Inferring the evolution of ontology axioms from RDF data dynamics

Fatiha Saïs*
LRI (CNRS & Université Paris Sud), Université Paris Saclay
Rue Noetzlin, Bât. Ada Lovelace (650)
Orsay, France 91405
fatiha.sais@lri.fr

Cédric Pruski
Luxembourg Institute of Science and Technology
5, avenue des hauts-fourneaux
Esch-sur-alzette, Luxembourg L-4362
cedric.pruski@list.lu

Marcos Da Silveira
Luxembourg Institute of Science and Technology
5, avenue des hauts-fourneaux
Esch-sur-alzette, Luxembourg L-4362
marcos.dasilveira@list.lu

ABSTRACT

One intrinsic characteristic of knowledge bases (KB), especially those published on the Web of data, is the frequent evolution of their data. Hence, changes that occur may lead to KB inconsistency and therefore, may generate contradictions between the KB facts and the KB axioms. In this paper, we propose an approach that is able to, first, compute and semantically represent the symmetric difference (*diff*) between two different versions of a KB and, second, use the generated *diff* to detect changes (addition and deletion) for the corresponding KB axioms. We further propose an experimental assessment of the approach on exisitng knowledge bases such as DBpedia.

CCS CONCEPTS

• **Computing methodologies** **Knowledge representation and reasoning**; *Ontology engineering*; • **Information systems** *Semantic web description languages*;

KEYWORDS

Ontology axioms, RDF Data, Change inference

ACM Reference format:
Fatiha Saïs, Cédric Pruski, and Marcos Da Silveira. 2017. Inferring the evolution of ontology axioms from RDF data dynamics. In *Proceedings of K-CAP 2017: Knowledge Capture Conference, Austin, TX, USA, December 4–6, 2017 (K-CAP 2017),* 4 pages.
https://doi.org/10.1145/3148011.3154472

1 INTRODUCTION

Today, we are experiencing an unprecedented production of large knowledge bases (KBs), such as DBpedia, YAGO, Wikidata and the Google Knowledge Graph as regards cross domain knowledge. This knowledge is typically expressed in RDF [8], i.e., as triples of the form $\langle Macron, presidentOf, France\rangle$. Some KBs provide an ontology expressed in OWL2 (Web Ontology Language) [12], which describes the vocabulary (the classes and properties) for the RDF facts. The ontologies can also declare logical axioms on the data to express disjunction between classes, the functionality of properties, key constraints and so on.

These new knowledge bases are mostly published as Linked Open Data (LOD, for short) where the KBs are linked one to each other in both ontological level through property and class mappings; and at data level, through an interlinking of resources of different KBs using links representing for most of them identity relation. This is leading to the creation of a global data space containing billions of RDF triples from different domains (e.g., life sciences, geography, government and social networking). One of the intrinsic features of the LOD is its dynamicity. LOD KBs are continuously evolving for different reasons such as information enrichment and correction, new knowledge generation and, every day big data production: *"Every day, we create 2.5 quintillion bytes of data [6]"*.

In the Web of data, when a KB evolves, the changes may concern the ontological level where changes may involve classes, properties, axioms, mappings to other ontologies or semantic annotations [2]. The changes may also concern the instance level where data modifications may affect instance typings, property values, or identity links between instances. Many data integration tasks (e.g. synchronization, data linking or fusion) are directly impacted by the evolution of data, which may lead to inconsistencies. This is mainly the case when ontology axioms are not any more satisfied once data evolves and there exist very few tools [11, 13] that allow to detect changes in the data and to represent them to be interpreted by both human experts and software applications. These changes in the data may also be propagated to the ontological level by deducing changes on the sets of classes and properties as well as on the set of axioms, such as class disjunction, symmetry, reflexivity, transitivity and (inverse) functionality of properties. When axioms are maintained up to date, the consistency of the knowledge base can eventually be ensured.

In this paper we present a novel approach which given two different versions $\mathcal{K}_1$ and $\mathcal{K}_2$ of a knowledge base, aims at detecting inconsistent axioms with respect to changes in the data and a method to currate them. To do so, we assume that the knowledge base $\mathcal{K}_1$ is consistent (i.e., there is an interpretation which entails $\mathcal{K}_1$). However, the proposed approach can be used to discover insconsistencies between the ontology and the KB induced by the evolution of the latter. In this work, we consider several axioms, namely, disjunction between classes and properties, cardinality of properties, inverse of properties as well as symmetry, reflexivity, transitivity, (inverse) functionality of properties. Our approach consists first, in computing the *diff* (i.e., the set of deleted and added triples) between $\mathcal{K}_1$ and $\mathcal{K}_2$. Then, we semantically represent the *diff* using a data evolution ontology that extends the ODE ontology [11]. Finally, thanks to the logical formalization that we propose for expressing the conditions when an axiom can be added or deleted from the ontology, our tool

*Fatiha Saïs (sais@lri.fr)

K-CAP 2017, December 4–6, 2017, Austin, TX, USA

ACM ISBN 978-1-4503-5553-7/17/12...$15.00
https://doi.org/10.1145/3148011.3154472

is able to determine the axioms that should be added or the ones that should be deleted from the ontology. For instance, if two classes A and B are declared as disjoint in $\mathcal{K}_1$ (i.e., there is no instance of A that is also an instance of B and vice versa) and if in $\mathcal{K}_2$ we have an instance i that is instance of A and instance of B then the disjointness axiom between the classes A and B should be removed from the ontology part of $\mathcal{K}_2$. We tested and evaluated the feasability and the efficiency of the approach on real datasets published on the Web of data: AgroVoc[1] and DBpedia[2].

The remainder of the paper is organized as follows: section 2 presents the most related works in knowkedge base enrichment and evolution. In section 3, we give the preliminaries and present our approach for axiom change inferring in section 4. The experiments are presented in section 5. Finally, 6 concludes the paper and give some future direction.

2 RELATED WORK

An extensive work has been conducted on detection, representation and management of the changes at the ontological level (see [16] for a survey). Our approach is rather related to works that studied evolution of datasets at the data (instance) level. The existing approaches [5, 10, 13, 17] may be classified into two categories. First, some works [5, 17] put the stress on low-level changes (i.e. addition and deletion). Approaches described in [5, 17] studied both ontology and data low-level changes. In [17] the authors proposed a new formalism for low-level change detection in RDFS datasets. Franconi et al. [5] discuss low-level changes for propositional Knowledge Bases by providing formal properties as delta uniqueness and change reversibility. Some recent approaches have investigated high-level changes.This is the case of the work of Roussakis et al. [13] which proposes the notion of simple and complex changes in RDF datasets. The complex changes are represented in an ontology of changes and allow to provide a more synthetic and comprehensible representation of the changes. However, they do not declare direct links between the RDF triples and the induced changes, which do not allow to perform specific queries on the modified triples. Papavasileiou et al. [10] proposed a fixed and predefined set of abstract changes without giving abilities to answer queries that combine different kinds of changes.

The problem of axiom change detection in a knowledge base is also related to the problem of ontology learning from data [3]. Some of the existing works are interested in concept definitions [9], others use association rule mining to induce statistical schema [4] while a last family of approaches use DL-Learner framework [1] for light-weight ontology learning. However, none of existing work has considered axiom induction from the data except [14, 15]. In [15] the authors proposed an approach for expressive ontology learning by discovering class disjunction axioms from the data. However, it does not consider other kinds of axioms. As for [14], the authors describe a new framework for possibilistic axiom scoring in order to enrich a knowledge base. However, they only validated their approach by considering only the subClassOf axiom. Furthermore, none of the existing works has considered RDF data changes to detect axiom changes as we propose in this paper.

[1] http://aims.fao.org/vest-registry/vocabularies/agrovoc-multilingual-agricultural-thesaurus
[2] http://wiki.dbpedia.org/

3 PRELIMINARIES

3.1 Knowledge Bases

We consider knowledge bases that are defined by an ontology $\mathcal{O}$, represented in OWL 2 [12], and a set of facts $\mathcal{D}$ represented as a collection of RDF triples [8]. More formally, a knowledge base $\mathcal{B}$ is defined by a couple $(\mathcal{O}, \mathcal{D})$ where:

$\mathcal{O} = (\mathcal{C}, \mathcal{P}, \mathcal{A})$ represents the ontological part of the knowledge base defined by a set of classes $\mathcal{C}$, a set of properties (*owl:DataTypeProperty* and *owl:ObjectProperty*) $\mathcal{P}$ and a set of axioms $\mathcal{A}$ that represents relations such as the subsumption and disjunction between classes, or (inverse) functionality for properties.

$\mathcal{D}$ is a collection of RDF triples $< s, p, o >$, of a subject s that is a URI, a property $p \in \mathcal{P}$ that is also a URI, and an object o that can be either a URI or a Literal. We consider the URIs as belonging to the set of resources $\mathcal{R}$ and the set of Literal values as belonging to the set of basic values $\mathcal{V}$. We note that the triple $< u, rdf{:}type, c >$ can be declared to state that the URI u, that can appear as a subject or an object, is an instance of the class c. It is worthwhile to mention that we do not consider blank nodes in this work.

3.2 Ontology axioms

OWL 2 provides an extensive set of axioms – statements that express what is true in the ontology domain – organized in several types. In this paper we consider axioms that belong to four kinds of axioms out of eight types provided in OWL2, namely, *DataPropertyAxiom*, *ObjectPropertyAxiom*, *ClassAxiom* and *HasKey*. As an example, the type *DataPropertyAxiom* represents the axioms of inverse, symmetry, reflexivity, irreflexivity, transitivity, functionality and inverse functionality of data type property. The detailed specification of OWL 2 axioms and their corresponding formal semantics can be found in [7, 12].

4 DATA EVOLUTION-BASED ONTOLOGY AXIOM CHANGE DETECTION

4.1 RDF data evolution ontology

Data evolution ontology that we have designed allows to represent semantically the changes that can occur in an RDF dataset. It organizes the deleted triples, added triples and stable triples, obtained by a direct comparison of the two versions of the dataset in a hierarchy with different kinds of changes.

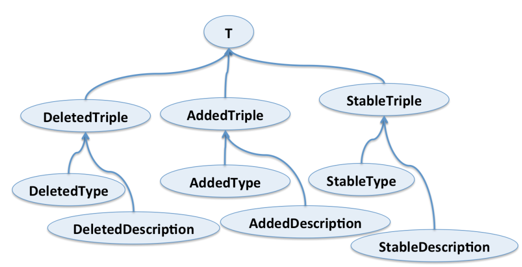

Figure 1: RDF data evolution Ontology

To this end, given a triple $t \in K_{v1}$, if $t \in K_{v2}$ then t is consider as stable, otherwise t will be labelled as deleted. As presented in Figure

1, three classes *AddedTriple*, *DeletedTriple* and *StableTriple* are put in the top of the ontology. In this ontology, each class is specialized into two subclasses: one is about changes affecting the instance types (*rdf:type*) and the other one deals with instance description changes (e.g. new property instance).

We note that the classes *AddedTriple*, *DeletedTriple* and *StableTriple* are pairwise disjoint and the subclasses of each of these classes are also pairwise disjoint. Finally, these classes are used to type the triples that are involved in data changes using the named graph representation.

4.2 Conditions for axiom change detection

To detect axiom deletion and addition, we formalised the conditions under whom axioms' changes can be detected. To do so, we used combination of set theory and First Order Logic (FOL) and predicates that are represented a the data evolution ontology to refer to the change type of each triple that is involved in the data evolution.

In what follows we consider the set $\{t_1, t_2, \ldots, t_n\}$ of RDF triples that are organized using the different classes of the data evolution ontology. In particular, to detect the axioms' changes we use the triples that are classified in the leaf level of the ontology, namely, *AddedType* (AT), *AddedDescription* (AD), *DeletedType* (DT), *DeletedDescription* (DD), *StableType* (ST) and *StableDescription* (SD). In the sequel, expressing that a triple t_k is of type T, is denoted by $t_k \in T$ in the data evolution ontology. Note that, we also consider in our method all elements linked with *owl:sameAs*, but for sake of readability we use only one variable name (e.g., $x = \{u \mid u = u_x$ or $\exists t_1 =< u\ owl : SameAs\ u_x >\}$

Axiom Deletion

Deletion of class disjunction axiom. The axiom which represents that two classes c_1 and c_2 are disjoint should be removed from the ontology when the following condition is fulfilled: $\exists t_1 =< x\ rdf{:}type\ c_1 >\in AT$ and $\exists t_2 =< x\ rdf{:}type\ c_2 >\in (AT \cup ST)$

Deletion of class equivalence axiom. The axiom which represents that two classes c_1 and c_2 are equivalent should be removed from the ontology when the following condition is fulfiled: $\exists t_1 =< x\ rdf{:}type\ c_1 >\in AT$ and $\nexists t_2 =< x\ rdf{:}type\ c_2 >\in (AT \cup ST)$

Deletion of inverse property axiom. The axiom which represents that a property p_1 is the inverse of a property p_2 should be removed from the ontology when the following condition is fulfilled: $\exists t_1 =< x\ p_1\ y >\in DD$ and $\exists t_2 =< y\ p_2\ x >\notin DD$

Deletion of symmetry of a property axiom. The axiom which represents that a property p is symmetric should be removed from the ontology when the following condition is fulfilled: $\exists t_1 =< x\ p\ y >\in DD$ and $\exists t_2 =< y\ p\ x >\notin DD$

Delete of irreflexivity of a property axiom. The axiom which represents that a property p should be removed from the ontology when the following condition is fulfilled:
$\exists t =< x\ p\ x >\in AD$

Deletion of transitivity of a property axiom. The axiom which represents that a property p is transitive should be removed from the ontology when the following condition is fulfilled:
$\exists t_1 =< x\ p\ z >\notin DD$ and ($\exists t_2 =< y\ p\ z >\in DD$ or $\exists t_3 =< x\ p\ y >\in DD$) OR $\exists t_1 =< x\ p\ z >\in IDI$ and $\exists t_2 =< x\ p\ y >\notin DD$ and $\exists t_3 =< y\ p\ z >\notin DD$

Deletion of functionality of a property axiom. The axiom which represents that a property p is functional should be removed from the ontology when the following condition is fulfilled: $\exists t_1 =< x\ p\ y >\in IDE$ and $\exists t_2 =< x\ p\ z >\notin IDI$ and $y \neq z$

Deletion of inverse functionality of a property axiom. The axiom which represents that a property p is inverse functional should be removed from the ontology when the following condition is fulfilled: $\exists t_1 =< y\ p\ x >\in AD$ and $\exists t_2 =< z\ p\ x >\in (SD \cup AD)$ and $y \neq z$

Deletion of key axiom. The key axiom which represents that a named instance has the inverse functional property p should be removed from the ontology when the following condition is fulfilled: $\exists t_1 =< y\ p\ x >\in AD$ and $\exists t_3 =< z\ p\ x >\in (SD \cup AD)$ and ($\nexists t_1 =< x\ rdf{:}type\ c >\in (SD \cup AD)$ or $y \neq z$)

Deletion of cardinality constraint of a property axiom. The axiom which represents that a property p has a cardinality restriction Card(p,n,rest) should be removed from the ontology when the following condition is fulfilled: Count (t_{card}) rest n, where Count(t_{card}) is a function that gives the number of triples from the set t_{card}, and $t_{card} \subseteq T$ given by $t_{card} = \{t \mid \exists t = < x\ p\ y >\in (SD \cup AD)\}$

For space reason, we will not give the detailed formalization for the axiom addition. It is analogous to the axiom deletion formalization except for some intricate cases, like key addition.

4.3 Axiom change detection algorithm

To detect the axiom changes, we developed two main algorithms *Axioms-Add* and *Axioms-Delete* that are used to detect whether an axiom should be deleted or added to the ontology. To compute the axioms changes that hold between two versions K_1 and K_2 of the KB, both algorithms take as input the data evolution ontology populated with the symmetric difference between K_1 and K_2 and provide as output the set of added axioms and the set of deleted axioms respectively.

The use of the data evolution otology allows the algorithms *Axioms-Add* and *Axioms-Delete* to consider only the part of the data that is relevant for the change detection, i.e., prevent from recomputing the axioms each time data changes. Hence, to detect an axiom addition and deletion, it sufficies to check the corresponding change conditions, presented bellow (see subsection 4.2) and translated into SPARQL queries. In the following is an extract of the SPARQL query for property semmetry checking:

```
PREFIX ode:<http://www.semanticweb.org/sais/ontologies/
2017/4/ode-ontology-1#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl:<http://www.w3.org/2002/07/owl#>

ASK {{{?s1 rdf:type ode:AddedProperty . ?s1 rdf:object ins1 .
?s1 rdf:predicate rel . ?s1 rdf:subject ins2} UNION
{?s5 rdf:type ode:IDE . ?s5 rdf:object ins1 .
?s5 rdf:predicate rel . ?s5 rdf:subject ins2} UNION
{?s2 rdf:type ode:StableProperty . ?s2 rdf:object ins1 .
?s2 rdf:predicate . ?s2 rdf:subject ins2}} .
{?s2 rdf:type ode:AddedProperty . ?s2 rdf:object ins2 .
?s2 rdf:predicate rel . ?s2 rdf:subject ins1} UNION
{?s3 rdf:type ode:IDE . ?s3 rdf:object ins2 .
?s3 rdf:predicate rel .  ?s3 rdf:subject ins1} UNION
{?s4 rdf:type ode:StableProperty .
?s4 rdf:object ins2 . ?s4 rdf:predicate rel .
```

```
?s4 rdf:subject ins1}}
```

where *ins1*, *ins2* and *rel* are artificial variables used in the runtime of the algorithm.

5 PRELIMINARY RESULTS

Through the various sets of experiments we have performed, our main goal was to evaluate the feasibility of the proposed approach as well as its capability to identify inconsistencies between the ontology that described the datasets and the RDF statements contained in the datasets.

In our initial assessment, we have used successive versions of DBPedia and AGROVOC. Regarding DBpedia, we restrict our experiments to versions 2016-04 and 2016-10 of the Persondata RDF dataset[3] while we have used version 2017-07 and 2017-08 of the AGROVOC dataset[4]. Table 1 hereafter shows the amount of RDF triples that have been added, deleted and those that remain stable in the *Diff*.

Table 1: Amount of added, deleted and stable triples in the datasets

	Added triples	Deleted triples	Stable triples	Total
DBPedia	8 186 431	3 943 370	2 123 663	14 253 464
AGROVOC	9351	1220	6 254 046	6 264 617

Once the Diff is computed (i.e. when all nodes composing the RDF triples that have been added and deleted are identified as being new classes, properties or instances w.r.t. the ODE ontology), we analyze, via SPARQL queries[5], if the axioms of the associated ontology are valid or not. The evoked queries are the SPARQL adaptations of the logic rules presented throughout this paper.

The analysis of the results reveals that no axioms of the used ontologies are inconsistent with the data however, as our approach exploits a subset (i.e. added and deleted triples) of the evolved dataset, the computation time is drasticaly reduced. Moreover, we could also detect some potential inconsistencies in the data. This is the case, for instance, in the "Persondata" sets of DBPedia when evaluating the reflexivity of some properties we found the following assertion in the dataset: <Kanthapuram_A._P._Aboobacker_Musalyar> <birthPlace> <Kanthapuram_A._P._Aboobacker_Musalyar> which is a clear inconsistency regarding the data.

Further experiments are ongoing to evaluate the gain, in terms of computation time, in comparison with well known reasoners to be able to quantity the added value of our method.

6 CONCLUSION

In this paper we presented a new approach for axiom change detection based on RDF data evolution representation. We designed a RDF data evolution ontology to semantically represent the data evolution (i.e., added triples, deleted triples and stable triples). Using predicates that are represented on the evolution ontology we proposed a formalisation of the conditions when the ontology axioms (e.g. class disjointness, functionality of properties) can be deleted. Therefore, our algorithm of axiom change detection, for each kind of axioms, exploits only the needed part of the data. The experiments on several versions of a cross domain dataset DBpedia and a domain-specifc AgroVoc have shown the feasability and the efficiency of the approach. As future work, we plan to test our approach on bigger datasets of different domains and evaluate qualitatively the results. In order to take into account possible errors and exceptions that can appear in the datasets, we plan also to extend our approach to be able to give scores for the axioms.

[3]Dbpedia person: http://wiki.dbpedia.org/datasets
[4]AGROVOC: https://aims-fao.atlassian.net/wiki/spaces/AGV/pages/2949126/Releases
[5]SPARQL queries are available at: http://www.lri.fr/~sais/axiom-evolution/queries

REFERENCES

[1] Lorenz Bühmann and Jens Lehmann. 2012. Universal OWL Axiom Enrichment for Large Knowledge Bases. In *Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*. Springer-Verlag, Berlin, Heidelberg, 57–71. https://doi.org/10.1007/978-3-642-33876-2_8

[2] Silvio Domingos Cardoso, Cédric Pruski, Marcos Da Silveira, Ying-Chi Lin, Anika Groß, Erhard Rahm, and Chantal Reynaud-Delaître. 2016. Leveraging the impact of ontology evolution on semantic annotations. In *Knowledge Engineering and Knowledge Management: 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings 20*. Springer, 68–82.

[3] P. Cimiano, A. MÃď dche, S. Staab, and J. VÃűlker. 2009. Ontology Learning. In *Handbook on Ontologies* (2nd revised edition ed.), S. Staab and R. Studer (Eds.). Springer, 245–267. http://www.uni-koblenz.de/~staab/Research/Publications/2009/handbookEdition2/ontology-learning-handbook2.pdf

[4] Daniel Fleischhacker, Johanna Völker, and Heiner Stuckenschmidt. 2012. *Mining RDF Data for Property Axioms*. Springer Berlin Heidelberg, Berlin, Heidelberg, 718–735. https://doi.org/10.1007/978-3-642-33615-7_18

[5] Enrico Franconi, Thomas Meyer, and Ivan Varzinczak. 2010. Semantic Diff as the Basis for Knowledge Base Versioning. In *NMR*.

[6] Amir Gandomi and Murtaza Haider. 2015. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management* 35, 2 (2015), 137 – 144. https://doi.org/10.1016/j.ijinfomgt.2014.10.007

[7] Ian Horrocks, Bijan Parsia, and Uli Sattler. 2012. OWL 2 Web Ontology Language Direct Semantics (Second Edition). (2012).

[8] F. Manola, E. Miller, and B. McBride. 2004. RDF primer. *W3C recommendation* 10 (2004), 1–107.

[9] José Eduardo Ochoa-Luna, Kate Revoredo, and Fábio Gagliardi Cozman. 2011. *Learning Probabilistic Description Logics: A Framework and Algorithms*. Springer Berlin Heidelberg, Berlin, Heidelberg, 28–39. https://doi.org/10.1007/978-3-642-25324-9_3

[10] Vicky Papavasileiou, Giorgos Flouris, Irini Fundulaki, Dimitris Kotzinos, and Vassilis Christophides. 2013. High-level Change Detection in RDF(S) KBs. *ACM Trans. Database Syst.* 38, 1, Article 1 (April 2013), 42 pages. https://doi.org/10.1145/2445583.2445584

[11] Nathalie Pernelle, Fatiha Saïs, Daniel Mercier, and Sujeeban Thuraisamy. 2016. RDF data evolution: automatic detection and semantic representation of changes. In *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016, Leipzig, Germany, September 12-15, 2016.* http://ceur-ws.org/Vol-1695/paper29.pdf

[12] W3C Recommendation. 27 October 2009. OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax. In *http://www.w3.org/TR/owl2-syntax/*, Parsia B. Motik B., Patel-Schneider P. F. (Ed.). W3C.

[13] Yannis Roussakis, Ioannis Chrysakis, Kostas Stefanidis, Giorgos Flouris, and Yannis Stavrakas. 2015. A Flexible Framework for Understanding the Dynamics of Evolving RDF Datasets. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*. 495–512. https://doi.org/10.1007/978-3-319-25007-6_29

[14] Andrea G. B. Tettamanzi, Catherine Faron-Zucker, and Fabien Gandon. 2014. *Testing OWL Axioms against RDF Facts: A Possibilistic Approach*. Springer International Publishing, Cham, 519–530. https://doi.org/10.1007/978-3-319-13704-9_39

[15] Johanna Völker, Peter Haase, and Pascal Hitzler. 2008. Learning Expressive Ontologies. In *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. 45–69. http://www.booksonline.iospress.nl/Content/View.aspx?piid=8216

[16] Fouad Zablith, Grigoris Antoniou, Mathieu d'Aquin, Giorgos Flouris, Haridimos Kondylakis, Enrico Motta, Dimitris Plexousakis, and Marta Sabou. 2015. Ontology evolution: a process-centric survey. *Knowledge Eng. Review* 30, 1 (2015), 45–75. https://doi.org/10.1017/S0269888913000349

[17] Dimitris Zeginis, Yannis Tzitzikas, and Vassilis Christophides. 2011. On Computing Deltas of RDF/S Knowledge Bases. *ACM Trans. Web* 5, 3, Article 14 (July 2011), 36 pages. https://doi.org/10.1145/1993053.1993056