graph. Three of them are depicted below.

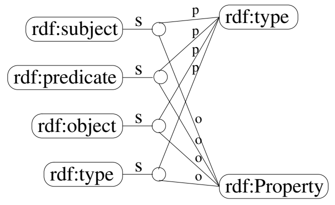
1)



2)



3)



In the first graph, each vertex is incident exactly once with an edge: It has so-to-speak a maximum number of vertices. Graphs which satisfy this conditions will be said to be in ANTI-NORMAL-FORM (ANF). For the third graph, we have an opposite situation: No different vertices are labeled the same, i.e., this graph contains the minimal number of vertices. Such graphs are said to be in NORMAL-FORM (NF). For graphs in ANF or NF, we assume moreover that they do not have REDUNDANT EDGES, i.e., for all edges $e = (v_1, v_2, v_3)$ and $f = (w_1, w_2, w_3)$ with $\kappa(v_i) = \kappa(w_i)$ for $i = 1, 2, 3$, we have $e = f$.

Let $\mathcal{T}$ be an RDF triple set over Voc. Then let $\Phi_N(\mathcal{T}) := (V, E, \nu, \kappa)$ be the following RDF graph in NF: For each name or blank x in $\mathcal{T}$, let v_x be fresh[1] vertex. Let V be the set of these vertices. Let $E := \mathcal{T}$. For $e = (s, p, o) \in E$, let $\nu(e) = (v_s, v_p, v_o)$. For $v = v_x \in V$, let $\kappa(v_x) := x$. Similarly, let $\Phi_{AN}(\mathcal{T}) := (V, E, \nu, \kappa)$ be the following RDF graph in ANF: For each edge $e = (s, p, o) \in E$, let e_s, e_p, e_o be fresh vertices. Let V be the set of these vertices, let $E := \mathcal{T}$, and for $e = (s, p, o) \in E$, let $\nu(e) := (e_s, e_p, e_o)$. For $v = e_x \in V$, we set $\kappa(v_x) := x$.

Now let $\mathcal{G} := (V, E, \nu, \kappa)$ be an RDF Graph. Let $\Psi(\mathcal{G})$ be the following RDF triple set: $\Psi(\mathcal{G}) := \{(\kappa(e|_1), \kappa(e|_2), \kappa(e|_3)) \mid e \in E\}$.

Each RDF graph in NF resp. in ANF is already been sufficiently described by the set of triples $(\kappa(e|_1), \kappa(e|_2), \kappa(e|_3))$ with $e \in E$. If $\mathcal{T}$ is an RDF triple set, we have $\mathcal{T} = \Psi(\Phi_N(\mathcal{T}))$ and $\mathcal{T} = \Psi(\Phi_{AN}(\mathcal{T}))$. Vive versa: If $\mathcal{G}$ is an RDF graph in NF, we have $\mathcal{G} = \Phi_N(\Psi(\mathcal{G}))$, and if $\mathcal{G}$ is an RDF graph in ANF, we have $\mathcal{G} = \Phi_{AN}(\Psi(\mathcal{G}))$. So the mappings Φ_N, Φ_{AN} and Ψ can be understood as translations between graphs in NF or ANF and triple sets.

A SUBGRAPH of an RDF triple set is simply a subset. A SUBGRAPH of an RDF graph (V, E, ν, κ) is an RDF Graph (V', E', ν', κ') with $V' \subseteq V$, $E' \subseteq E$, $\nu' = \nu|_{E'}$ and $\kappa' = \kappa|_{V'}$. If we have moreover $E_v \subseteq E'$ for each $v \in V'$, the subgraph is called CLOSED.

[1] A vertex or edge x is called FRESH iff it is not already an element of the set of vertices and edges we consider.

We will implicitly identify RDF Graphs if they differ only in the names of the occurring blanks. As in [3], graphs like these are called EQUIVALENT.

Finally, we have to define the JOIN and MERGE of RDF graphs. The join of RDF triple sets in defined in [3] to be the set-theoretical union. But only if the joined RDF triple sets have no blanks in common, their join corresponds to their logical conjunction. Then one speaks of the MERGE of the RDF triple sets. Analogously, if $\mathcal{G}_i := (V_i, E_i, \nu_i, \kappa_i)$ is for $i = 1, \ldots, n$ an RDF graph such that that all V_i and E_i, $i = 1, \ldots, n$, are pairwise disjoint, the JOIN OF THE GRAPHS $\mathcal{G}_i$ is defined to be the RDF graph $\mathcal{G} := (V, E, \nu, \kappa)$ with $V := \bigcup_i V_i$, $E := \bigcup_i E_i$, $\nu := \bigcup_i \nu_i$ and $\kappa := \bigcup_i \kappa_i$. If the set of blanks of the graphs $\mathcal{G}_i$ are pairwise disjoint, the join of the graphs $\mathcal{G}_i$ is also called MERGE of the graphs $\mathcal{G}_i$.

3 Semantics

In this section, the semantics for RDF graphs is defined. It is based on mathematical model theory and adopted from [3, 4] for RDF triple sets.

Definition 3. *A* MODEL $\mathcal{I}$ FOR $\mathsf{Voc} := (\mathsf{URIs}, \mathsf{TypedLit}, \mathsf{PlainLit})$ *is a structure* $(\mathsf{IR}, \mathsf{IP}, \mathsf{IEXT}, \mathsf{IS}, \mathsf{IL})$ *where* IR *is a set of* RESOURCES*, called the* DOMAIN *or* UNIVERSE *of* $\mathcal{I}$*,* IP *is a set of* PROPERTIES *of* $\mathcal{I}$*,* $\mathsf{IEXT} : \mathsf{IP} \to \mathfrak{P}(\mathsf{IR} \times \mathsf{IR})$ *is a mapping,* $\mathsf{IS} : \mathsf{URIs} \to \mathsf{IR} \cup \mathsf{IP}$ *is a mapping, and* $\mathsf{IL} : \mathsf{Lit} \to \mathsf{IR}$ *is a mapping.*

Note that the relationship between IR and IP is not specified. From the viewpoint of mathematical logic, one would assume that IR and IP are disjoint. From the RDF viewpoint, it is quite normal to assume that $\mathsf{IP} \subseteq \mathsf{IR}$ holds. Both cases and arbitrary other relations between IR and IP are allowed.

In the next definition, RDF graphs are evaluated in models. We do not assume that the graph and model are based on the same vocabulary. But if a name in a graph is not interpreted in the model, the semantics will always yield that the graph evaluates to false. Thus we will usually assume that the vocabularies of a graph and an interpretation are the same.

Definition 4. *Let* $\mathcal{G} := (V, E, \nu, \kappa)$ *be an RDF graph over* $\mathsf{Voc}_\mathcal{G}$ *and let* $\mathcal{I} := (\mathsf{IR}, \mathsf{IP}, \mathsf{IEXT}, \mathsf{IS}, \mathsf{IL})$ *be an interpretation over* $\mathsf{Voc}_\mathcal{I}$*. A function* $I : V \to \mathsf{IR}$ *is an* INTERPRETATION FUNCTION (FOR $\mathcal{G}$ IN $\mathcal{I}$)*, iff for all* $v_1, v_2 \in V$ *with* $\kappa(v_1) = \kappa(v_2)$*, we have* $I(v_1) = I(v_2)$*, and for each each name* $n \in \mathsf{URIs}_\mathcal{I} \cup \mathsf{PlainLit}_I \cup \mathsf{TypedLit}_I$ *and each vertex* $v \in V$ *with* $\kappa(v) = n$*, we have* $I(v) = (\mathsf{IL} \cup \mathsf{IR})(n)$*.*

We say the GRAPH $\mathcal{G}$ HOLDS IN THE MODEL $\mathcal{I}$ *and write* $\mathcal{I} \models \mathcal{G}$*, iff for each* $v \in V$*,* $\kappa(v)$ *is a name of* $\mathsf{Voc}_\mathcal{I}$ *or a blank, and there exists an interpretation function* $I : V \to \mathsf{IR}$ *such that for each edge* $e = (v_1, v_2, v_3) \in E$ *and* $s := \kappa(v_1)$*,* $p := \kappa(v_2)$*,* $o := \kappa(v_3)$*, we have* $(I(s), I(o)) \in \mathsf{IEXT}(I(p))$ *and* $I(p) \in \mathsf{IP}$*.*

If $\mathcal{G}_a, \mathcal{G}_b$ *are RDF graphs such that* $\mathcal{I} \models \mathcal{G}_b$ *holds for each* $\mathcal{I}$ *with* $\mathcal{I} \models \mathcal{G}_a$*, we say that* $\mathcal{G}_a$ (SEMANTICALLY) ENTAILS $\mathcal{G}_b$*, and write* $\mathcal{G}_a \models \mathcal{G}_b$*.*

4 Calculus

We provide a calculus with five rules (which should be understood to manipulate the diagrams of RDF graphs) for Gs. Let $\mathcal{G} := (V, E, \nu, \kappa)$ be given.

Erasing an edge: Let e be an edge. Then e may be erased (and each vertex v which is only incident with e has to be erased as well). That is, we construct the graph $\mathcal{G}^e := (V^e, E^e, \nu^e, \kappa^e)$ with $V^{(e)} := V \backslash \{v \in V_e \mid v \notin V_f$ for any $f \in E$ with $f \neq e\}$, $E^{(e)} := E \backslash \{e\}$, $\nu^{(e)} := \nu\big|_{E^e}$, and $\kappa^{(e)} := \kappa|E^e$.
Generalizing a label: Let $bl \in \mathsf{Blanks}$ be a blank which does not appear in $\mathcal{G}$. Let $V' \subseteq V$ be a set of vertices which are identically labeled, i.e., we have $\kappa(v_1) = \kappa(v_2)$ for all $v_1, v_2 \in V'$. Then, for each $v \in V$, $\kappa(v)$ may be replaced by $\kappa(v) := bl$. That is, we construct the graph $\mathcal{G}^g := (V, E, \nu, \kappa^g)$ with $\kappa^g(w) := \kappa(w)$ for all $w \notin V'$ and $\kappa^g(v) := bl$ for all $w \in V'$.
Merging two vertices: Let $v_1 \neq v_2$ be two vertices with $\kappa(v_1) = \kappa(v_2)$. Then v_2 may be merged into v_1 (i.e., v_2 is erased and, for every edge $e \in E$, $e\big|_i = v_2$ is replaced by $e\big|_i = v_1$). That is, we construct the graph $\mathcal{G}^m := (V^m, E, \nu^m, \kappa)$ with $V^m := V \backslash \{v_2\}$, and for all $e \in E$ and $i = 1, 2, 3$ we have $\nu^m(e)(i) := \nu(e)(i)$, if $\nu(e)(i) \neq v_2$, and $\nu^m(e)(i) := v_1$ else.
Splitting a vertex: Let v_1 be a vertex, incident with edges $e_1, \ldots, e_n$. Then we can insert a fresh vertex v_2 with $\kappa(v_2) := \kappa(v_1)$, and on $e_1, \ldots, e_n$, arbitrary occurrences of v_1 may be replaced by v_2. That is, we construct a graph $\mathcal{G}^i := (V^i, E, \nu^i, \kappa)$ with $V^m := V \dot\cup \{v_2\}$, for all $e \in E$ and $i = 1, 2, 3$ we have $\nu^m(e)(i) := \nu(e)(i)$, if $\nu(e)(i) \neq v_1$ and $\nu^m(e)(i) \in \{v_1, v_2\}$ else, and which satisfies $E_{v_1} \neq \emptyset \neq E_{v_2}$ (otherwise the structure is not well-formed).
Iterating an edge: Let e be an edge with $\nu(e) = (v_1, v_2, v_3)$. Then a fresh edge e' with $\nu(e') := (v_1, v_2, v_3)$ and $\kappa(e') := \kappa(e)$ may be inserted. That is, we construct the graph $\mathcal{G}^i := (V, E^i, \nu^i, \kappa)$ with $E^{(i)} := E \dot\cup \{e'\}$ and $\nu^i := \nu \dot\cup \{(e', (v_1, v_2, v_3))\}$.
Isomorphism: Two graphs which are isomorphic are implicitly identified.

A finite sequence $(\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n)$ such that each $\mathcal{G}_{i+1}$ is derived from $\mathcal{G}_i$ by applying one of the rules above is a PROOF. We say that $\mathcal{G}_n$ can be DERIVED FROM $\mathcal{G}_1$ and write $\mathcal{G}_a \vdash \mathcal{G}_b$. Two graphs $\mathcal{G}_a, \mathcal{G}_b$ with $\mathcal{G}_a \vdash \mathcal{G}_b$ and $\mathcal{G}_b \vdash \mathcal{G}_a$ are said to be PROVABLY EQUIVALENT. The calculus is adequate, i.e. we have:

Theorem 1. *Two graphs $\mathcal{G}_a$, $\mathcal{G}_b$ satisfy $\mathcal{G}_a \vdash \mathcal{G}_b \Leftrightarrow \mathcal{G}_a \models \mathcal{G}_b$*

5 Restricting the Graphs to Normal Forms

We have seen that triple sets directly correspond to graphs in NF (or in ANF). Thus the question arises whether we can use the results of the last sections, particularly the sound- and completeness of the calculus, for the system of RDF graphs in NF and the system of RDF graphs in ANF.

Clearly, the rules of the calculus are still sound if the system of graphs is syntactically restricted. But assume we restrict ourselves to graphs in NF: It is not allowed to split a vertex, as this yields a graph which is not in NF. Vice versa, it is not possible to apply the rule 'merging two vertices': We never find two different vertices which are identically labeled. Let us consider the following valid entailment between two RDF triple sets: $(o_1\ p\ o_2) \models (o_1\ p\ o_2)$, $(\_1\ p\ o_2)$. This entailment can directly translated to a valid entailment for graphs in NF or in ANF, using the mappings Φ_N or Φ_{AN}. But in none of the restricted systems, we can find a formal proof within the given calculus. So for each restricted system, we have to alter the calculus to obtain completeness.

The rules 'erasing an edge' and 'generalizing a label' transform RDF graphs in ANF into RDF graphs in ANF again. A third rule is obtained as follows: By iterating an edge e and then splitting each of its three incident vertices, we can obtain a copy of e with *fresh* vertices. The combination of these rules is called *iterating and isolating the edge* e. This rule transforms RDF graphs in ANF into RDF graphs in ANF, too. We will write $\mathcal{G}_a \vdash_{an} \mathcal{G}_b$, if $\mathcal{G}_b$ can be derived from $\mathcal{G}_a$ with these three rules.

The rules 'erasing an edge' and 'generalizing a label' can be used for RDF graphs in NF as well. If a graph $\mathcal{G}_b$ is derived from $\mathcal{G}_a$ by firstly splitting a vertex v_1 into v_1 and a fresh vertex v_2, and then by generalizing the label of v_2, then we say that $\mathcal{G}_b$ is derived from $\mathcal{G}_a$ by SPLITTING AND GENERALIZING A VERTEX. These three rules transform RDF graphs in NF into RDF graphs in NF. We will write $\mathcal{G}_a \vdash_n \mathcal{G}_b$, if $\mathcal{G}_b$ can be derived from $\mathcal{G}_a$ with these three rules.

Theorem 2. *Let $\mathcal{G}_a$ and $\mathcal{G}_b$ be two RDF graphs with $\mathcal{G}_a \models \mathcal{G}_b$. If both graphs are in NF, we have $\mathcal{G}_a \vdash_n \mathcal{G}_b$. If both graphs are in ANF, we have $\mathcal{G}_a \vdash_{an} \mathcal{G}_b$.*

6 Outlook

This paper is only a first step for developing the underlying logic of the Semantic Web as mathematical DRS. The expressivity of RDF is rather weak. So, of course, it has to be investigated how well-known extensions of RDF can be developed as DRS. A paper which investigates how Description Logics, which are the underying logics of the state-of-the-art SW-language OWL, is in preparation.

References

[1] J. Hayes: A Graph Model for RDF. Diploma thesis, Technische Universität Darmstadt, Department of Computer Science, Germany, `http://purl.org/net/jhayes/rdfgraphmodel.html` (2004)

[2] J. Hayes, C. Gutierrez: *Bipartite Graphs as Intermediate Model for RDF.* In: Proceedings of ISWC 2004, LNCS 3298, Springer, 2004.

[3] P. Hayes: *RDF Semantics: W3C Recommendation.* 10 February 2004. `http://www.w3.org/TR/rdf-mt/`

[4] P. Hayes, C. Menzel: *A Semantics for the Knowledge Interchange Format.* Proceedings of 2001 Workshop on the IEEE Standard Upper Ontology, August 2001. `http://reliant.teknowledge.com/IJCAI01/HayesMenzel-SKIF-IJCAI2001.pdf`

[5] G. Klyne, J.J. Carroll: *Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation.* `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`

[6] R. Kremer: *Visual Languages for Konwledge Representation.* Proc. of (KAW'98) Banff, Morgan Kaufmann, 1998.

[7] J. H. Larkin H. A. Simon: *Why a diagram is (sometimes) worth ten thousand words.* Cognitive Science, 11, 65-99.

[8] F. Manola, E. Miller: *RDF Primer.* `http://www.w3.org/TR/rdf-primer/`