# Bipartite Graphs as Intermediate Model for RDF[⋆]

Jonathan Hayes[1,2] and Claudio Gutierrez[1]

[1] Center for Web Research, Dept. of Computer Science, Universidad de Chile
[2] Dept. of Computer Science, Technische Universität Darmstadt, Germany
`jonathan.hayes@gmx.de` `cgutierr@dcc.uchile.cl`

**Abstract.** RDF Graphs are sets of assertions in the form of subject-predicate-object triples of information resources. Although for simple examples they can be understood intuitively as directed labeled graphs, this representation does not scale well for more complex cases, particularly regarding the central notion of connectivity of resources.
We argue in this paper that there is need for an intermediate representation of RDF to enable the application of well-established methods from graph theory. We introduce the concept of *RDF Bipartite Graph* and show its advantages as intermediate model between the abstract triple syntax and data structures used by applications. In the light of this model we explore the issues of transformation costs, data/schema-structure, and the notion of RDF connectivity.

**Keywords:** RDF Model, RDF Graph, RDF Databases, Bipartite Graph

## 1 Introduction

The World Wide Web was originally built for human consumption, and although everything on it is machine-readable, the data is not machine-understandable [1]. The Resource Description Framework, RDF [2], is a language to express meta-data about information resources on the Web proposed by the WWW Consortium (W3C). It is intended that this information is suitable for processing by applications and thus is the foundation of the Semantic Web [3]. RDF statements are triples consisting of a subject (the resource being described), a predicate (the property), and an object (the property value). A set of RDF triples is called an *RDF Graph*, a term formally introduced by the RDF specification [4] and motivated by the underlying graph data model.

The graph-like nature of RDF is indeed intuitively appealing, but a naive formalization of this notion presents problems. Currently, the RDF specification does not distinguish clearly among the term "RDF Graph", the mathematical concept of graph, and the graph-like visualization of RDF data. An RDF Graph

[⋆] The first author was supported by Millennium Nucleus, Center for Web Research (P01-029-F), Mideplan. The second author was supported by FONDECYT No 1030810.

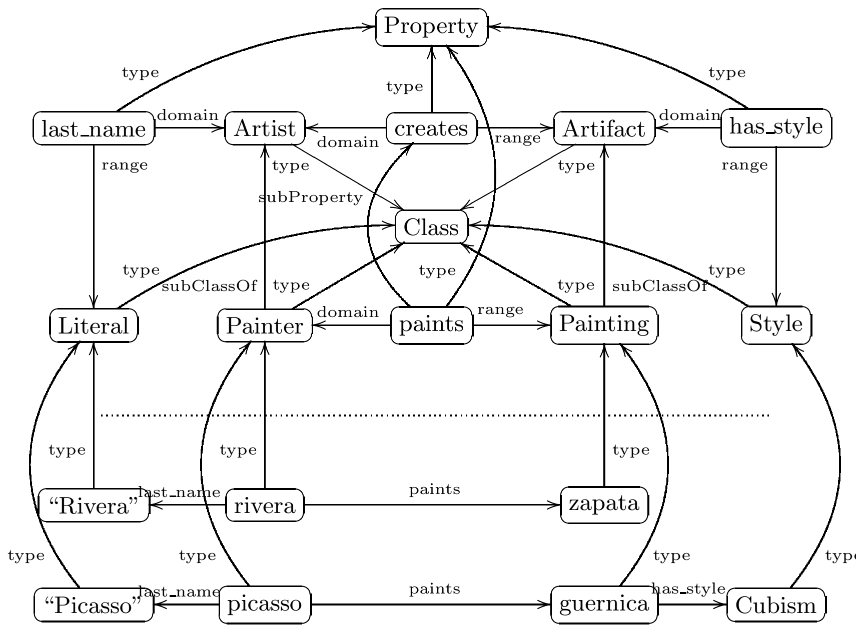

**Fig. 1.** The museum example. A non-standard graph where edge labels and nodes can represent the same object. For example, *paints* occurs as a node as well as arbitrarily often in the role of edge labels.

is a set of triples and therefore, by itself, not a graph in the classic sense. *RDF Concepts and Abstract Syntax* [4] presents "node and directed-arc diagrams" (or, as referred to in [1], *directed labeled graphs*) as a visualization scheme for RDF by representing a triple <`a b c`> by $a \xrightarrow{b} c$. However, the document leaves open how to deal with a statement property (an edge label) which occurs as the subject or object of another statement: one could either duplicate resources as nodes and as edge labels (as shown in figure 1), or allow edges to connect not only to nodes, but also to other edges.

Both approaches are inconvenient from several points of view: allowing multiple occurrences of resources as labels jeopardizes one of the most important aspects of graph visualization, which is the implicit assumption that the complete information regarding a node in a graph is obtained by its place in the drawing and its incident edges. On the other hand, the essential drawback of the second approach is the fact that the resulting construct is not a graph in the standard sense to which we could apply well-established techniques from graph theory. However, this is a principal reason for representing RDF by graphs: when reasoning formally over RDF data, e.g., as described in *RDF Semantics* [5], one has to operate with sets of triples. Although well-defined formally, a set of triples is a model that due to multiple occurrences of the same resource in the data structure leads to undesirable redundancies and does not capture the graph-like nature of RDF data, particularly regarding the connectivity of resources.

We propose to model RDF Graphs as bipartite graphs. RDF Graphs can be represented naturally by hypergraphs, and hypergraphs can be represented naturally by bipartite graphs. (Bipartite) graphs are well-known mathematical objects which, as formal representation, have several advantages over the triple or directed labeled graph representation discussed above. Among these advantages are: algorithms for the visualization of data for humans [6, 7], a formal framework to prove properties and specify algorithms, the availability of libraries with generic implementations of graph algorithms, and of course, techniques and results of graph theory. Representing RDF data by (standard) graphs allows to reduce application demands to well-studied problems of graphs. A few examples at hand: Difference between RDF Graphs: When are two RDF Graphs the same? [8, 9] Entailment: Determining entailment between RDF Graphs can be reduced to graph mappings: Is graph A isomorphic to a subgraph of graph B? [5]. Minimization: Finding a minimal representation of an RDF Graph is important for compact storage and update in databases [10]. Semantic relations between information resources: metrics and algorithms for semantic distance in graphs [11, 12]. Clustering [13, 14] and graph pattern mining algorithms [15] to reveal regularities in RDF data.

**Contributions.** In this paper we provide a formal graph-based intermediate model of RDF, which intends to be more concrete than the abstract RDF model to allow the exploit of results from graph theory, but still general enough to allow specific implementations. The contributions are the following: (1) We present a class of *RDF bipartite graphs* as an alternate graph representation for RDF. (2) We study properties of this class of graphs and the transformation of RDF data into it. (3) We provide an approach to stratify an RDF Graph into data and schema layers. (4) We explore the notion of *RDF connectivity* and how it is conveyed by the graph model.

**Related Work.** There is little work on formalization of the RDF graph model besides the guidelines given in the official documents of the W3C, particularly *RDF Concepts and Abstract Syntax* [4] and *RDF Semantics* [5]. There are works about algorithms on different problems on RDF Graphs, among them T. Berners-Lee's discussion of the *Diff* problem [8] and J. Carroll's study of the RDF graph matching problem [9]. Although not directly related to graph issues, there is work on the formalization of the RDF model itself that touches our topic: a logical approach that gives identities to statements and so incorporates them to the universe [16], a study oriented to querying that gives a formal typing to the model [17] and results on normalization of RDF Graphs [10].

In the thesis [18] there is an extended discussion of the model to be presented in this paper, containing all the proofs and further investigations on RDF maps and applications to RDF storage and querying.

## 2 Preliminaries

**RDF.** The atomic structure of the RDF language is the statement. It is a triple, consisting of a subject, a predicate and an object. These elements of a triple can be *URIs* (Uniform Resource Identifiers), representing information resources; *literals*, used to represent values of some datatype; and *blank nodes*, which represent anonymous resources. There are restrictions on the subject and predicate of a triple: the subject cannot be a literal, and the predicate must be a URI resource. Resources, blanks and literals are referred to together as *values.*

An *RDF Graph* is a set of RDF triples. Let T be an RDF Graph. Then $\text{univ}(T)$, the set of all values occurring in all triples of $T$, is called the *universe* of $T$; and $\text{vocab}(T)$, the *vocabulary* of $T$, is the set of all values of the universe that are not blank nodes. The *size* of $T$ is the number of statements it contains and is denoted by $|T|$. With $\text{subj}(T)$ (respectively $\text{pred}(T)$, $\text{obj}(T)$) we designate all values which occur as subject (respectively predicate, object) of $T$.

Let $V$ be a set of URIs and literal values. We define $\text{RDFG}(V) := \{T \mid T$ is RDF Graph and $\text{vocab}(T) \subseteq V\}$, i.e. the set of all RDF Graphs with a vocabulary included in $V$. There is a distinguished vocabulary, *RDF Schema* [19] that may be used to describe properties like attributes of resources (traditional attribute-value pairs), and to represent relationships between resources. It is expressive enough to defines classes and properties that may be used for describing groups of related resources and relationships between resources.

**Example RDF Graph 1** The `prefix:suffix` notation abbreviates URIs. The *wor* prefix identifies a "Web of Researchers" vocabulary (*rdfs* is RDF Schema)

```
1: <wor:Ullman> <wor:coauthor> <wor:Aho>
2: <wor:Greibach> <wor:coauthor> <wor:Hopcroft>
3: <wor:coauthor> <rdfs:subPropertyOf> <wor:collaborates>
4: <wor:Greibach> <wor:researches> <wor:topics/formalLanguages>
5: <wor:Valiant> <wor:researches> <wor:topics/formalLanguages>
6: <wor:Erdös> <wor:researches> <wor:topics/graphTheory>
7: <wor:Aho> <wor:collaborates> <wor:Kernighan>
8: <wor:Hopcroft> <wor:coauthor> <wor:Ullman>
```

**Graphs.** A *graph* is a pair $G = (N, E)$, where $N$ is a set whose elements are called *nodes*, and $E$ is a set of unordered pairs $\{u, v\}$ of nodes $u, v \in N$, the *edges* of the graph. A node $v$ and an edge $e$ are *incident* if $v \in e$; two edges $e_1$ and $e_2$ are *adjacent* if they both are incident to a node $v$. Observe that the definition implies that the sets $N$ and $E$ are disjoint.

A graph G is a *multigraph* if multiple edges between two nodes are permitted. A graph $G = (N, E)$ is said to be *bipartite* if $N = U \cup V, U \cap V = \emptyset$ and for all $\{u, v\} \in E$ it holds that $u \in U$ and $v \in V$. A *directed graph* is a graph where the elements of $E$ are ordered, i.e. there are functions from : $E \to N$ and to : $E \to N$ which yield the source and the target of each edge. In order to express more information, a graph can be *labeled.* A graph $(N, E)$, together with

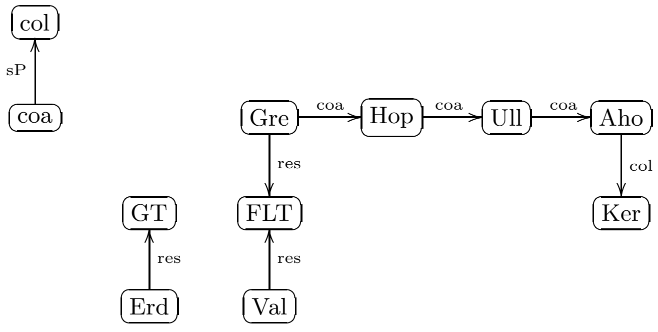

**Fig. 2.** Example RDF Graph 1 represented by a directed labeled graph. URI prefixes have been omitted, and labels have been abbreviated as follows: **col**laborates, sub**P**ropertyOf, **coa**uthor, **G**raph **T**heory, **res**earches, **Erd**ős, **Gre**ibach, **F**ormal **L**anguage **T**heory, **Val**iant, **Hop**croft, **Ull**man, **Aho**, **Ker**nighan.

a set of labels $L_E$ and an edge labeling function $l_E : E \to L_E$ is an *edge-labeled* graph. A graph is said to be *node-labeled* when there is a node label set and a node labeling function, as above. We will write $(N, E, l_N, l_E)$ for a node- and edge-labeled graph.

The notions of path and connectivity will be important in what follows. A *path* is a sequence of edges $e_1, \ldots, e_n$ where each edge $e_i$ is adjacent to $e_{i-1}$, for $i \in [2, n]$. The *label* of the path is $l_E(e_1) \cdot l_E(e_2) \cdot \ldots \cdot l_E(e_n)$ Two nodes $x, y$ are *connected* if there exists a path $e_1, \ldots, e_n$ with $x \in e_1$ and $y \in e_n$. The *length* of a path is the number of edges it consists of.

**RDF as Directed Labeled Graphs.** Now we can formalize the *directed labeled graph* representing an RDF Graph, as outlined in [4]. Let $T$ be an RDF Graph. Then define $\delta(T) = (N, E, l_N, l_E)$ as the node- and edge-labeled multigraph with $N = \{n_x : x \in \mathrm{subj}(T) \cup \mathrm{obj}(T)\}$ with $l_N(n_x) = x$ and $E = \{\, e_{s,p,o} : (s, p, o) \in T\,\}$ with $\mathrm{from}(e_{s,p,o}) = n_s$, $\mathrm{to}(e_{s,p,o}) = n_o$, and $l_E(e_{s,p,o}) = p$.

Figures 1 and 2 presents examples of such a graph. This definition yields a standard graph, but observe that node and edge label sets might not be disjoint (in figure 2, the resource `coa` appears at the same time as edge label and as node). This leads to problems as described in the introduction.

**Hypergraphs.** Informally, hypergraphs are systems of sets which extend the notion of graphs allowing edges to connect any number of nodes. For background see [20]. Formally, let $V = \{v_1, \ldots, v_n\}$ be a finite set, the *nodes*. A *hypergraph on $V$* is a pair $\mathcal{H} = (V, \mathcal{E})$, where $\mathcal{E}$ is a family $\{E_i\}_{i \in I}$ of subsets of $V$. The members of $\mathcal{E}$ are called *edges*. A hypergraph is *simple* if all edges are distinct. A hypergraph is said to be *r-uniform* if all edges have the cardinality $r$. An r-uniform hypergraph is said to be *ordered* if the occurrence of nodes in every edge is numbered from 1 to $r$.

$\mathcal{E}$ = { { coauthor, subPropertyOf, collaborates },{ Ullman, coauthor, Aho },{ Greibach, coauthor, Hopcroft } }

V= { collaborates, coauthor, subPropertyOf, Aho, Greibach, Hopcroft, Ullman }

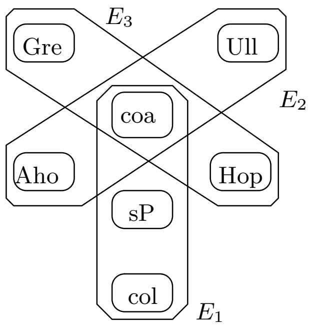

**Fig. 3.** Example of a simple 3-uniform hypergraph. This hypergraph represents the first three statements of the example RDF Graph 1.

Hypergraphs can be described by binary edge-node *incidence matrices*. In this matrix rows correspond to edges, columns to nodes: entry $m_{i,j}$ equals 1 or 0, depending on whether $E_i$ contains node $n_j$ or not. To the incidence matrix of a hypergraph $\mathcal{H} = (V, \mathcal{E})$ corresponds a *bipartite incidence graph* $B = (N_V \cup N_{\mathcal{E}}, E)$, which is defined as follows. Let $N_V$ be the set of node names of $\mathcal{H}$ which labeled the columns of the matrix, and $N_{\mathcal{E}}$ the set of edge names labeling its rows. Then $E$ contains an edge $\{e_i, v_j\}$ for each $e_i \in N_{\mathcal{E}}, v_j \in N_V$ where the matrix entry $m_{i,j}$ is 1. The obtained graph $B$ can be read to have an edge $\{e, v\}$ exactly when the hypergraph node represented by $v$ is member of the hypergraph edge represented by $e$. It is evident that $B$ is bipartite. Figure 4 shows the incidence matrix of a hypergraph and the bipartite incidence graph derived from it.

$$\begin{array}{c|ccccccc|} & \text{col} & \text{coa} & \text{sP} & \text{Aho} & \text{Gre} & \text{Hop} & \text{Ull} \\ E_1 & 1 & 1 & 1 & & & & \\ E_2 & & 1 & & 1 & & & 1 \\ E_3 & & 1 & & & 1 & 1 & \end{array}$$

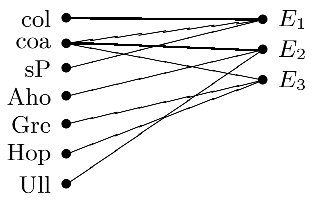

**Fig. 4.** Incidence matrix representing the hypergraph of Example 3 and the corresponding incidence graph. In the case of an ordered hypergraph, matrix entries will indicate the position of the occurrence of the node in the edge.

## 3 RDF Bipartite Graphs

**Deriving Bipartite Graphs from Hypergraphs.** One of the major problems encountered in trying to model RDF Graphs as classical graphs is the fact that an edge or labeled edge cannot represent the ternary relation given by an RDF triple. Therefore it is natural to turn the attention to graphs with 3-node-connecting edges instead of classical 2-node edges, that is, hypergraphs.

**Proposition 1.** *Any RDF Graph can be represented by a simple ordered 3-uniform hypergraph: every RDF triple corresponds to a hypergraph edge, the nodes being the subject, predicate and object in this order. The node set of the hypergraph is the union of all the edges. (Trivial)*

The converse of the proposition also holds when imposing constraints on the occurrences of blank nodes and literals: blank nodes may not be predicates and literals may not serve as subjects or predicates.

As stated in the preliminaries section, hypergraphs can be represented by incidence matrices where membership of a node in an edge is marked with a '1'. In the case of the hypergraph representing an RDF Graph, the nodes of an edge are ordered and we label them by S, P, or O to represent the role (subject, predicate, or object) of the information resource. Hence, when deriving the bipartite incidence graph of this incidence matrix, an edge will be added for every S, P, O entry of the matrix, and this edge will be labeled with the corresponding character. Thus, the only difference between the graph derived from the incidence matrix of any hypergraph and an RDF Graph hypergraph is the fact that each edge has one of three labels.

**Mapping RDF to RDF Bipartite Graphs.** This section presents a map of RDF Graphs to bipartite graphs. Let $\mathcal{B}$ be the set of bipartite labeled graphs $G = (V \cup St, E, \mathrm{nl}, \mathrm{el})$, $V \cap St = \emptyset$, where each edge in $E$ connects a node in $V$

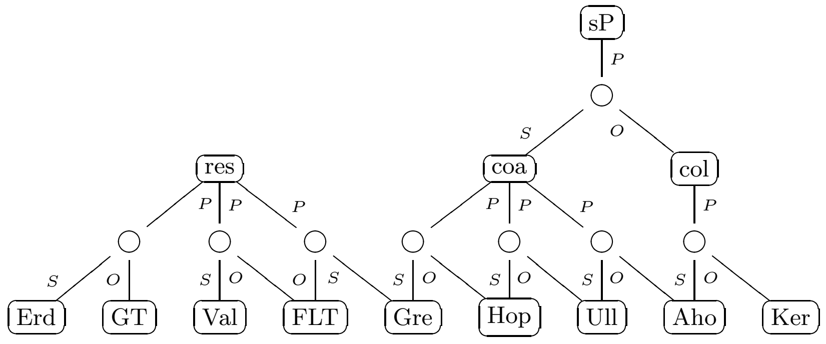

**Fig. 5.** The RDF bipartite graph of the example RDF Graph 1. Statement nodes are represented by circles; edge labels S, P, O indicate their subject, predicate and object.

with a node in $St$, and $\mathrm{el} : E \to \mathrm{EL}$ and $\mathrm{nl} : V \to \mathrm{NL}$ are labeling functions. The elements of $V$ are called the *value nodes* and those of $St$ the *statement nodes*.

**Definition 1 (RDF Bipartite Graph).** *Let $V$ be a vocabulary and $T$ an RDF Graph. Then we define a map $\beta : \mathrm{RDFG}(V) \to \mathcal{B}$ as follows: $\beta(T) = (V \cup St, E, \mathrm{nl}, \mathrm{el}) \in \mathcal{B}$ is the* RDF bipartite graph representing *$T$, with $V = \{v_x : x \in \mathrm{univ}(T)\}$; $St = \{st_t : t \in T\}$; and the set of edges $E$ is built as follows: for each triple $t = (x, y, z) \in T$ add the edges $\{st_t, v_x\}$ with label $S$, $\{st_t, v_y\}$ with label $P$, and $\{st_t, v_z\}$ with label $O$. The labeling of the nodes is given by:*

$$\mathrm{nl}(v_x) := \begin{cases} (x, d_x) & \textit{if } x \textit{ is literal } (d_x \textit{ is the datatype identifier of } x) \\ x & \textit{else} \end{cases}$$

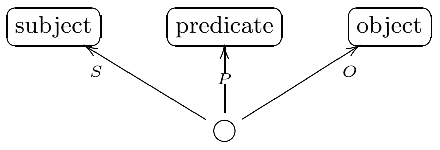

**Fig. 6.** A statement as RDF bipartite graph.

Note that $\beta(T)$ is a *3-regular* bipartite graph, because the degree of each node in $St$ is 3. This representation incorporates explicitly the statements as nodes into the graph.

*Example 1.* Figure 6 illustrates how a single statement is represented as RDF bipartite graph. Figure 5 shows the RDF bipartite graph representation of the Web Of Scientists example. The drawing convention is as follows: unlabeled circles represent statement nodes, boxes with rounded corners value nodes. Edge labels S, P, and O indicate the subject, predicate, and object of a statement.

The model is well-defined and one can go back and forth between $T$ and $\beta(T)$:

**Proposition 2.** *For each RDF Graph $T$ there is a uniquely defined RDF bipartite graph $\beta(T)$ representing it. Moreover, a function $\beta^{-1} : \beta(\mathrm{RDFG}(V)) \to \mathrm{RDFG}(V)$ exists satisfying $\beta^{-1}(\beta(T)) = T$.*

The graph created from $T$ has reasonable size, and can be obtained efficiently[1]:

**Proposition 3.** *Let $T$ be an RDF Graph and $\beta(T) = (V \cup St, E, \mathrm{nl}, \mathrm{el})$. Then:*

1. *$\beta(T)$ can be computed in time $O(|T| \lg |T|)$.*
2. *The graph $\beta(T)$ is bounded as follows: $|St| = |T|$, $|V| = |\mathrm{univ}(T)|$ and $|E| = 3\,|T|$.*

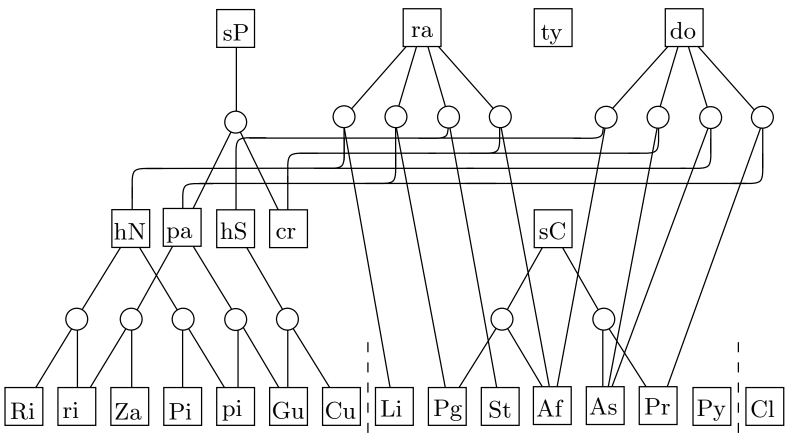

**Fig. 7.** RDF bipartite graph of the museum example. Edge labels have been omitted for clarity. Drawing levels indicate the order of the values nodes. At the bottom level are values which never occur as predicates: class instances like (bold letters indicate the abbreviations) the literal "**Ri**vera", **ri**vera (resource), **Za**pata, classes such as **P**ainting, **A**rti**f**act, **A**rti**s**t, **P**ainte**r**, **P**roperty, and the meta-class **Cl**ass. Simple properties include **h**as_**N**ame and **pa**ints, and properties of properties are sub**P**roperty, **do**main, **ra**nge and **ty**pe. Declarations with property "type" are shown in figure 10.

## 4 The Structure of RDF Graphs

RDF Graphs consist of values and statements. A first coarse-grained division of the statements is the following: those that define a schema, and those that are data structured under this schema (see figure 1—the dotted line represents this division). Although RDF does not distinguish between schema-defining and data statements, this distinction is natural when considering storage [21] and querying [17] in databases. Unfortunately, a plain discrimination between the data and schema parts of an RDF specification is not always possible. Moreover, features like extensibility of specifications and reification make this divide difficult to grasp formally. In the following we present two approaches to this issue.

**Definition 2.** *A* data subgraph *of an RDF Graph $T$ is a maximal subgraph $T'$ satisfying* $(\mathrm{subj}(T') \cup \mathrm{obj}(T')) \cap \mathrm{pred}(T') = \emptyset$*. The* schema subgraph *associated to $T'$ is $T \setminus T'$.*

The terms of data and schema subgraph correspond to the *description base* and *description schema* in the wording of [22]. By the definition, note that an RDF Graph $T$ does not have a uniquely defined data subgraph, e.g. consider $\{(a, b, c), (b, d, e)\}$ where each statement alone is a data subgraph. Moreover, an RDF Graph could have exponentially many different ones (proposition 13 in [18]).

[1] The proofs of all propositions can be found in [18].

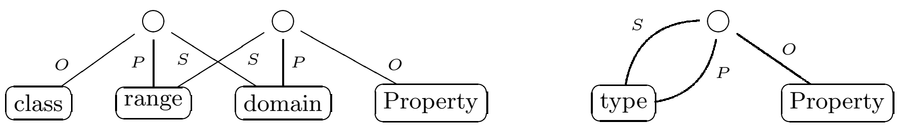

**Fig. 8.** Unstratified RDF Graphs, both examples are part of the RDF axiomatic triples.

An alternate approach is to decompose the RDF Graph $T$ into *strata*. This is grounded on the idea that a property describing basic-level entities is a predicate, and a property describing predicates is a predicate of higher order.

**Definition 3 (Stratification).** *Let $T$ be an RDF Graph. Define $V_i(T)$ and $St_j(T)$ by mutual recursion as follows: $V_0(T)$ is the set of values of $T$ that are not predicates, $V_n(T)$ is the set of values of $T$ that are predicates of statements in $St_n(T)$, and $St_{n+1}(T)$ is the set of statements of $T$ whose subject and object are elements of $\bigcup_{j\leq n} V_j(T)$.*
*The* order *of $T$ is the maximum $n$ such that $V_{n-1}$ is not empty.*

*$T$ is* stratified *if* $\mathrm{univ}(T) = \bigcup_{j\geq 0} V_j(T)$.

*Example 2 (Stratification).*

– $T = \{(a, a, b)\}$ cannot be stratified: $V_0(T) = \{b\}$ and $V_j(T) = \emptyset$ for $j > 0$. (e.g., <`rdf:type rdf:type rdf:Property`> is an RDF *axiomatic triple* [5]—see figure 8.)
– The reification of a triple <`a b c`> is stratified (see figure 9).
– The museum example (figure 1) can be partitioned into 3 levels (see figure 7).

**Proposition 4 (Unstratifiedness).** *$T$ is not stratified if and only if there is a cycle from a value node in $\beta(T)$ with label in $[(O|S)P]^+$.*

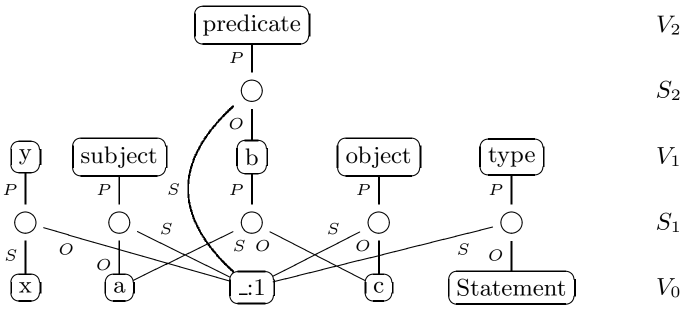

**Fig. 9.** Stratified drawing of a reification. A resource `x` makes a proposition `y` about the statement <`a b c`>.

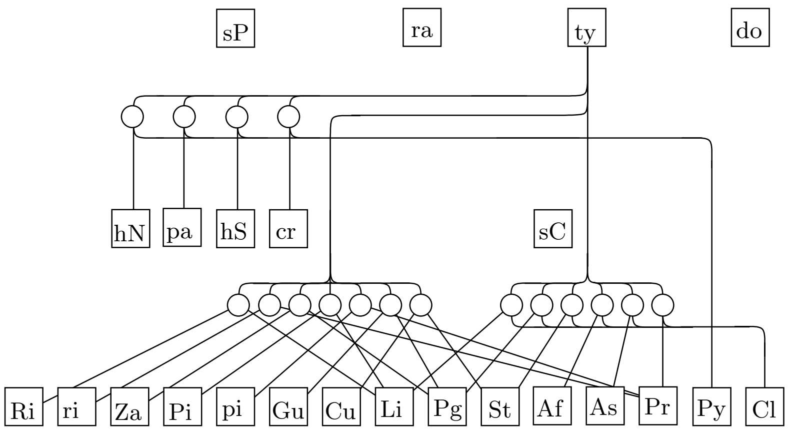

**Fig. 10.** Type declarations of the RDF bipartite graph in figure 7.

Although the complete axiomatic specification of RDF is not stratified (see figure 8), most RDF data currently found in practice is stratified—if RDF axiomatic triples are not considered—and has small order (no bigger than 3). Also, if $T$ is stratified, the graph obtained by reifying each of its triples is stratified. However, the union of two stratified RDF Graphs is not necessarily stratified.

Both approaches embrace RDF's extensibility in that they do not rely on the `rdfs` namespace prefix. The proposition of an alternate RDF Schema *RDFS(FA)* by Pan and Horrock [23, 24] is based on similar considerations and also provides a potentially unbounded stratification.

## 5 Connectivity of RDF Data

An RDF Graph conveys more meaning than the sum of its statements considered individually. For example, Greibach is a coauthor of Hopcroft, who is a coauthor of Ullman (figure 2). We can deduce that Greibach and Ullman are related by a "transitive co-authorship", although this is not stated explicitly by a single statement. In the museum example—consider the fragment in figure 11—Picasso and Rodin are related in that they both paint. Complex relationships between resources are called *semantic associations* in [25, 11]; for storage and querying the importance of considering the *connectedness* of resources has been argued in [21, 26–28]. Guha et al. distinguish between a *logical* and a *physical* model for querying and inferencing RDF [26]. Such a logical model would have to truly represent the graph nature of RDF in contrast to a concrete serialization syntax (the "physical model") such as RDF/XML or triple storage in databases.

This section introduces the concept of *connectivity* for RDF by means of *paths* and relates these terms to their counterparts from graph theory.

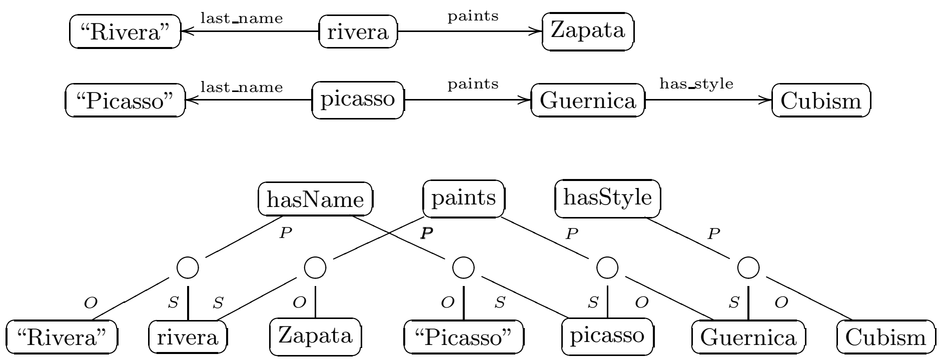

**Fig. 11.** Fragment of the museum example, presented first as a directed labeled graph, and below as RDF bipartite graph.

**Definition 4 (Triple Path).** *Let $T$ be an RDF Graph. A* path in an RDF Graph*—or* triple path*—$P$ is a sequence of RDF triples $(t_1, t_2, \ldots, t_n)$ with $t_k = (s_k, p_k, o_k) \in T$, for which it holds that*

$$\textit{for all } i < n, \{s_i, p_i, o_i\} \cap \{s_{i+1}, p_{i+1}, o_{i+1}\} \neq \emptyset.$$

From this notion the concept of *RDF Connectedness* naturally follows: A triple path $(t_1, t_2, \ldots, t_n)$ is said to be *connecting* resources $x$ and $y$ if it holds that $x \in \{s_1, p_1, o_1\}$, $x \notin \{s_i, p_i, o_i : 1 < i \leq n\}$ and $y \in \{s_n, p_n, o_n\}$, $y \notin \{s_i, p_i, o_i : 1 \leq i < n\}$. The concepts "triple path" and "RDF connectedness" stress the anchoring in the RDF model and are independent of the graph representation (e.g., RDF bipartite graph or directed labeled graph) or serialization syntax.

RDF connectivity corresponds to the well-established notion of connectedness in classical graphs:

**Proposition 5.** *Let $T$ be an RDF Graph. The resources $x$, $y$ are connected in $T$ if and only if there exists a path in $\beta(T)$ between the corresponding nodes $v_x$ and $v_y$.*

*Example 3 (Paths in RDF Bipartite Graphs).* The lower part of figure 11 shows the RDF bipartite graph version of the directed labeled graph depicted above. All paths of the former also exist in the latter, but not vice versa: the resources `picasso` and `rivera` now appear connected (via the property `paints`, or `hasName`), although the directed labeled graph version does not show this. In the sense of the definition of RDF Graph connectivity given above, these resources are indeed related, and the directed labeled graph version fails to represent that.

The example shows that not all paths which exist in an RDF Graph are present in its directed labeled graph. The following definition describes those paths which are represented:

**Definition 5 (Horizontal Path).** *A* horizontal (triple) path *$P$ in an RDF Graph $T$ is a sequence of RDF triples $(t_1, t_2, \ldots, t_n)$, $t_k = (s_k, p_k, o_k) \in T$, for which it holds that for all $i < n$, $\{s_i, o_i\} \cap \{s_{i+1}, o_{i+1}\} \neq \emptyset$.*

*A horizontal path is said to be* oriented *if $o_i = s_{i+1}$ for all $i < n$.*

**Proposition 6.** *For every oriented horizontal path $P$ in an RDF Graph $T$ there exists a corresponding path $\delta(P)$ in its directed labeled graph $\delta(T)$, and vice versa.*

Note that the non-horizontal paths are sequences of triples $t_1, \ldots, t_n$ where for some $j$, $t_j$ and $t_{j+1}$ are linked via a predicate. Examples are the relation of Picasso and Rivera via `paints` presented at the beginning of the section, reifications, and statements about a predicate. If we make a parallel to relational databases, the notion of horizontal path corresponds roughly to values of tuples linked via joins; on the other hand, "vertical" paths are paths passing through the schema of the database. Vertical paths are also relevant for RDF querying when sub-class and sub-property semantics are incorporated (e.g., RQL [29]). Other examples for the use of vertical connectedness arise from the various types of "semantic associations" presented in [25]: *similar* paths are horizontal paths where the properties are sub-properties of a common ancestor property.

## 6 Conclusions

We introduced a representation of RDF Graphs in terms of classical bipartite graphs as an intermediate model between the abstract RDF triple syntax and concrete implementations. We presented preliminary results about the structure and lines of development of the model. We argued the advantages of the model compared to the triple representation and to the directed labeled graph representation used currently by default.

One of the main advantages of our model from a developer point of view is the possibility to directly use standard graph libraries. We are using the model to approach diverse algorithmic problems of RDF databases, particularly graph-like notions in querying and storage. Future work includes refinement of the RDF bipartite graph model and aspects of visualization.

## References

1. Lassila, O., Swick, R.R.: Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation. World Wide Web, `http://www.w3.org/TR/1999/REC-rdf-syntax-19990222`, (1999)
2. Miller, E., Swick, R., Brickley, D.: Resource Description Framework (RDF) / W3C Semantic Web Activity. World Wide Web, `http://www.w3.org/RDF/` (2004)
3. Berners-Lee, T.: Semantic Web Road Map. World Wide Web, `http://www.w3.org/DesignIssues/Semantic.html` (September 1998)
4. Klyne, G., Carroll, J.J.: Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation. World Wide Web, `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/` (10 February 2004)

5. Hayes, P.: RDF Semantics. W3C Recommendation. World Wide Web, `http://www.w3.org/TR/2004/REC-rdf-mt-20040210/` (10 February 2004)
6. di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Algorithms for Drawing Graphs: An Annotated Bibliography. Comput. Geom. Theory Appl. **4** (1994) 235–282 `http://www.cs.brown.edu/people/rt/gd-biblio.html`.
7. Mäkinen, E.: How to Draw a Hypergraph. Intern. J. Computer Math. **34** (1990) 177–185
8. Berners-Lee, T., Connolly, D.: Delta: an Ontology for the Distribution of Differences between RDF Graphs. World Wide Web, `http://www.w3.org/DesignIssues/Diff` (2004)
9. Carroll, J.J.: Matching RDF Graphs. Technical Report HPL-2001-293, Digital Media Systems Laboratory, HP Laboratories, Bristol, `http://www.hpl.hp.com/techreports/2001/HPL-2001-293.html` (2001)
10. Gutierrez, C., Hurtado, C., Mendelzon, A.O.: Foundations of Semantic Web Databases. In: Proceedings of the Twenty-third ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), June 14-16, 2004, Paris, France, ACM (2004)
11. Aleman-Meza, B., Halaschek, C., Arpinar, I.B., Sheth, A.P.: Context-Aware Semantic Association Ranking. In Cruz, I.F., Kashyap, V., Decker, S., Eckstein, R., eds.: Proceedings of SWDB'03. (2003)
12. Rodríguez, A., Egenhofer, M.: Determining Semantic Similarity Among Entity Classes from Different Ontologies. IEEE Transactions on Knowledge and Data Engineering **15(2)** (2003) 442–456
13. Carrasco, J.J.M., Fain, D., Lang, K., Zhukov, L.: Clustering of Bipartite Advertiser-Keyword Graph. IEEE Computer Society (2003)
14. Zha, H., He, X., Ding, C.H.Q., Gu, M., Simon, H.D.: Bipartite Graph Partitioning and Data Clustering. In: Proceedings of the 2001 ACM CIKM, Atlanta, USA, ACM (2001) 25–32
15. Vanetik, N., Gudes, E., Shimony, S.E.: Computing Frequent Graph Patterns from Semistructured Data. In: Proceedings of ICDM 2002, Maebashi City, Japan, IEEE Computer Society (2002)
16. Yang, G., Kifer, M.: On the Semantics of Anonymous Identity and Reification. In Meersman, R., Tari, Z., eds.: On the Move to Meaningful Internet Systems, 2002 - DOA/CoopIS/ODBASE, Irvine, USA, Proceedings. Volume 2519 of Lecture Notes in Computer Science., Springer (2002)
17. Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., Scholl, M.: RQL: A Declarative Query Language for RDF. In: Proceedings of 2002 WWW Conference, ACM Press (2002) 592–603
18. Hayes, J.: A Graph Model for RDF. Diploma thesis, Technische Universität Darmstadt, Department of Computer Science, Germany, `http://purl.org/net/jhayes/rdfgraphmodel.html` (2004)
19. Brickley, D., Guha, R.: RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation. World Wide Web, `http://www.w3.org/TR/2004/REC-rdf-schema-20040210/` (10 February 2004)
20. Duchet, P.: Hypergraphs. In Graham, R., Grötschel, M., Lovász, L., eds.: Handbook of Combinatorics. Elsevier Science B.V., Amsterdam (1995) 381–432
21. Matono, A., Amagasa, T., Yoshikawa, M., Uemura, S.: An Indexing Scheme for RDF and RDF Schema based on Suffix Arrays. In Cruz, I.F., Kashyap, V., Decker, S., Eckstein, R., eds.: Proceedings of SWDB'03. (2003) 151–168

22. Karvounarakis, G., Magkanaraki, A., Alexaki, S., Christophides, V., Plexousakis, D., Scholl, M., Tolle, K.: RQL: A Functional Query Language for RDF. In Gray, P., Kerschberg, L., King, P., Poulovassilis, A., eds.: The Functional Approach to Data Management. Springer-Verlag (2004)
23. Pan, J., Horrocks, I.: Metamodeling Architecture of Web Ontology Languages. In Cruz, I.F., Decker, S., Euzenat, J., McGuinness, D.L., eds.: Proceedings of SWWS'01, The first Semantic Web Working Symposium, Stanford University, California, USA, July 30 - August 1, 2001. LNCS, Springer-Verlag (2001)
24. Pan, J., Horrocks, I.: RDFS(FA) and RDF MT: Two Semantics for RDFS. In Fensel, D., Sycara, K., Mylopoulos, J., eds.: Proc. of the 2003 International Semantic Web Conference (ISWC 2003). Number 2870 in Lecture Notes in Computer Science, Springer (2003) 30–46
25. Anyanwu, K., Sheth, A.: $\rho$-Queries: Enabling Querying for Semantic Associations on the Semantic Web. In: Proceedings of the Twelfth International World Wide Web Conference, ACM Press (2003) 690–699
26. Guha, R., Lassila, O., Miller, E., Brickley, D.: Enabling Inferencing. World Wide Web, `http://www.w3.org/TandS/QL/QL98/pp/enabling.html` (1998)
27. Karvounarakis, G., Christophides, V., Plexousakis, D.: Querying Semistructured (Meta)Data and Schemas on the Web: The case of RDF and RDFS. Technical Report 269, FORTH Institute of Computer Science (2000)
28. Haase, P., Broekstra, J., Eberhart, A., Volz, R.: A Comparison of RDF Query Languages. World Wide Web, `http://www.aifb.uni-karlsruhe.de/WBS/pha/rdf-query/rdfquery.pdf` (2004)
29. FORTH Institute of Computer Science `http://139.91.183.30:9090/RDF/RQL/Manual.html`: RQL v2.1 User Manual. (2003)