Linkex: A Tool for Link Key Discovery Based on Pattern Structures

Nacira Abbas[1], Jérôme David[2], and Amedeo Napoli[1]

[1] Université de Lorraine, CNRS, Inria, Loria, F-54000 Nancy, France
Nacira.Abbas@inria.fr, Amedeo.Napoli@loria.fr
[2] Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France
Jerome.David@inria.fr

Abstract. Links constitute the core of Linked Data philosophy. With the high growth of data published in the web, many frameworks have been proposed to deal with the link discovery problem, and particularly the identity links. Finding such kinds of links between different RDF data sets is a critical task. In this position paper, we focus on link key which consists of sets of pairs of properties identifying the same entities across heterogeneous datasets. We also propose to formalize the problem of link key discovery using Pattern Structures (PS), the generalization of Formal Concept Analysis dealing with non binary datasets. After providing the proper definitions of link keys and setting the problem in terms of PS, we show that the intents of the pattern concepts correspond to link keys and their extents to sets of identity links generated by their intents. Finally, we discuss an implementation of this framework and we show the applicability and the scalability of the proposed method.

Keywords: Link key· Pattern Structures· Linked Data· RDF.

1 Introduction

Data interlinking is the task of finding the same entities described in different RDF datasets. To perfom this task, it has been proposed to use *link keys* [1] that extend the notion of a key used in databases. Link keys are rules allowing to infer identity links between RDF datasets. In practice, a link key is a set of pairs of properties from two datasets that generate `same-as` links. An example of a link key is:
$\{\langle \mathsf{auteur}, \mathsf{creator}\rangle\}\{\langle \mathsf{titre}, \mathsf{title}\rangle\}$ *linkkey* $\langle \mathsf{Livre}, \mathsf{Book}\rangle$ stating that whenever an instance of the class Livre has the same values for property auteur as an instance of class Book has for property creator and they share at least one value for their property titre and title, then they denote the same entity. An algorithm to extract link key candidates is proposed in [1]. However, these candidates are not yet link keys. In order to select the best link key candidate to apply as a link key, quality measures have been proposed but this is out of the scope of this paper.

The question of using Formal Concept Analysis (FCA) to extract such kind of keys has naturally arisen, given that the set of link key candidates is an ordered structure. A first formalization is given in [2] then developed in [3]. Yet, to apply FCA techniques, scaling (binarizing) data is applied, which may not be efficient, particularly when we have to deal with a large number of entries like RDF datasets. In this position paper, instead of scaling, we propose to work directly on link key expressions which are the syntactic formulations of link keys. To do that, we use Pattern Structures (PS) [4], a generalization of FCA to deal with non binary data. We show that the intents of resulting pattern concepts correspond to link key candidates and their extents to sets of identity links generated by their intents.

2 Notations and definitions

An RDF dataset is a set of triples:
$\langle subject, property, object\rangle \in (U \cup B) \times U \times (U \cup B \cup L)$, where U is a set of IRIs (Internationalized Resource Identifier), B a set of blank nodes and L a set of literals, i.e., values depending on datatypes. In this work, we consider only objects which are literals. To avoid confusion with PS objects, we refer to "object" in an RDF triple as "rdf object". Let $D, D^{'}$ be two RDF datasets, we aim to discover link keys between two specific classes $C, C^{'}$ from $D, D^{'}$ respectively. $C = \{s | \langle s$,*rdf:type*,$C\rangle \in D\}$ and $C^{'} = \{s^{'} | \langle s^{'}$,*rdf:type*,$C^{'}\rangle \in D^{'}\}$ are the sets of instances of the classes C and $C^{'}$ respectively.
$P = \{p | \exists s, o, \langle s, p, o\rangle \in D\}$ and $P^{'} = \{p^{'} | \exists s^{'}, o^{'}, \langle s^{'}, p^{'}, o^{'}\rangle \in D^{'}\}$ are the sets of properties from D and $D^{'}$ respectively. Unlike databases, the properties in RDF are not functional i.e. for one property, an instance may have more than one value i.e. rdf object. Here, $p(s) = \{o | \exists s, p, \langle s, p, o\rangle \in D\}$ is the set of the values of an instance s for the property p. Figure 1 represents an example of two datasets represented by instances of the classes *Person* and *Inhabitant* respectively. The goal here is to find a link key which identifies the same entities described in both classes. One may expect to identify that z_1 is the same entity as i_1 i.e. the *link* $\langle z_1, i_1\rangle$ as well as the links $\langle z_2, i_2\rangle$ and $\langle z_3, i_3\rangle$.

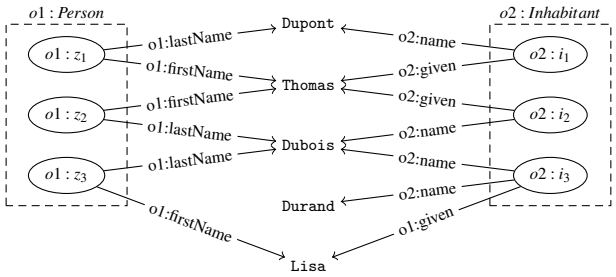

Fig. 1. Example of two datasets representing respectively instances of classes *Person* and *Inhabitant*.

A link key is a statement of the form:

$$\{\langle p_1, p_1^{'}\rangle, .., \langle p_k, p_k^{'}\rangle\}\{\langle p_1, p_1^{'}\rangle, .., \langle p_k, p_k^{'}\rangle, .., \langle p_n, p_n^{'}\rangle\}\ linkkey\ \langle C, C^{'}\rangle$$

meaning that for all pairs of instances $\langle s, s^{'}\rangle \in C \times C^{'}$, if s and $s^{'}$ have the same values,

i.e. rdf objects, for each pairs of properties $p_{1,..,k}$ and $p^{'}_{1,..,k}$ respectively, and share at least one value for each pair of properties $p_{k+1,..,n}$ and $p^{'}_{k+1,..,n}$ respectively, then they represent the same entity i.e. $(\langle s, \texttt{owl:sameAs}, s^{'}\rangle)$.

Definition 1 (Link key expression). *Let Eq and In be sets of pairs of properties $\langle p, p^{'}\rangle \in P \times P^{'}$ where $Eq \subseteq In$. $K = \langle Eq, In, \langle C, C^{'}\rangle\rangle$ is called a link key expression over two classes $C, C^{'}$. Notice that Eq and In may be empty.*

Since $Eq \subseteq In$, we write $K = Eq, In \setminus Eq$ for short. Which means that when a pair of properties belongs to the Eq part, we don't represent it in the In part. As example of link key expression over the two classes *Person* and *Inhabitant*, we have:
$\emptyset, \{\langle lastName, name\rangle, \langle firstName, given\rangle\}$ where Eq is empty.

Definition 2 (Satisfaction of a link key expression). *We say that a link $\langle s, s^{'}\rangle \in C \times C^{'}$ satisfies the link key expression $K = Eq, In$ and we write $K \models \langle s, s^{'}\rangle$ iff:*

$$\forall \langle p, p^{'}\rangle \in Eq \Longrightarrow p(s) = p^{'}(s^{'}) \neq \emptyset \text{ and } \forall \langle p, p^{'}\rangle \in In \Longrightarrow p(s) \cap p^{'}(s^{'}) \neq \emptyset$$

Notice that a link may satisfies more than one link key expression.
The set $L(K) = \{\langle s, s^{'}\rangle \in C \times C^{'} | K \models \langle s, s^{'}\rangle\}$ is the set of all links satisfying K and called the *link set* generated by K.

Definition 3 (meet, join, subsumption of link key expressions). *Let $K_1 = Eq_1, In_1$ and $K_2 = Eq_2, In_2$ two link key expressions. The meet $\sqcap$ and join $\sqcup$ of K_1 and K_2 are defined respectively as follow:*

$$K_1 \sqcap K_2 = Eq_1 \cap Eq_2, In_1 \cap In_2 \text{ and } K_1 \sqcup K_2 = Eq_1 \cup Eq_2, In_1 \cup In_2$$

We say K_2 subsumes K_1 and we write $K_1 \sqsubseteq K_2$ iff $K_1 \sqcap K_2 = K_1$ where $\sqsubseteq$ is a partial order between link key expressions.

3 Link key extraction with Pattern Structure

We propose here to formalize the problem of link key candidate extraction using PS [4] which is a generalization of FCA [5] to deal with non binary data.

Definition 4 (Pattern Structure for link key extraction). *A Pattern Structure for link keys extraction is a triple $(C \times C^{'}, (D, \sqcap, \sqcup), \delta)$ where, $C \times C^{'}$ is the set a pair of instances $\langle s, s^{'}\rangle \in C \times C^{'}$. D is a set of link key expressions over the two classes C and $C^{'}$. δ is a mapping associating each pair of instances $\langle s, s^{'}\rangle \in C \times C^{'}$ to its description K, defined as the maximal link key expression satisfied by $\langle s, s^{'}\rangle$ i.e. the join of the link key expressions satisfied by $\langle s, s^{'}\rangle$. Formally, $\delta(\langle s, s^{'}\rangle) = \sqcup\{K | K \models \langle s, s^{'}\rangle\}$.*

$(D, \sqcap, \sqcup)$ is a lattice. The *meet*, *join* and subsumption of two link key expressions are defined as in Definition 3. From the previous example of datasets represented in Figure 1, we obtained the Pattern Structure in Table 1. Here for example, to find $\delta(\langle z_1, i_1\rangle)$, we have computed the join of all link key exepressions satisfied by the link $\langle z_1, i_1\rangle$ and we obtained the description $\delta(\langle z_1, i_1\rangle) = \{\langle lastName, name\rangle, \langle firstName, given\rangle\}, \{\}$.

Notice that we omit pairs of instances for which descriptions correspond to the empty link key expression $K = \emptyset, \emptyset$.

Derivation operators $.^{\square}$ are defined as follows for $A \subseteq C \times C^{'}$ and $K \in D$:

$$A^{\square} = \sqcap_{\langle s,s' \rangle \in A} \delta(\langle s,s^{'} \rangle) \qquad \text{and} \qquad K^{\square} = \{\langle s,s^{'} \rangle \mid K \sqsubseteq \delta(\langle s,s^{'} \rangle)\}$$

(A,K) is a *pattern concept* if $A^{\square} = K$ and $K^{\square} = A$. We say that a pattern concept (A,K) is subsumed by another pattern concept (A_1,K_1) if $A \subseteq A_1$ (or equivalently if $K_1 \sqsubseteq K$). We aim here to extract link key candidates using the PS defined above.

Definition 5 (Link key candidate). *A link key expression K is a link key candidate over two classes $C,C^{'}$ if $L(K) \neq \emptyset$ and there is no link key expression H such as $H \sqsubseteq K$ that generates the same link set as K i.e. $L(K) \neq L(H)$*

One can see that, if (A,K) is a pattern concept, then its intent K represents a link key candidate and its extent is the link set generated by this link key candidate. Figure 2 represents the pattern concept lattice generated from the PS in Table 1. For example, the pattern concept:$\{\langle z_1,i_1\rangle,\langle z_2,i_2\rangle,\langle z_3,i_3\rangle\}$, $\{\langle firstName,given\rangle\}$,$\{\langle lastName,name\rangle\}$, identify the link key candidate in the intent, which generates the link set in the extent. This link key candidate can be applied as a link key. In fact, it generates the expected links. For example the instances z_3 and i_3 represent the same person since both of them had the same first name and share at least one family name. The class *Inhabitant* gives, for example, the birth name and the married name while the class *Person* gives just one of these two.

Table 1. The pattern structure computed from RDF datasets represented in Figure 1

	Eq	*In*
$\langle z_1,i_1\rangle$	$\{\langle lastName,name\rangle, \langle firstName,given\rangle\}$	$\{\}$
$\langle z_1,i_2\rangle$	$\{\langle firstName,given\rangle\}$	$\{\}$
$\langle z_2,i_1\rangle$	$\{\langle firstName,given\rangle\}$	$\{\}$
$\langle z_2,i_2\rangle$	$\{\langle lastName,name\rangle, \langle firstName,given\rangle\}$	$\{\}$
$\langle z_2,i_3\rangle$	$\emptyset$	$\{\langle lastName,name\rangle\}$
$\langle z_3,i_2\rangle$	$\{\langle lastName,name\rangle\}$	$\{\}$
$\langle z_3,i_3\rangle$	$\{\langle firstName,given\rangle\}$	$\{\langle lastName,name\rangle\}$

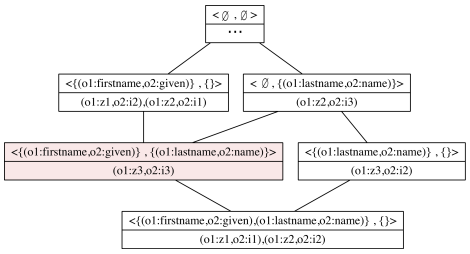

Fig. 2. The resulting lattice from the pattern structure in Table 1.

4 Tool Description

We have implemented a prototype tool called `Linkex` [1], which aims to extract link key candidates using PS. It starts from two RDF data sets and computes a PS, then generates a pattern concept lattice using a modified version of `AddIntent` algorithm [6]. The intent of each resulting pattern concept corresponds to a link key candidate and its extent to the set of identity links generated by this link key candidate. When the resulting pattern concept lattice is small, `Linkex` gives the possibility to visualize link key candidates in a nice way i.e. organized in a lattice represented by a Hasse diagram. For example the Figure 2, which represents a pattern concept lattice, is a direct output of this tool. `Linkex` offers also the possibility to assess the quality of the extracted link keys candidates using the measures proposed in [1].

[1] https://gitlab.inria.fr/moex/linkex

To demonstrate the applicability and the scalability of the proposed tool, we have experimented with two bibliographical datasets. The first one[2] is produced by BnF, the "Bibliothèque nationale de France". The second one[3] is produced by ABES, "Agence Bibliographique de l'Enseignement Supérieur". BnF and ABES, have provided us with two samples of data for the experiments. The first sample is an extraction of authors instances (and their books instances) that have a combination firstname, name which is in the top 1000 most frequent homonyms. The second sample is an extraction of all authors instances (and their book instances) that have a name starting with letter 'A'. We aim here to find link key candidates between instances of the classes representing authors contained in both datasets. As showed in Table 2, we got short running times comparing to the size of processed data. Notice that, the processing time is correlated to the size of datasets. Yet, the most interesting result here is the number of the obtained pattern concepts i.e. the number of link key candidates represented by the column **#PsConcepts** in Table 2. For example, in sample 1, from 1,564,495 pattern structure descriptions, we got only 155 link key candidates. As a direct consequence of this, is that an expert could easily navigate these link key candidates and evaluate them. Thus, we can say that the proposed method implemented as Linkex is applicable and scalable. l

Table 2. Experimentation results

Samples	#BnF	#ABES	#PsEntries	#PsConcepts	Processing time
Sample 1	8,162	15,421	1,564,495	155	2m10
Sample 2	18,637	142,571	12,348,012	186	6m50

5 Conclusions

In this work, we formalize the problem of extracting link key candidates using PS and we propose a tool allowing to automatically build a pattern concept lattice where each pattern concept intent is a candidate link key. We aim to extend this work to consider interdependent classes i.e when rdf objects are instances of other classes and we plan also to relax equality constraints used to compare literals by considering similarity measures instead.

Acknowledgments

This work has been supported by the ANR project Elker (ANR-17-CE23-0007-01) and the BnF in the context of the agreement between Inria and Ministère de la culture.

References

1. Atencia, M., David, J., Euzenat, J.: Data interlinking through robust linkkey extraction. In: ECAI. pp. 15–20 (2014)

[2] https://data.bnf.fr
[3] https://www.idref.fr

2. Atencia, M., David, J., Euzenat, J.: What can fca do for database linkkey extraction? In: 3rd ECAI workshop on What can FCA do for Artificial Intelligence?(FCA4AI). pp. 85–92. No commercial editor. (2014)
3. Atencia, M., David, J., Euzenat, J., Napoli, A., Vizzini, J.: Link key candidate extraction with relational concept analysis. Discrete Applied Mathematics (2019)
4. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: ICCS. pp. 129–142. Springer (2001)
5. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer Science & Business Media (2012)
6. Van Der Merwe, D., Obiedkov, S., Kourie, D.: Addintent: A new incremental algorithm for constructing concept lattices. In: ICFCA. pp. 372–385. Springer (2004)