A guided walk into link key candidate extraction with relational concept analysis

Manuel Atencia, Jérôme David, Jérôme Euzenat, Amedeo Napoli, and Jérémy Vizzini

Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France
Firstname.Lastname@inria.fr

Abstract. Data interlinking is an important task for linked data interoperability. One of the possible techniques for finding links is the use of link keys which generalise relational keys to pairs of RDF models. We show how link key candidates may be directly extracted from RDF data sets by encoding the extraction problem in relational concept analysis. This method deals with non functional properties and circular dependent link key expressions. As such, it generalises those presented for non dependent link keys and link keys over the relational model. The proposed method is able to return link key candidates involving several classes at once.

In this extended abstract, we describe the content of [2] by avoiding technical descriptions related to the domain. Interested readers are referred to the original paper for more details. To facilitate the compared reading, we explicitly refer to the original definitions and properties.

1 Interlinking RDF data sets

Interlinking RDF data sets is an important operation for linked (open) data on which research is currently very active.

1.1 RDF data sets and linked data

Linked data aims at publishing data expressed in RDF (Resource Description Framework) at the scale of the worldwide web [6,12]. In this paper we consider RDF data sets as sets of triples. We simply identify properties as the IRIs in predicate position in one of these triples and classes as the IRIs found in the codomain of the `rdf:type` predicates. This is the only schema information that is used. These data sets are not required to share vocabulary elements beside the RDF (for `rdf:type`) and OWL (for `owl:sameAs`) vocabularies.

Heterogeneous data sets interoperate through links which identify individuals across them. An important added value of linked data arises from the links

that identify the same entity in different data sets. They allow one to develop innovative applications exploiting data cross-references and making inferences between data sets.

1.2 Data interlinking

Data interlinking is the problem of finding IRIs described in RDF over different data sets that refer to the same resource. It is thus a critical task for widening and enhancing linked data. This is a knowledge discovery task as it infers knowledge —the condition for identity of resources— from large volumes of data.

Different approaches and methods have been proposed to address the problem of automatic data interlinking [10,18]. Two main trends are used to tackle this problem: numerical methods and logical methods. Numerical methods usually compute a similarity between resources based on their property values to establish links between those that are very similar [6,19,25]. Logical methods use an axiomatic characterisation of what makes two resources the same so as to find the `owl:sameAs` links between IRIs of different data sets [23,3,13].

2 Link key extraction

Our work belongs to the logic-based approach and is more precisely based on *link keys*. Thus we first introduce them before considering approaches to extract them from data.

2.1 Link keys

We introduced the notion of *link keys* as a way to find pairs of IRI identifying the same resource [9,1]. Link keys generalise keys in relational algebra in three ways adapted to RDF: (a) they apply across two data sets instead of a single one, (b) they take into account multiple values for the same property, and (c) property values may be other objects. The latter makes link keys eventually dependent on each others.

A *link key expression* [2, Def. 4] specifies the pairs of properties to compare for linking individuals belonging to different classes of the data sets. An example of a link key expression is:

$$\{\langle \mathsf{auteur}, \mathsf{creator}\rangle\}\{\langle \mathsf{titre}, \mathsf{title}\rangle\}\ \mathit{linkkey}\ \langle \mathsf{Livre}, \mathsf{Book}\rangle$$

stating that whenever an instance of the class Livre has the same values for the property auteur as an instance of class Book has for the property creator and they share at least one value for their property titre and title, then they denote the same entity.

A link key expression thus applies to a pair of classes ($\langle \mathsf{Livre}, \mathsf{Book}\rangle$). In the above expression, there are two sets of conditions (each reduces to a singleton): the eq-link key condition ($\{\langle \mathsf{auteur}, \mathsf{creator}\rangle\}$) and the in-link key condition ($\{\langle \mathsf{titre}, \mathsf{title}\rangle\}$). The abbreviations eq- and in- stand for equality and intersection.

It is easy to understand what *link set is generated by such a link key expression* given two RDF data sets [2, Def. 6].

Such a link key may depend on another one as, for instance, properties auteur and creator have values in the Écrivain and Writer classes respectively. Identifying their values will then depend on another link key:

$$\{\langle \mathsf{nom}, \mathsf{lastname}\rangle\}\{\langle \mathsf{prénom}, \mathsf{firstname}\rangle\}\ \mathit{linkkey}\ \langle \mathsf{Écrivain}, \mathsf{Writer}\rangle$$

This situation may be rendered even more intricate if Écrivain and Writer were instead identified from the values of their properties ouvrage and hasWritten refering to instances of Livre and Book. We would then face circularly dependent link keys.

2.2 Link key extraction

The goal of link key extraction is, given two data sets, to find which link keys hold.

Key-based data interlinking requires to have pairs of keys related by alignment (or schema mapping). Keys are either given or extracted from the data. Key extraction (and more generally functional dependencies discovery) has been addressed both in databases [24,14] and in the semantic web [3,20]. Systems for data interlinking from keys [20] can only extract a specific (strong) kind of in-link keys, which may not be the best candidates.

In [1], we have proposed to discover directly in-link keys between two classes from two data sets. The proposed algorithm does not require an initial alignment between properties of both data sets and also avoids the generation of keys that are specific to only one data set. It may be decomposed in two distinct steps: (i) identifying *link key candidates*, i.e. maximal link key expressions that would generate at least one link if used as a link key, followed by (ii) selecting the best link key candidate according to quality measures. To select the best link key candidate, we have introduced two sets of measures.

The purpose of this work is to reformulate Step (i) within the framework of formal concept analysis and to extend it towards dependent link key expressions.

3 Link key candidate extraction and formal concept analysis

The presented paper [2] aims at performing link key extraction through using Formal Concept Analysis (FCA). We thus first briefly introduce it. Then we summarize our previous work relating link key extraction algorithms and FCA. We finally present the full reformulation of link key extraction within FCA.

3.1 Formal Concept Analysis

Formal Concept Analysis (FCA) is a set of techniques, allowing data analysis and classification, based on solid mathematical foundations and efficient algorithms

[11]. It starts with a binary table, called formal context, relating objects to attributes (or properties) and outputs a set of concepts which are pairs composed of a maximal set of objects sharing a corresponding maximal set of attributes. In addition, concepts are partially ordered within a concept lattice.

Using lattices is common place for extracting functional dependencies [16,8,17]. The problem was considered in FCA in a functional setting [11] and further refined in pattern structures, the extension of FCA to complex data [5,7]. However, FCA is defined for functional properties and must be adapted to the other cases, especially with multi-valued and relational attributes.

3.2 Link key candidates are formal concepts

We have previously shown how to encode the link key extraction problem in relational databases into formal concept analysis so that link key candidates correspond to formal concepts [4]. This was a simplification of the method used for extracting functional dependencies in [11] in which the formal context relates pairs of properties to pairs of objects. The relation holds between a pair of objects and a pair of properties if the objects have the same value for this property.

This works in the functional case of relational data bases in which eq-conditions and in-conditions are equivalent (for non NULL cells).

This preliminary work showed that the approach was promising, but extensions were needed in several directions, among which:

1. Non-functional properties lead to distinguish eq- and in-conditions. This requires integrating them in the proposed formal contexts.
2. Relational properties lead to the possible existence of link keys depending of others link keys, e.g. individuals of the pair of classes $\langle \mathsf{Livre}, \mathsf{Book} \rangle$ will depend on the classes $\langle \mathsf{Écrivain}, \mathsf{Writer} \rangle$ as the value of the $\langle \mathsf{auteur}, \mathsf{creator} \rangle$ property pairs.
3. Moreover, some classes may depend on themselves, directly or indirectly. For instance, if identifying individuals of the pair of classes $\langle \mathsf{Écrivain}, \mathsf{Writer} \rangle$ may, in turn, depend on the pair of classes $\langle \mathsf{Livre}, \mathsf{Book} \rangle$ through the pair of properties $\langle \mathsf{ouvrage}, \mathsf{hasWritten} \rangle$.

3.3 Reformulation and extensions

A first contribution of [2] is the reformulation of the former encoding and the proof that it indeeds finds all and only link key candidates.

We first define the notion of *meet, join and subsumption between link key expresions* [2, Def. 5] and *reformulate link key candidates as closed by join* [2, Def. 8]. This covers relevant link key expressions missed by the previous definitions. We prove that reformulated *link key candidates are either previous link key candidates or join of link key candidates* [2, Prop. 2].

We then define *formal contexts for dependent link key candidates* [2, Def. 9]. They extend the former contexts by qualifying the pairs of properties as acting either as an in-condition or an eq-condition. The former relates two objects sharing at least one value for these properties; the latter relates them if they

share all their values. We show that, as redefined, *link key candidates are formal concepts* [2, Prop. 3].

4 Dependent link key candidate extraction

So far, we were only concerned with extracting link key candidates between two identified classes independently of other classes. This means that, either this can only take into account literal values, i.e. data properties, or the equality between objects that appear as relation values require `owl:sameAs` statements.

A first enhancement is thus to express dependent link key expressions, e.g. link keys of the class Book that contains the author property will depend on link keys of the class Writer. For that purpose, we extend formal contexts to cover object properties such that a pair of object properties be qualified by the link key expression used for judging the equality of instances. For example, this can resort to expressing that the object should share a value in the property auteur and creator judging this equality on the equality between their property noinsee and ssnumber. This is integrated in the definition of *formal contexts for dependent link key candidates* [2, Def. 11] by adding for each pair of object properties the keys used for linking them.

This raises a subsequent problem: it is now possible to have the same pair of classes (Livre and Book) using different link keys for judging the equality of the same pair of classes (Personne and Person) depending on the pair of property that is compared, e.g. $\langle$auteur, author$\rangle$ and $\langle$traducteur, translator$\rangle$. This seems weird. It may be preferable to always compare the instances of the same pair of classes with the same link key expression.

This is the reason why we define *coherent families of link key candidates* [2, Def. 12] which assign at most one single link key candidate to any pair of classes.

5 Circular dependencies and relational concept analysis

Circular dependencies across link keys raise the problem of self-supported links, i.e. EEUU is the same as USA if Nueva York is the same as New-York which relies on EEUU being the same as USA. Since the goal of link keys is to find links between data, self-supported links should not be discarded upfront.

Circular dependencies are commonplace in ontologies, e.g. the class Person depending on itself through the property parent. The use of a relation and its inverse (author-hasWritten) is already a simple case of circular dependency.

Such characteristics require extensions of the genuine FCA framework. We have used relational concept analysis for that purpose.

5.1 Relational Concept Analysis

Relational Concept Analysis [21] has been designed for dealing with relational attributes and circular dependencies in the FCA framework. As for FCA, its

main goal is to find concept descriptions, i.e. descriptions of classes expressed in function of other class descriptions through properties, like in description logics. For that purpose, it deals with a relational context family which is made of a set of classical formal contexts, one per class, and a set of relational contexts, one per object property. The relational contexts is a binary table between objects of two contexts. It simply express which objects are in relation with which other objects through a particular object property. In addition, RCA uses scaling operators to introduce new discriminating properties within the formal contexts.

Its concept extraction process starts by (a) applying FCA to each formal context. This provides new concept lattices. Then (b) the scaling operators add new attributes to the formal contexts of classes depending on the extracted concepts for the co-domain of their object properties. The process returns to Step (a) until no change occurs. This has been proved to converge [22]. As a result, RCA returns one concept lattice for each formal context.

5.2 Extension to circular dependencies

Extracting link keys, i.e. conditions for pairs of objects to be the same, is not exactly like extracting class descriptions, i.e. conditions for an object to belong to a class. However, as we defined a suitable encoding for FCA, it is possible to extend it, even for RCA. In fact, the formal contexts are the same as in the previous section and the *relational contexts for dependent link key candidates* [2, Def. 13] are the same as in RCA except that they relate pairs of objects through pairs of object properties. A *relational context family for dependent link key candidates* [2, Def. 15] is made of such contexts, eventually restricted by an alignment.

The novelty lies on the introduction of two specific scaling operators, the *link key condition scaling operators* [2, Def. 14], which generate the corresponding in- and eq-conditions when necessary, i.e. when this discriminates between generated concepts. Coherent families of link key candidates can be extracted from the resulting lattices and evaluated.

6 Complexity and implementation

It may be strange that what the paper offers is a set of formal context definitions. This is the way a problem is expressed in formal concept analysis. If this is rightly done, the FCA/RCA process provides the solution to the problem, i.e. concept lattices.

We have implemented a full proof-of-concept in Python[1] which supports all encodings described here and returns the concept lattice from data sets expressed in genuine RDF. It is also provided with many toy examples for torturing the model. This implementation is not particularly efficient.

[1] https://moex.inria.fr/software/linkky/

We are currently reimplementing it in Java in our LinkEx tool[2]. This is not yet fully completed.

The paper discusses the exponential data and 'schema' complexity of the proposed algorithm as well as the observation that, in practice, the data sets do not generate that many link keys. LinkEx is able to generate and evaluate 834 link key candidates from 99116 (349×284) possible links and 255 (15×17) pairs of properties within a few seconds on a laptop.

7 Conclusion

This paper provides a full account of using relational concept analysis to extract link key candidates from two RDF data sets containing various classes. All the contexts defined in this paper are extension of others, i.e. each considered problem is expressed in the same way in the context corresponding to it as in the more expressive contexts and all provide exactly the same results. The proposed approach works as well for unary link keys, i.e. keys, or n-ary link keys. No alignment is necessary, contrary to [1], though it is always possible to inject an alignment in the definitions to reduce the search space.

Once coherent families of link key candidates are extracted, they need to be evaluated. Available measures can apply in the same way because they are usually defined on the generated links.

Among further development of this approach, we plan to consider two research directions:

- Optimising the extraction process through interleaving evaluation and extraction and through using sampling techniques;
- Relaxing the data equality constraint by using similarity measures considered as tolerance relations [15].

8 Acknowledgements

This work has been partially supported by the ANR project Elker (ANR-17-CE23-0007-01, for the four first authors). Jérémy Vizzini has been partially supported by a grant from the PERSYVAL-Lab LabEx (ANR-11- LABX-0025-01) funded by the French "Investissement d'avenir" program.

References

1. Atencia, M., David, J., Euzenat, J.: Data interlinking through robust linkkey extraction. In: Proc. 21st european conference on artificial intelligence (ECAI), Praha (CZ). pp. 15–20 (2014)
2. Atencia, M., David, J., Euzenat, J., Napoli, A., Vizzini, J.: Link key candidate extraction with relational concept analysis. Discrete applied mathematics (2019), to appear

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

3. Atencia, M., David, J., Scharffe, F.: Keys and pseudo-keys detection for web datasets cleansing and interlinking. In: Proc. 18th international conference on knowledge engineering and knowledge management (EKAW), Galway (IE). Lecture Notes in Computer Science, vol. 7605, pp. 144–153. Springer (2012)
4. Atencia, M., David, J., Euzenat, J.: What can FCA do for database linkkey extraction? In: Proc. 3rd ECAI workshop on What can FCA do for Artificial Intelligence? (FCA4AI), Praha (CZ). pp. 85–92 (2014)
5. Baixeries, J., Kaytoue, M., Napoli, A.: Characterizing functional dependencies in formal concept analysis with pattern structures. Annals of Mathematics and Artificial Intelligence 72(1–2), 129–149 (2014)
6. Bizer, C., Heath, T., Berners-Lee, T.: Linked data — the story so far. International Journal of Semantic Web Information Systems 5(3), 1–22 (2009)
7. Codocedo, V., Baixeries, J., Kaytoue, M., Napoli, A.: Characterization of order-like dependencies with formal concept analysis. In: Proc. 30th International Conference on Concept Lattices and Their Applications (CLA), Moscow (RU). CEUR Workshop Proceedings, vol. 1624, pp. 123–134 (2016)
8. Demetrovics, J., Libkin, L., Muchnik, I.: Functional dependencies in relational databases: A lattice point of view. Discrete Applied Mathematics 40(2), 155–185 (1992)
9. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (DE), 2nd edn. (2013)
10. Ferrara, A., Nikolov, A., Scharffe, F.: Data linking for the semantic web. International Journal of Semantic Web and Information Systems 7(3), 46–76 (2011)
11. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer (1999)
12. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool (2011)
13. Hogan, A., Zimmermann, A., Umbrich, J., Polleres, A., Decker, S.: Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora. Journal of Web Semantics 10, 76–110 (2012)
14. Huhtala, Y., Kärkkäinen, J., Porkka, P., Toivonen, H.: TANE: An efficient algorithm for discovering functional and approximate dependencies. The Computer Journal 42(2), 100–111 (1999)
15. Kaytoue, M., Kuznetsov, S., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Information science 181(10), 1989–2001 (2011)
16. Levene, M.: A lattice view of functional dependencies in incomplete relations. Acta cybernetica 12(2), 181–207 (1995)
17. Lopes, S., Petit, J.M., Lakhal, L.: Functional and approximate dependency mining: database and FCA points of view. Journal of Experimental & Theoretical Artificial Intelligence 14(2-3), 93–114 (2002)
18. Nentwig, M., Hartung, M., Ngonga Ngomo, A.C., Rahm, E.: A survey of current link discovery frameworks. Semantic web journal 8(3), 419–436 (2017)
19. Ngonga Ngomo, A.C., Auer, S.: LIMES: A time-efficient approach for large-scale link discovery on the web of data. In: Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI). pp. 2312–2317. Barcelona (ES) (2011)
20. Pernelle, N., Saïs, F., Symeounidou, D.: An automatic key discovery approach for data linking. Journal of Web Semantics 23, 16–30 (2013)
21. Rouane Hacene, M., Huchard, M., Napoli, A., Valtchev, P.: Relational concept analysis: mining concept lattices from multi-relational data. Annals of Mathematics and Artificial Intelligence 67(1), 81–108 (2013)

22. Rouane Hacene, M., Huchard, M., Napoli, A., Valtchev, P.: Soundness and completeness of relational concept analysis. In: Proc. 11h International Conference on Formal Concept Analysis (ICFCA). Lecture Notes in Computer science, vol. 7880, pp. 228–243. Springer (2013)
23. Saïs, F., Pernelle, N., Rousset, M.C.: L2R: A logical method for reference reconciliation. In: Proc. 22nd National Conference on Artificial Intelligence (AAAI), Vancouver (CA). pp. 329–334. AAAI Press (2007)
24. Sismanis, Y., Brown, P., Haas, P., Reinwald, B.: GORDIAN: efficient and scalable discovery of composite keys. In: Proc. 32nd international conference on very large databases (VLDB). pp. 691–702 (2006)
25. Suchanek, F., Abiteboul, S., Senellart, P.: PARIS: Probabilistic Alignment of Relations, Instances, and Schema. PVLDB 5(3), 157–168 (2012)