Using formal concept analysis for checking the structure of an ontology in LOD: the example of DBpedia

Pierre Monnin, Mario Lezoche, Amedeo Napoli, Adrien Coulet

HAL Id: hal-01511909

https://hal.inria.fr/hal-01511909

Submitted on 21 Apr 2017

Using Formal Concept Analysis for Checking the Structure of an Ontology in LOD: the Example of DBpedia

Pierre Monnin[1], Mario Lezoche[2], Amedeo Napoli[1], and Adrien Coulet[1]

[1] LORIA (CNRS, Inria NGE, Université de Lorraine), Vandœuvre-lès-Nancy, France,
[2] CRAN (CNRS, Université de Lorraine), Vandœuvre-lès-Nancy, France

Abstract. Linked Open Data (LOD) constitute a large and growing collection of inter-domain data sets. LOD are represented as RDF graphs that allow interlinking with ontologies, facilitating data integration, knowledge engineering and in a certain sense knowledge discovery. However, ontologies associated with LOD are of different quality and not necessarily adapted to all data sets under study. In this paper, we propose an original approach, based on Formal Concept Analysis (FCA), which builds an optimal lattice-based structure for classifying RDF resources w.r.t. their predicates. We introduce the notion of lattice annotation, which enables comparing our classification with an ontology schema, to confirm subsumption axioms or suggest new ones. We conducted experiments on the DBpedia data set and its domain ontologies, DBpedia Ontology and YAGO. Results show that our approach is well-founded and illustrates the ability of FCA to guide a possible structuring of LOD.

Keywords: Linked Open Data, Formal Concept Analysis, classification, ontology engineering

1 Introduction

Linked Open Data (LOD) are resulting from a community effort for building a web of data, where data resources may be freely accessed and interpreted by human or software agents for various problem-solving activities. LOD are composed of a large and growing collection of data sets represented within Semantic Web standards that include the use of RDF (Resource Description Framework) and URIs (Uniform Resource Identifiers). In LOD, resources are identified by a URI and can represent entities of the real world (*e.g.*, persons, organizations, places). RDF statements link resources to other resources or to literals (*e.g.* strings, numbers) using properties, also called predicates. Predicates can be used to link resources from the same data set, from different data sets, or from a data set and an ontology [3]. Here, an ontology is a formal representation of a particular domain consisting of classes and relationships between classes. Indeed, resources of a data set may be typed by ontology classes, resulting in class instantiation.

In this paper, we consider DBpedia, an RDF data set of the LOD, built from Wikipedia [12]. Wikipedia consists of a very large number of articles, mostly

composed of plain text, but that may also include structured data as ones in *infoboxes*. For example, in the Wikipedia article about the French philosopher Voltaire[3], the infobox includes his birth date, his nationality and his occupation. The DBpedia project aims at extracting this structured content and making it available respecting Semantic Web standards. In DBpedia, each Wikipedia article is represented as a *DBpedia page*. Each page can instantiate multiple classes from several ontologies of the DBpedia data set.

Because of their rapid growth and popularity, LOD data sets and their associated ontologies are of various quality and completeness [16]. Consequently, several research papers aim at correcting and completing LOD data sets and ontologies [14]. Some of the proposed approaches extract association rules that express relations between variables, *e.g.*, implications (see [13] for more details on association rule mining with FCA). For example, D'Amato *et al.* mine both data and an associated ontology to learn assertional axioms (*e.g.*, class instantiations) using association rule mining [5]. AMIE and AMIE+ use Inductive Logic Programming to mine association rules over large knowledge bases composing the LOD [6,7]. Alternatively, Alam *et al.* use FCA to mine DBpedia and obtain implications, subsequently transformed into definitions of ontology classes [1].

In this paper, we compare the structure of an ontology with a hierarchical structure built from RDF resources and the predicates they are subject of. Thanks to this comparison, we are able to check if the structure of the ontology is in agreement with regularities in RDF data. For this purpose, we use Formal Concept Analysis (FCA), which is a mathematical framework well adapted to data analysis and knowledge engineering purposes [2,4]. First, we propose applying standard FCA to classify RDF resources within a concept lattice w.r.t. their predicates. Second, we propose to extend FCA with the definition of the *annotation* of a concept, which associates concepts of the lattice with classes of an ontology. Finally, because they have similar properties to intents, annotations yield rules that we use for checking axioms in ontologies. In the following, we assume that the reader is familiar with the basics of FCA that can be found in [9].

The paper is organized as follows. Section 2 introduces LOD, ontologies and DBpedia. Section 3 presents the method we propose for checking the structure of an ontology. Section 4 details an experimentation with DBpedia data. Finally, we conclude in Section 5 and 6 with a discussion on how FCA could contribute to the structuring of LOD.

2 Preliminaries

2.1 Linked Open Data (LOD) and Ontologies

Semantic web standards facilitate the publication of LOD online and their connection to existing data sets [3]. Indeed, LOD are represented in the form of

---

[3] Wikipedia article about Voltaire: `https://en.wikipedia.org/wiki/Voltaire`

graphs encoded using RDF. Atomic elements of an RDF graph are triples denoted by:

$$\langle subject, predicate, object \rangle \in (U \cup B) \times (U \cup B) \times (U \cup B \cup L)$$

where U is the set of URIs, L is the set of literals and B represent blank nodes. We note that B is absent in DBpedia.

Ontologies consist of classes and relationships between these classes [10]. In this paper, we are interested in the subsumption relation. It is a transitive relation denoted by $\sqsubseteq$, where $c \sqsubseteq d$ means that c is a subclass of d. LOD resources can be linked to several ontologies, *i.e.*, a subject can instantiate one or several classes from one or several ontologies. The semantics of the subsumption relation as well as the semantics of the instantiation of a class by a resource depend on the representation formalism of the ontology. Indeed, taxonomies expressed with SKOS (Simple Knowledge Organization System) use the `skos:broader` predicate to express subsumption relations and `dcterms:subject` to express instantiations. Alternatively, ontologies using RDFS (RDF Schema) express subsumption relations using the `rdfs:subClassOf` predicate and instantiations using the `rdf:type` predicate. For example, the DBpedia page `Voltaire` is typed with the class `Writer` from the DBpedia Ontology with the following triple:

⟨`http://dbpedia.org/page/Voltaire`, `rdf:type`,

`http://dbpedia.org/ontology/Writer`⟩

In this paper, we consider that we have at our disposal an abstract ontology $\mathcal{O}$ using `abstract:subClassOf` to express subsumption relations and `abstract:type` to express instantiations. We denote $\mathcal{C}_\mathcal{O}$ the set of classes of $\mathcal{O}$. We define the *type* of a resource e as the set of classes of $\mathcal{O}$ that e instantiates. Formally,

$$\mathrm{type}(e) = \{c \in \mathcal{C}_\mathcal{O} \mid \langle e, \texttt{abstract:type}, c \rangle\}$$

Accordingly to the definition of the subsumption, an instance of a class c is also an instance of all the classes subsuming c in $\mathcal{C}_\mathcal{O}$. Thus, we define the *extended type* of e as the whole set of superclasses of the classes of type(e). Formally,

$$\mathrm{extdtype}(e) = \mathrm{type}(e) \cup \{d \in \mathcal{C}_\mathcal{O} \mid \exists c \in \mathrm{type}(e),\ c \sqsubseteq d\}$$

2.2 DBpedia

Our study focuses on DBpedia as it is a large LOD data set and as it is associated with ontologies. Particularly, we are interested in DBpedia pages and their classifications w.r.t. available ontologies in DBpedia.

Each DBpedia page is a RDF resource representing an article from Wikipedia. DBpedia pages instantiate classes from multiple ontologies encoded with various languages such as SKOS or RDFS. We consider, in our experiments, the DBpedia Ontology [12] and YAGO [15], which are both encoded in RDFS. The DBpedia Ontology has been manually created based on the most common structured contents available in Wikipedia. YAGO facts have been automatically extracted from Wikipedia and WordNet and their accuracy has been manually evaluated.

3 Bridging a Concept Lattice and an Ontology

Although our experiments were performed on DBpedia, we explain our approach considering the example of an abstract RDF data set. It is composed of the RDF triples from Table 1 containing 4 different resources that are subjects of various triples. We also consider an abstract ontology $\mathcal{O}$ (Figure 2) composed of *(i)* classes that are instantiated by the subjects of the triples and *(ii)* subsumption relationships between these classes.

3.1 Construction of the Concept Lattice

We build a formal context (G, M, I), where formal objects G are subjects of triples in our data set and attributes M are predicates of the same triples. The incidence relation I indicates that there exists at least one triple where a subject and a predicate appear simultaneously. Here, only two elements of the triples are considered: the subject and the predicate, while the object is not taken into account. Then, we apply standard FCA on this context to build a concept lattice. For example, Table 1 shows RDF triples and the associated formal context. The resulting formal lattice is displayed in Figure 1.

Table 1. Example of RDF triples and the associated formal context. Triples are represented using the Turtle syntax.

```
e₁   rdf:type      k₁, k₂ .
e₁      pred₁         o₁ .
e₁      pred₂         o₂ .
e₂   rdf:type   k₁, k₂, k₄, k₅ .
e₂      pred₁         o₃ .
e₂      pred₂         o₄ .
e₂      pred₃         o₅ .
e₃   rdf:type      k₁, k₂ .
e₃      pred₁         o₆ .
e₄   rdf:type    k₁, k₂, k₅ .
e₄      pred₂         o₇ .
e₄      pred₃         o₈ .
```

	`rdf:type`	$pred_1$	$pred_2$	$pred_3$
e_1	×	×	×	
e_2	×	×	×	×
e_3	×	×		
e_4	×		×	×

3.2 Annotation of Formal Concepts

In the lattice in Figure 1, each extent contains subjects and each intent contains predicates. To compare the lattice with the ontology $\mathcal{O}$, we need to "link" formal concepts with ontology classes. To do so, given $A \subseteq G$ and $K \subseteq \mathcal{C}_\mathcal{O}$, we define two new dual derivation operators denoted by $(\cdot)^\diamond : 2^G \to 2^{\mathcal{C}_\mathcal{O}}$ and $(\cdot)^\diamond : 2^{\mathcal{C}_\mathcal{O}} \to 2^G$ such as:

$$A^\diamond = \bigcap_{e \in A} \text{extdtype}(e) \quad \text{and} \quad K^\diamond = \{e \in G \mid K \subseteq \text{extdtype}(e)\}$$
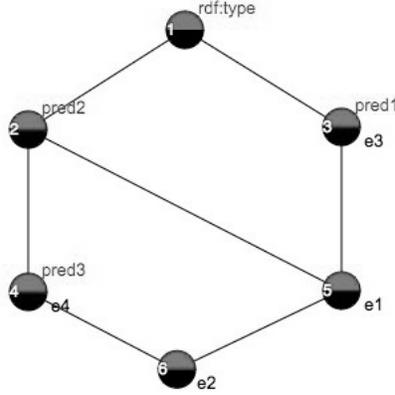
Fig. 1. Line diagram representing the concept lattice built from the formal context in Table 1. The lattice is displayed using the reduced notation. Following this notation, formal objects, depicted in black, are associated with one concept and, implicitly, with all its superconcepts. Attributes, in grey, are associated with one concept and, implicitly, with all its subconcepts. Formal concepts are arbitrarily numbered from 1 to 6.

$A^\diamond$ contains the ontology classes shared by the extended types of all the subjects in A. $K^\diamond$ represents the set of subjects having all the ontology classes of K in their extended type. We prove in Appendix that $(\cdot)^{\diamond\diamond} : 2^{\mathcal{C}_\mathcal{O}} \to 2^{\mathcal{C}_\mathcal{O}}$ and $(\cdot)^{\diamond\diamond} : 2^G \to 2^G$ are *closure operators*.

Given a formal concept (A, B), we call $A^\diamond$ the *annotation* of the concept and we define the triple $(A, B, A^\diamond)$ as an *annotated concept*. For example, consider the concept 4 $(\{e_2, e_4\}, \{\texttt{rdf:type}, \texttt{pred}_2, \texttt{pred}_3\})$ in Figure 1. Based on the ontology in Figure 2, we have $\{e_2, e_4\}^\diamond = \{k_1, k_2, k_3, k_5\}$. So, the annotated concept is $(\{e_2, e_4\}, \{\texttt{rdf:type}, \texttt{pred}_2, \texttt{pred}_3\}, \{k_1, k_2, k_3, k_5\})$. Accordingly, we define the *annotated lattice* as the lattice where each concept is replaced by its corresponding annotated concept.

Given two concepts such as $(A_1, B_1) \leqslant (A_2, B_2)$. As $A_1 \subseteq A_2$, we have $A_2^\diamond \subseteq A_1^\diamond$ (see Equation (1) in Appendix). Therefore, we also extend the reduced notation to represent an annotated lattice: extents and intents are depicted as usual and annotations are depicted showing only the new classes that are not already in the annotations of the superconcepts. For example, on the annotated lattice in Figure 3, the annotation of the concept 4 shows only k_5 because k_1, k_2 and k_3 are already in the annotation of its superconcepts.

3.3 Comparing Rules Extracted in the Lattice and Axioms in the Ontology

In the annotated lattice in Figure 3, extents contain subjects, intents contain predicates and annotations contain classes of $\mathcal{O}$. To check the structure of the
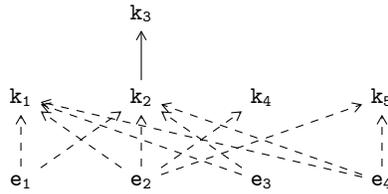
Fig. 2. Example of ontology classes being instantiated by subjects. Dotted arrows represent instantiations whereas solid arrows represent subsumption relations. For example, e_1 instantiates k_1 and k_2, and k_2 is subsumed by k_3.
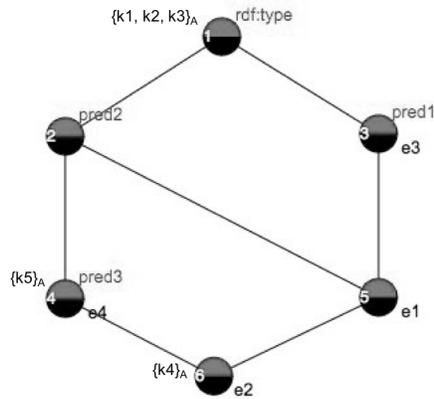

Fig. 3. Line diagram representing the annotated lattice based on the concept lattice in Figure 1 and the ontology in Figure 2. It is displayed using our extension of the reduced notation. Formal objects are depicted in black, attributes in grey and annotations are depicted by $\{\cdot\}_A$.

ontology, in other words, to check the subsumption axioms existing between classes of $\mathcal{O}$, we have to extract rules from the lattice that can be considered similar to subsumption axioms. To do so, we consider each pair composed of an annotated concept $(A, B, A^\diamond)$ and one of its covering concept $(E, F, E^\diamond)$, *i.e.* one of its direct superconcepts, such as the pair composed of concepts 6 and 4 in Figure 3. Similarly to Figure 3, we denote $A_A^\diamond = \{x_1, x_2, \ldots, x_p\}$ and $E_A^\diamond = \{y_1, y_2, \ldots, y_q\}$ the reduced notation of the annotations $A^\diamond$ and $E^\diamond$.

Then, we consider each pair of ontology classes (x_i, y_j) from $A_A^\diamond \times E_A^\diamond$. Considering concepts 6 and 4, the only pair that can be considered is (k_4, k_5). Because $A \subseteq E$ and $E^\diamond \subseteq A^\diamond$, we can say that all subjects typed by x_i are also typed by y_j. Moreover, y_j can type subjects that are not typed by x_i. Hence, x_i can be viewed as "more specialized" than y_j. This can be interpreted as follows: the annotated lattice "suggests" x_i as a subclass of y_j. Because we use the cover relation, we consider this suggestion as a direct subclass suggestion. Here, k_4 is suggested as a direct subclass of k_5.

This rule extracted from the lattice is then compared with the subsumption axioms defined in $\mathcal{O}$:

- If x_i is already explicitly declared as a subclass of y_j, we call this rule a *confirmed axiom*.
- If x_i is an indirect subclass of y_j, *i.e.* x_i is not explicitly stated as a subclass of y_j but it can be inferred from $\mathcal{O}$ thanks to the transitivity of the subsumption relation, we call the rule an *inferable axiom*.
- Otherwise, the rule does not exist in the ontology and we consider it as a potential *new axiom*, proposed to improve the ontology.

In our example, according to the ontology in Figure 2, there is no link existing between k_4 and k_5. Thus, $k_4 \sqsubseteq k_5$ is a new axiom proposal.

If we consider the pair composed of concepts 6 and 5 in Figure 3, we notice that the reduced notation of the annotation of concept 5 is empty. Thus, no rule can be extracted from this pair of concepts. Similarly, no rule involving k_1, k_2 and k_3 can be extracted with the current approach because the reduced notation of the annotations of concepts 2 and 3 are empty.

4 Experimentation

We experimented this approach, classifying 904 pages selected from the DBpedia 2014 data set that describe persons who died between 01/01/2000 and 07/01/2000 (included). The resulting lattice consisted of 15,234 concepts. Axioms from the DBpedia Ontology (11) and YAGO (199) were retrieved among the rules and, therefore, were considered as confirmed axioms. For example, the DBpedia Ontology class `Boxer` was found as a subclass of `Athlete`. Inferable axioms (2,250) as well as new axioms (1,372) were only found for YAGO. This may be due to the small size of the DBpedia Ontology which only contains 683 classes. Therefore, pages have more similar extended types than with a more

fine-grained ontology such as YAGO. Consequently, annotations of formal concepts are also similar, leading to empty reduced notation of annotations from which no rule is extracted (see the example with concepts 5 and 6 in Figure 3). For example, the lattice we obtained from our subset of DBpedia pages suggested to add `BostonUniversityAlumni` as a subclass of `Scholar110557854`. Nevertheless, this subsumption could be inferred from the ontology. Another example was the suggestion of `FilipinoChildActors` as a subclass of `FilipinoActors`. It was neither explicitly stated in the ontology nor inferable and seemed to make sense in a general case.

5 Discussion

Regarding the results of the experiments, when classifying RDF subjects w.r.t. their related predicates, we are able to suggest direct subclass relations between ontology classes typing these subjects. Some of the suggestions already exist as direct subclass relations in the considered ontology. Thus, predicates allow to classify subjects in a lattice whose structure is similar to some parts of the existing ontology. Consequently, it seems that some predicates applied to subjects are specific of the classes typing these subjects. Therefore, predicates could be considered as "indicators" of the classification of subjects by ontology classes. Some of the suggested axioms indirectly exist in the ontology, meaning that the structure of the lattice is not as fine-grained as the ontology. This may be due to the fact that predicates applied to subjects are not the only indicators of their classification by classes of an ontology. In the annotated lattice in Figure 3, we also notice that the reduced notation of the annotation is empty for concepts 2, 3 and 5 because these concepts are not closed w.r.t. the $(\cdot)^{\diamond\diamond}$ operator. Therefore, we do not extract rules from them, in addition the classes k_1, k_2 and k_3 never appear in rules extracted from the lattice.

As future work, we will consider additional features in classification to provide more granularity. For example, the formal context could be built using predicates, $(predicate, \text{extdtype}(object))$ and $(predicate, object)$ pairs as attributes. Pattern Structures [8] could be used on extended types as there is a partial order on classes of an ontology. Because $(\cdot)^{\diamond\diamond}$ is a closure operator, we could also use the triadic approach to FCA [11] to model the three sets that describe each of our concepts (extents, intents and annotations). Another perspective of the current work is to reduce the lattice by keeping only the closed concepts relatively to the $(\cdot)^{\diamond\diamond}$ operator. This will increase the number of suggestions. For example, in Figure 3, removing concepts 2, 3 and 5 leads to suggest k_5 and k_4 as subclasses of k_1, k_2 and k_3. Finally, in the case of subjects typed by two (or more) ontologies, two annotations can be computed for each concept of the lattice, providing potential relationships between independent ontologies. In this case, the lattice can be seen as a pivot structure for ontology matching experiments.

6 Conclusion

In this paper, we apply FCA to classify RDF subjects w.r.t. their related predicates in a concept lattice. A method is proposed to annotate this lattice with ontology classes used to type these subjects. Based on the structure of the resulting lattice, subsumption axioms between the ontology classes are suggested and then compared with existing axioms in the ontology. Running this method on a subset of pages from DBpedia and considering two associated reference ontologies (DBpedia Ontology and YAGO), we are able to suggest axioms that already exist. This means that the structure of the annotated lattice is similar to some parts of the existing ontologies. Consequently, predicates applied to pages can be considered as indicators of classes instantiated by these pages. New axioms are also proposed. One of the next challenges resides in performing a qualitative study of these new axioms.

Appendix: Proving that $(\cdot)^{\diamond\diamond}$ is a Closure Operator

An operator on a partial ordered set is a closure operator if it is monotonous, extensive and idempotent. Firstly, let us prove that:

$$X_1 \subseteq X_2 \Rightarrow X_2^{\diamond} \subseteq X_1^{\diamond} \tag{1}$$

To do so, let's consider $E_1 \subseteq G$ and $E_2 \subseteq G$ such as $E_1 \subseteq E_2$ and, dually, $K_1 \subseteq \mathcal{C}_{\mathcal{O}}$ and $K_2 \subseteq \mathcal{C}_{\mathcal{O}}$ such as $K_1 \subseteq K_2$. We have:

$$E_2^{\diamond} = \bigcap_{e \in E_2} \text{extdtype}(e) = \left(\cap_{e \in E_1}\text{extdtype}(e)\right)\bigcap\left(\cap_{e \in E_2 \setminus E_1}\text{extdtype}(e)\right)$$

$$= E_1^{\diamond}\bigcap\left(\cap_{e \in E_2 \setminus E_1}\text{extdtype}(e)\right) \subseteq E_1^{\diamond}$$

$$K_2^{\diamond} = \{e \in G \mid K_2 \subseteq \text{extdtype}(e)\}$$

$$= \{e \in G \mid K_1 \subseteq \text{extdtype}(e) \;\wedge\; K_2 \setminus K_1 \subseteq \text{extdtype}(e)\}$$

$$= \{e \in G \mid K_1 \subseteq \text{extdtype}(e)\}\bigcap\{e \in G \mid K_2 \setminus K_1 \subseteq \text{extdtype}(e)\}$$

$$= K_1^{\diamond}\bigcap\{e \in G \mid K_2 \setminus K_1 \subseteq \text{extdtype}(e)\} \subseteq K_1^{\diamond}$$

Monotonicity. Considering $X_1 \subseteq X_2$. Because of (1), we have $X_2^{\diamond} \subseteq X_1^{\diamond}$ and then $X_1^{\diamond\diamond} \subseteq X_2^{\diamond\diamond}$.

Extensivity. Considering $E \subseteq G$, we have $E^{\diamond} = \bigcap_{e \in E} \text{extdtype}(e)$. So, $E^{\diamond\diamond} = \{f \in G \mid \bigcap_{e \in E} \text{extdtype}(e) \subseteq \text{extdtype}(f)\}$. Therefore, $E \subseteq E^{\diamond\diamond}$. Dually, considering $K \subseteq \mathcal{C}_{\mathcal{O}}$, we have $K^{\diamond} = \{e \in G \mid K \subseteq \text{extdtype}(e)\}$. So, $K^{\diamond\diamond} = \bigcap_{f \in \{e \in G \mid K \subseteq \text{extdtype}(e)\}} \text{extdtype}(f)$. Therefore , $K \subseteq K^{\diamond\diamond}$.

Idempotence. Because of the extensivity, we know that $X \subseteq X^{\diamond\diamond}$. Therefore, because of (1), $X^{\diamond\diamond\diamond} \subseteq X^{\diamond}$ and $X^{\diamond\diamond} \subseteq X^{\diamond\diamond\diamond\diamond}$. Because of the extensivity, $X^{\diamond} \subseteq X^{\diamond\diamond\diamond}$. So because of (1), $X^{\diamond\diamond\diamond\diamond} \subseteq X^{\diamond\diamond}$. Consequently, $X^{\diamond\diamond} = X^{\diamond\diamond\diamond\diamond}$.

References

1. Alam, M., Buzmakov, A., Codocedo, V., Napoli, A.: Mining definitions from RDF Annotations Using Formal Concept Analysis. In: Proceedings of IJCAI 2015. pp. 823–829 (2015)
2. Bendaoud, R., Napoli, A., Toussaint, Y.: Formal Concept Analysis: A Unified Framework for Building and Refining Ontologies. In: EKAW 2008. Proceedings. pp. 156–171 (2008)
3. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. Int. J. Semantic Web Inf. Syst. 5(3), 1–22 (2009)
4. Cimiano, P., Hotho, A., Staab, S.: Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. J. Artif. Intell. Res. (JAIR) 24, 305–339 (2005)
5. d'Amato, C., Staab, S., Tettamanzi, A.G., Minh, T.D., Gandon, F.: Ontology enrichment by discovering multi-relational association rules from ontological knowledge bases. In: Proceedings of ACM SAC'16. pp. 333–338 (2016)
6. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Fast rule mining in ontological knowledge bases with AMIE+. VLDB J. 24(6), 707–730 (2015)
7. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In: WWW'13. pp. 413–422 (2013)
8. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: International Conference on Conceptual Structures. pp. 129–142. Springer (2001)
9. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer (1999)
10. Gruber, T.R., et al.: A translation approach to portable ontology specifications. Knowledge acquisition 5(2), 199–220 (1993)
11. Lehmann, F., Wille, R.: A triadic approach to formal concept analysis. Conceptual structures: applications, implementation and theory pp. 32–43 (1995)
12. Lehmann, J., *et al.*: DBpedia - A large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web 6(2), 167–195 (2015)
13. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattices. Information systems 24(1), 25–46 (1999)
14. Paulheim, H.: Automatic Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods. Semantic Web Journal (To appear)
15. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In: Proceedings of WWW'07. pp. 697–706. ACM (2007)
16. Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auer, S.: Quality assessment for Linked Data: A Survey. Semantic Web 7(1), 63–93 (2016)