Stratifying Semantic Data for Citation and Trust: an Introduction to RDFDF

Dario De Nart, Dante Degl'Innocenti,
Marco Peressotti, and Carlo Tasso

Department of Mathematics and Computer Science
University of Udine, Italy
{dario.denart,marco.peressotti,carlo.tasso}@uniud.it,
{deglinnocenti.dante}@spes.uniud.it

Abstract. In this paper we analyse the functional requirements of linked data citation and identify a minimal set of operations and primitives needed to realise such task. Citing linked data implies solving a series of data provenance issues and finding a way to identify data subsets. Those two tasks can be handled defining a simple type system inside data and verifying it with a type checker, which is significantly less complex than interpreting reified RDF statements and can be implemented in a non data invasive way. Finally we suggest that data citation should be handled outside of the data, and propose a simple language to describe RDF documents where separation between data and metainformation is explicitly specified.

Keywords: RDF, Semantic Web, Data Citation, Theory, Data Trust, Semantic Publishing

Over the last years data has become a more and more critical asset both in research and in application. While there is a general agreement on the need for data citation to ensure research reproducibility and to facilitate data reuse, the research community is still debating how to concretely realize it. Citing data is not a trivial task since it has a few notable differences from citing literature: data evolve over time, data availability might change over time, only a subset of data might be relevant, and on top of that the authorship of data is not always clear since it may be the result of an automated process (e.g. sensor data), involve a large number of contributors (e.g. crowdsourcing), or even be built on the top of other data (e.g. inferring a taxonomy from a document corpus). Leveraging on the insights provided by [2], [3], [13], and [15] we outline the following Data Citation functional requirements:

- *Identification and Access*: Data Citation should provide a persistent, machine readable, and globally unique identifier for data; Moreover a reference to a persistent repository should also be provided to facilitate data access.
- *Credit and Attribution*: Data citation should facilitate giving credit and legal attribution to all contributors to the data. Such contributors might be humans as well as automated processes such as reasoners;

– *Evolution*: Data Citation should provide a reference to the exact version of the cited data, since data might change over time. This is a fundamental requirement for research reproducibility purposes.

An additional, non functional requirement, is efficiency: the data citation should lead to the data in practical time, which means fast enough for the purposes of data consumer applications. For instance a database query allows to access the data in practical time, while solving a complex set of logical clauses probably does not. In the last years Linked Data has rapidly emerged as the preferred format for publishing and sharing structured data, creating a vast network of interlinked datasets [9]. However the open nature of the format makes data provenance hard to track, moreover the RDF Recommendation does not provide a clear mechanism for expressing metainformation about RDF documents. Semantic Web technologies such as OWL, RDF, and RDFS leverage upon description logic and first order logic and it is well known that an incautious usage of their primitives may lead to non decidable sets of conditions [10]. With respect to the requirements of a good data citation expressed above, the Semantic Web community has proposed a number of solutions to the data provenance problem which addresses the problem of assessing the authorship of data. Methods for partitioning RDF graphs have been proposed as well and also version identification and storage of RDF data have already been discussed. However most of those solutions imply the embedding of metainformation inside RDF data. This practice tends to make data cumbersome and the usage of reification [8] to realize tasks such as generating data subsets may lead to a combinatorial explosion of triples. In this paper we discuss a simple framework to satisfy data citation requirements leveraging on the stratification of linked data, which basically means providing a separation between proper data and metainformation. Such separation can be effectively guaranteed with the usage of a simple type system allowing programs such as reasoners to discriminate in an efficient way. We'd also like to show that the fact that Linked Data technologies such as RDF and OWL are powerful enough to let you seamlessly represent and embed metainformation inside the data does not mean that you really *should*.

1 Related Work

Data citation has already been explored by the Semantic Web community and it significantly overlaps with the problem of assessing data provenance since determining the authorship of data is vital for citation purposes and both tasks need metainformation over data. Provenance has already been widely discussed by the Semantic Web community leveraging on the insights provided by the Database commmunity [5]. Provenance information can be represented exploiting two approaches: the annotation approach and the inversion approach [12]. In the first approach all metainformation is explicitly stated, while in the latter is computed when needed in a lazy fashion which requires external resources containing the information upon which provenance is entailed to be constantly available. The annotation approach is favored since it provides richer information

and allows data to be self-contained; several vocabularies have been proposed to describe metainformation over linked data such as *VoID* (Vocabulary of Interlinked Datasets) [1], that offers a rich language for describing Semantic Web resources built on top of well known and widely used ontologies such as foaf[1] and Dublincore[2], and *PROV Ontology* (PROV-O)[3], which is the lightweight ontology for provenance data standardized by the W3C Provenance Working Group. Regardless of the vocabulary used, adopting the annotation approach will result in producing a lot of metainformation which might exceed the actual data in size: provenance data in particular increases exponentially with respect to its depth [14]. For more information about the problem of data provenance, we reference the curious reader to [4]. The state of the art technique for embedding metainformation in RDF, is reification [15] which consists in assigning a URI to a RDF triple by expressing it as an *rdf:Statement* object. Recently the RDF 1.1 Recommendation [11] introduced the so called "RDF Quad Semantic" which consists in adding a fourth element to RDF statements which should refer to the name of the graph which the triple belongs to. The actual semantic of the fourth element however is only hinted, leaving room for interpretation and therefore allowing semantics tailored to fit application needs. In [13] is presented a methodology for citing linked data exploiting the quad semantics: the fourth element is used as identifier for RDF predicates allowing the definition of data subsets. Other usages of the fourth element include specification of a time frame, uncertainty marker, and provenance information container [6]. Finally, the idea of using a type system to ease the fruition of semantic resources is not new to the Semantic Web community: the authors of [7] propose a type system to facilitate programmatic access to RDF resources.

2 Well Stratified Linked Data

To introduce the concept of stratification of data we need to define a formal representation of metainformation in linked data that abstracts over the actual representation of such information (e.g. reification). Let us assume all the vocabulary of a collection of RDF data to be included in a set of labels named V and all the IRIs to be in U, subset of V. Given those definitions, associating labels to RDF graphs -thus creating named graphs- can be described by a partial function $n : u \mapsto (v, u', v')$ mapping each $u \in U$ to at most one triple $(v, u', v') \in V \times U \times V$. This function, that we will refer to herein as *naming function* maps identifiers to non void RDF graphs. Intuitively, assigning an IRI to a triple puts that IRI in the rôle of *metainformation* with respect to that triple whence thought as *information*. Note that the separation between information and metainformation is not absolute but *relative* to the context i.e. the level at which the reasoning happens. For a concrete example, consider the following RDF snipped:

[1] `http://xmlns.com/foaf/spec/`
[2] `http://dublincore.org/documents/dcmi-terms/`
[3] `http://www.w3.org/TR/prov-o/`

x `type` `statement`
x `subject` y
x `predicate` b
x `object` c
y `type` `statement`
y `subject` a
y `predicate` b
y `object` c

Accordingly to the reification semantics, here x is assigned to triple (y, b, c) and y to the triple (a, b, c) hence, the naming function over the considered data includes the following associations:

$$x \mapsto (y, b, c) \qquad \text{and} \qquad y \mapsto (a, b, c).$$

Clearly, y plays the rôle of metainformation with respect to the triple (a, b, c) and x plays the rôle of metainforamtion about (y, b, c) whence (a, b, c). From now on we will refer to a vocabulary with an associated naming function as a *Named Graph Family* (herein NG family), moving along the lines of [6].

We consider a particular class of NG families called *well-stratified* with the fundamental property of stratifying metainformation over information in a way that prevents any infinite chain of "downward" references where the direction is interpreted as crossing the boundary between metainformation and information. Since practical NG families (hence triple stores) contain only a finite amount of explicit information, absence of such chains corresponds to the absence of cycles of references like, for instance, in the NG family:

$$x \mapsto (y, b, c) \sqcup y \mapsto (x, b, c).$$

In more formal terms, a relation R on a set X is *well-founded* whenever every non-empty subset S of X has a minimal element i.e. there exists $m \in S$ that is not related by $s\, R\, m$ for $s \in S$. This means that we can intuitively walk along R going from right to left for finitely many steps i.e we have to stop, eventually. This implies that R does not contain infinite descending chains (i.e. an infinite sequence $x_0, x_1, x_2, \ldots$ such that $x_{n+1}\, R\, x_n$). We call a NG family *well-stratified* whenever it comes equipped with a well-founded relation $\prec$ on its vocabulary such that the naming function n descends along $\prec$ i.e.:

$$n(u) = (a, b, c) \implies u \succ a \wedge u \succ b \wedge u \succ c.$$

The relation $\prec$ is called *witness* for n. It is intuitive that identifying what serves as metainformation and information is easily computable over well-stratified NG families, while non well-founded NG families include cycles wherein it is impossible to separate information from metainformation.

We now introduce the concept of *abstract reasoner* as anything that might alter data either by simply adding new triples (monotonic reasoners) or by deleting and replacing existing triples (non-monotonic reasoners). With abstract reasoners we represent both automated reasoning (and other data managing tasks) and

human annotation. Abstract reasoners may easily break well-stratification. Intuitively most reasoning tasks and well-engineered human annotation processes should preserve stratification, however breaking the well-stratification of data is subtle and can be achieved even with monotonic reasoning. For instance, consider a set of triples where there exists a triple $(y, \texttt{type}, \texttt{statement})$ labelled with some IRI x and an abstract reasoner γ that adds a new triple $(x, \texttt{type}, \texttt{statement})$ labelled as y. This insertion is totally legit if we are using reification but introduces a circularity in the chain of meta data since the family now contains the following assignments:

$$x \mapsto (a, \texttt{type}, \texttt{statement}) \qquad a \mapsto (x, \texttt{type}, \texttt{statement})$$

and hence is no more well-stratified. We are interested in a class of abstract reasoners, called *coherent*, that preserve the well-stratification property of named graph families they operate on. Intuitively, reasoners for provenance, subsetting and versioning are coherent as they cross the boundary between information and meta information only in one direction: descent. However in order to let such reasoners terminate the well-stratification of data must be guaranteed, therefore, every abstract reasoner that interacts with data prior to the resolution of a data citation must be coherent as well. Wrapping up these considerations, data citation can happen if data is well stratified and data can be well stratified if all the abstract reasoners that built, expanded, and verified it are coherent. It is therefore needed a practical solution to assess the coherence of abstract reasoners.

3 Resource Description Framework Description Framework

In this section we propose a possible specification of a simple language called *RDFDF* whose main purpose is to guarantee the well stratification of data, hence the coherence of reasoning. RDFDF can be considered an extension to regular RDF, however it introduce some new top level concepts making it a new language built on the top of RDF rather than its extension (like OWL and RDFS). By design, RDFDF is a superset of RDF, thus every well formed RDF document is a well formed RDFDF document as well. RDFDF makes explicit the so-called fourth element introduced by the RDF 1.1 recommendation [11] and the basic unit of the RDFDF language is the quadruple (s, p, o, i) where s, p, and o are the subject, predicate, and object of regular RDF triples, and i is the optional *identifier* labelling the RDF triple. With this fourth element the naming function described in 2 can be easily represented since $x \mapsto (s, p, o)$ can be conveniently written as (s, p, o, x). IRIs that appear as fourth object are implicitly of type *meta-resource*, a sibling class to *rdfs:resource* which now is a subclass of a broader class called *data* which includes both information and metainformation. Resource and meta-resource are not disjoint, indeed an IRI that is used as identifier and as subject, object or predicate of another triple

belongs to both of them. The rationale behind this choice is that identifiers can be described using resources such as data provenance ontologies, therefore interoperability between resources and meta-resources is needed, and there can exist multiple levels of meta-information such that the meta-information of a certain level serves as information for the upper one. In Figure 1 we show a practical example of a case where an IRI, in this case U, belongs to both *meta-resource* and *rdfs:resource* since it is used as identifier for the triple (A, B, C) and as subject of $(U, author, foaf : x)$.

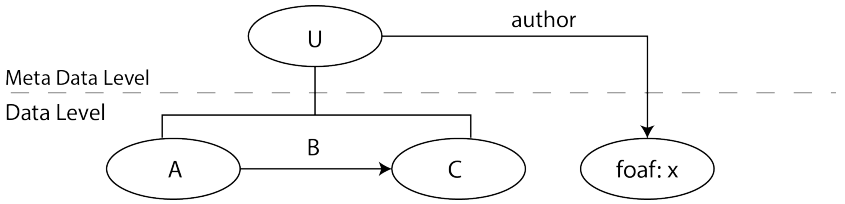

Fig. 1. An example of metainformation stratification in RDFDF.

At the semantics level, however, the most notable difference is the introduction of a basic *type system*: while the *rdf:type* property still indicates class membership, RDFDF introduces types at the data set level to check the well-stratification of data. In this context each IRI belongs to a specific level of information: the first one is information, the second meta-information, the third meta-meta-information, and so on. The absence of loops between such layers can be checked with a simple *type system*, similar to the ones used in programming language, to ensure the well-stratification of RDFDF documents. RDFDF introduces a simple type system whose only type ✓ is inhabited by exactly well-stratified data. Judgements are of the form

$$\Gamma \vdash n \colon \checkmark$$

where $n \colon U \rightharpoonup V \times U \times V$ is a data set (which corresponds to a family of named graphs) and the stage Γ is a partial function from the vocabulary V to a well-founded structure. For instance, could map V to the set of natural numbers under the successor relation: $\Gamma \colon V \rightharpoonup \mathbb{N}$. The proposed type system is composed by three typing rules:

$$\frac{}{\Gamma \vdash \varnothing \colon \checkmark}$$

$$\frac{\Gamma(x) > \Gamma(a) \quad \Gamma(x) > \Gamma(b) \quad \Gamma(x) > \Gamma(c)}{\Gamma \vdash x \mapsto (a, b, c) \colon \checkmark}$$

$$\frac{\Gamma_1 \vdash n_1 \colon \checkmark \quad \Gamma_2 \vdash n_2 \colon \checkmark \quad \Gamma = \Gamma_1 \sqcup \Gamma_2 \quad n = n_1 \sqcup n_2}{\Gamma \vdash n \colon \checkmark}$$

The first captures the fact that an empty document is always well-stratified. The second ensures that Γ describes relations on V such that the assignment

$x \mapsto (a, b, c)$ is well-stratified. Finally, the third rule allows to break n and Γ reducing the problem to smaller objects which can then be checked separately (clearly, applying this rule with either n_1 or n_2 being $\varnothing$ is pointless). We do not need to "guess" Γ. This function can be obtained by applying the above typing judgements while considering Γ as an unknown collecting all the hypotheses on it (e.g. $\Gamma(x) > \Gamma(a)$ from the second rule) in a set of constraints. Any partial function satisfying these constraints can be used as Γ to derive $\Gamma \vdash n\colon \checkmark$. Computing such solutions can be done pretty efficiently and the choice of the algoritm to be used is left to the implementer, anyway, RDFDF allows to provide explicit typing annotations as separate meta-data with the *level* property, as in the following snippet.

```
a, rdf:type, b, x
x, rdfdf:level, 1
```

The *level* property is defined as an *owl:AnnotationProperty* and therefore is ignored by OWL reasoners and is intended solely for type inference purposes. An explicitly typed document can be verified efficiently.

4 Conclusions and Future Works

In this paper we briefly introduced the concept of well stratification over linked data and highlighted how achieveing well stratification over data is a fundamental requirement to realise data citation over RDF data. With respect to the problem of data citation, the expressive power of OWL and RDF is largely overabundant and might be harmful since a misuse of their primitives might break the stratification of information and metainformation, thus making resolving citations an undecidable problem. In our opinion, a more restricted language, designed specifically to grant the stratification of data, like RDFDF, should be taken into consideration to effectively enable problems such as data citation and provenance assessment to be resolved in practical time, allowing the creation of an effective data trust layer. Attaching RDFDF data or some other kind of well stratified metainformation to data published on the Web might be, in our opinion, Linked Open Data's sixth star, like publishing versioned code is a fundamental quality requirement for Open Source software. The similarity between data metainformation handling and source code versioning is striking since they address similar problems: tracking who and how edited something, identifying subsets of the managed items, and allowing external application or documents to refer to a specific revision. In our opinion this separation is also consistent with the present development of the Semantic Web stack: OWL itself, thought being a logical extension of RDFS, is not built on the top of RDFS but is rather a distinct language sharing concepts and primitives with RDFS. In a similar way a new language for data metainformation management could be built compatibly with RDF and the Linked Data philosophy without being RDF.

References

1. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing linked datasets with the void vocabulary (2011)
2. Altman, M., Borgman, C., Crosas, M., Matone, M.: An introduction to the joint principles for data citation. Bulletin of the American Society for Information Science and Technology 41(3), 43–45 (2015)
3. Altman, M., Crosas, M.: The evolution of data citation: From principles to implementation. IAssist Quarterly 63 (2013)
4. Anam, S., Kang, B.H., Kim, Y.S., Liu, Q.: Linked data provenance: State of the art and challenges. In: Proceedings of the 3rd Australasian Web Conference (AWC 2015). vol. 27, p. 30 (2015)
5. Buneman, P., Khanna, S., Wang-Chiew, T.: Why and where: A characterization of data provenance. In: Database Theory—ICDT 2001, pp. 316–330. Springer (2001)
6. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: Proceedings of the 14th international conference on World Wide Web. pp. 613–622. ACM (2005)
7. Ciobanu, G., Horne, R., Sassone, V.: Minimal type inference for linked data consumers. Journal of Logical and Algebraic Methods in Programming (2014)
8. Hayes, P., McBride, B.: Rdf semantics (2004)
9. Heath, T., Bizer, C.: Linked data: Evolving the web into a global data space. Synthesis lectures on the semantic web: theory and technology 1(1), 1–136 (2011)
10. Horrocks, I., Patel-Schneider, P.F., Van Harmelen, F.: From shiq and rdf to owl: The making of a web ontology language. Web semantics: science, services and agents on the World Wide Web 1(1), 7–26 (2003)
11. Klyne, G., Carrol, J.J., Mc Bride, B.: Rdf 1.1 concepts and abstract syntax (2014)
12. Omitola, T., Gibbins, N., Shadbolt, N.: Provenance in linked data integration (2010)
13. Silvello, G.: A methodology for citing linked open data subsets. D-Lib Magazine 21(1), 6 (2015)
14. Simmhan, Y.L., Plale, B., Gannon, D.: A survey of data provenance techniques. Computer Science Department, Indiana University, Bloomington IN 47405 (2005)
15. Zhao, J., Bizer, C., Gil, Y., Missier, P., Sahoo, S.: Provenance requirements for the next version of rdf. Citeseer