An FCA Framework for Knowledge Discovery in SPARQL Query Answers

Melisachew Wudage Chekol, Amedeo Napoli

An FCA Framework for Knowledge Discovery in SPARQL Query Answers

Melisachew Wudage Chekol and Amedeo Napoli

LORIA (INRIA, CNRS, and Université de Lorraine), France
{melisachew.chekol,amedeo.napoli}@inria.fr

Abstract. Formal concept analysis (FCA) is used for knowledge discovery within data. In FCA, concept lattices are very good tools for classification and organization of data. Hence, they can also be used to visualize the answers of a SPARQL query instead of the usual answer formats such as: RDF/XML, JSON, CSV, and HTML. Consequently, in this work, we apply FCA to reveal and visualize hidden relations within SPARQL query answers by means of concept lattices.

1 Introduction

Recently, the amount of semantically rich data available on the Web has grown considerably. Since the conception of linked data publishing principles, over 295 linked (open) datasets (LOD) have been produced[1]. A reasonable number of these datasets provide endpoints for accessing (querying) their contents. Querying is mainly done through the W3C recommended query language SPARQL. The answers of SPARQL queries have often the following formats: RDF/XML, JSON, CSV, text, TSV, Java Script, XML, Spreadsheet, Ntriples, and HTML. It might be interesting to analyse, mine, and then visualize hidden relations within the answers. These tasks can be carried out using Formal Concept Analysis (FCA).

FCA is used for knowledge discovery within data represented by means of objects and their attributes [3]. Concept lattices can reveal hidden relations within data and can be used for organizing, classifying, and even mining data. A survey of the benefits of FCA to semantic web (SW) and vice versa has been proposed in [6] (in particular ontology completion [1]). Additionally, studies in [2] and [4] are based on FCA for managing SW data. The former provides an entry point to linked data using questions in a way that can be navigated. It gives a transformation of an RDF graph into a formal context where the subject of an RDF triple becomes the object, a composition of the predicate and object of the triple becomes an attribute. The latter obliges the user to specify variables corresponding to objects and attributes of a context. These variables are used to create a SPARQL query which is used to extract content from linked data in order to build the formal context. Following this line, we propose a way to

[1] http://linkeddata.org/

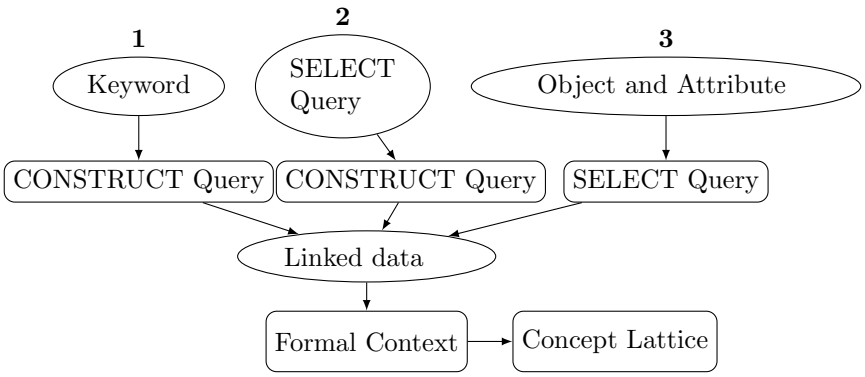

Fig. 1: Architecture of SPARQL answers organization.

organize Semantic Web data, and more precisely, the organization of SPARQL query answers by means of concept lattices. As a result, the user is able to visualize, navigate and classify the answers w.r.t. their context. For that, we propose the architecture depicted in Figure 1, based on three components which are discussed below:

1. *Keyword search:* In this component, a keyword (for instance, "14 juillet") is used to search (to find information regarding this word) a specified dataset. To do so, a URI produced from the keyword is sent to the dataset to check its existence. When this is the case, a CONSTRUCT query containing the keyword is directed to the endpoint of the dataset. The answers are collected and organized to create a formal context as explained in the next section.
2. *SELECT Query:* This component builds a formal context out of the answers of a SPARQL query. SPARQL queries are converted into CONSTRUCT queries to form RDF graphs from the answers. Again, this will be illustrated in the next section.
3. *Variables corresponding to objects and attributes of a formal context:* This component enables the user to precisely specify the objects and attributes of the formal context. Out of which a SPARQL query is formed and sent to a chosen SPARQL endpoint. The answers of the query are collected to build a formal context. From that, a concept lattice is constructed. This is the approach considered by the authors in [4]. An example is proposed in the next section.

In each case, the objective is to build a formal context and then to build the associated concept lattice.

2 Proposal

A formal context represents data using objects and their attributes. Formally, it is a triple $K = (G, M, I)$ where G is a set of objects, M is a set of attributes,

and $I \subseteq G \times M$ is a binary relation. A derivation operator ($'$) is used to compute *formal concepts* of a context. Given a set of objects A, a derivation operator $'$ computes the maximal set of attributes shared by objects in A and is denoted by A' (this is done dually with set of attributes B). A formal concept is a pair (A, B) where $A' = B$ and $B' = A$. A set of formal concepts ordered with the set inclusion relation form a *concept lattice* [3].

Definition 1 (RDF as a Formal Context). *Given an RDF graph G and a transformation function σ, a formal context is obtained from G as follows:*

- *If* $\langle o_1, \text{rdf} : \text{type}, C\rangle \in G$, *then* $\sigma(\langle o_1, \text{rdf} : \text{type}, C\rangle) =$

	C
o_1	x

- *If* $\langle o_1, R, o_2\rangle \in G$, *then* $\sigma(\langle o_1, R, o_2\rangle) =$

	$\exists R.\top$	$\exists R^-.\top$
o_1	x	
o_2		x

We consider a core fragment of RDFS called ρdf [5] which contains the minimal vocabulary, $\rho\text{df} = \{\texttt{sp}, \texttt{sc}, \texttt{type}, \texttt{dom}, \texttt{range}\}$, where `sp` denotes the *subproperty* relation, `sc` is *subclass*, and `dom` stands for *domain*. This fragment was proven to be minimal and well-behaved in [5]. Its semantics corresponds to that of full RDFS. Triples containing schema information are transformed as:

1. If $\langle C_1, \text{rdfs} : \text{subClassOf}, C_2\rangle \in G$, then $\sigma(\langle C_1, \text{rdfs} : \text{subClassOf}, C_2\rangle) = \forall o \in G.\ (o, C_1) \in I \Rightarrow (o, C_2) \in I$.
2. If $\langle R_1, \text{rdfs} : \text{subPropertyOf}, R_2\rangle \in G$, then $\sigma(\langle R_1, \text{rdfs} : \text{subPropertyOf}, R_2\rangle) = \forall o_1, o_2 \in G.\ (o_1, \exists R_1.\top) \in I \Rightarrow (o_1, \exists R_2.\top) \in I$.
3. If $\langle R, \text{rdfs} : \text{domain}, C\rangle \in G$, then $\sigma(\langle R, \text{rdfs} : \text{domain}, C\rangle) = \forall o_1 \in G.\ (o_1, \exists R.\top) \in I \Rightarrow (o_1, C) \in I$.
4. If $\langle R, \text{rdfs} : \text{range}, C\rangle \in G$, then $\sigma(\langle R, \text{rdfs} : \text{range}, C\rangle) = \forall o_2 \in G.\ (o_1, \exists R^-.\top) \in I \Rightarrow (o_2, C) \in I$.

The users above are able to build a formal context from an RDF graph or a set of SPARQL query answers. Then, there are several algorithms that can compute the concept lattice associated with a formal context and that can be used in our framework.

Example: consider a SPARQL query that selects film titles (as objects of the formal context) and genres (as attributes) from DBpedia to populate a formal context. Consequently, this formal context is shown in Figure 2.

A possible concept lattice obtained from the formal context associated with the query answers is depicted in Figure 3. Now we have a classification of the results of the query w.r.t. a given topic or constraint. Additionally, this is exactly the same thing as if we were querying the Web with Google and here we have a classification of the answers w.r.t. a user constraints.

3 Conclusion

SPARQL query answers are provided in different formats (RDF/XML, CSV, JSON, TTL, and others), which do not reveal hidden semantics in the answers.

	Orchestral	Action, A...	Western (...	Drama	Novelization
Jaws: The Revenge	X				X
Shanghai Surprise		X			
Brighty of the Grand Canyon (film)			X	X	
The Fish That Saved Pittsburgh					X
Tokyo Dragon					
Do You Speak American?					
Freaked					X

Fig. 2: A part of the formal context associated with the query.

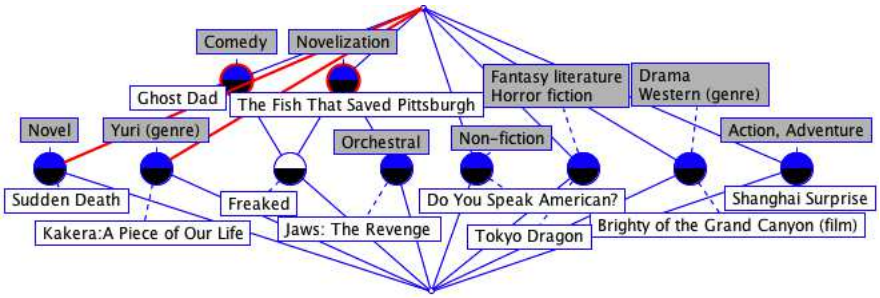

Fig. 3: Concept lattice of DBpedia movies and genres.

Concept lattices are useful in this regard. In this work, we used concept lattices to hierarchically organize and analyse the content of query answers.

This is an ongoing work and we are currently implementing the procedure. We should investigate how well it scales, given the size of SPARQL query answers over linked data. Overall, this work shows some of the benefits of FCA that can be provided to the semantic web.

References

1. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In: Proc. of IJCAI. vol. 7, pp. 230–235 (2007)
2. d'Aquin, M., Motta, E.: Extracting relevant questions to an RDF dataset using formal concept analysis. In: Proceedings of the sixth international conference on Knowledge capture. pp. 121–128. ACM (2011)
3. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Berlin (1999)
4. Kirchberg, M., Leonardi, E., Tan, Y.S., Link, S., Ko, R.K., Lee, B.S.: Formal concept discovery in semantic web data. In: ICFCA. pp. 164–179. Springer-Verlag (2012)
5. Muñoz, S., Pérez, J., Gutierrez, C.: Minimal deductive systems for RDF. In: The Semantic Web: Research and Applications, pp. 53–67. Springer (2007)
6. Sertkaya, B.: A survey on how description logic ontologies benefit from FCA. In: CLA. vol. 672, pp. 2–21 (2010)