Mapping Patterns for Virtual Knowledge Graphs (A Report on Ongoing Research) *

Diego Calvanese[1,2][0000−0001−5174−9693], Avigdor Gal[3][0000−0002−7028−661X], Davide Lanti[1][0000−0003−1097−2965], Marco Montali[1][0000−0002−8021−3430], Alessandro Mosca[1][0000−0003−2323−3344], and Roee Shraga[3][000−0001−8803−8481]

[1] Free-University of Bozen-Bolzano, Bolzano, Italy, *lastname*@unibz.it
[2] Umeå University, Umeå, Sweden, diego.calvanese@umu.se
[3] Technion – Israel Institute of Technology, Haifa, Israel
avigal@technion.ac.il, shraga89@campus.technion.ac.il

1 Introduction

In data integration and access to legacy data sources using end user-oriented languages, the approach based on *Virtual Knowledge Graphs (VKG)* is gaining importance [8]. A VKG specification consists of three main components: *(i)* (relational) *data sources*, where the actual data are stored; *(ii)* a domain *ontology*, capturing the relevant concepts, relations, and constraints of the domain of interest; and *(iii)* a set of *mappings* linking the data sources to the ontology. One of the most critical bottlenecks towards the adoption of the VKG approach, especially in complex, enterprise scenarios, is precisely the definition and management of mappings. Indeed, on the one hand, VKG mappings map complex queries to complex queries, similar to mappings typically used in data integration and exchange [4]. Thus, they are inherently more sophisticated than mappings used, e.g., in schema matching [6] and ontology matching [2]. On the other hand, they need to overcome the abstraction mismatch between the relational schema of the underlying data storage, and the target ontology; consequently, they are required to explicitly handle how (tuples of) data values extracted from the DB lead to the creation of corresponding objects in the ontology [5].

As a consequence, management of VKG mappings throughout their entire life-cycle is currently a labor-intensive, essentially manual effort, which requires highly-skilled professionals [7]. Even for such professionals, writing mappings is demanding and poses a number of challenges related to semantics, correctness, and performance. More concretely, no comprehensive approach currently exists to support ontology engineers in the creation of VKG mappings, exploiting all the involved information artifacts to their full potential: the relational schema with its constraints and the extensional data stored in the DB, the ontology axioms, and a conceptual schema that lies, explicitly or implicitly, at the basis of the relational schema.

2 Contributions

In our ongoing work, we build on this key observation and provide the contributions described in the following.

2.1 A Catalog of VKG Mapping Patterns

We propose a catalog of mapping patterns that emerge when linking DBs to ontologies. To do so, we build on well-established methodologies and patterns studied in data management (such as W3C direct mappings – W3C-DM [1] – and their extensions), data analysis (such as algorithms for discovering dependencies), and conceptual modeling (such as relational mapping techniques). In specifying each pattern, we consider not only the three main components of a VKG specification – namely the relevant portions of the DB schema, the ontology, and the mapping between the two – but also the conceptual schema of the domain of interest and the underlying data, when available. We do not fix which of these information artifacts are given and which are produced as output, but we simply describe how they relate to each other, on a per-pattern basis.

We organize patterns in two major groups: *schema-driven patterns*, shaped by the structure of the DB schema and its explicit constraints, and *data-driven patterns*, which in addition consider constraints emerging from specific configurations of the data in the DB. For each schema-driven pattern, we actually identify a data-driven version in which the constraints over the schema are now not explicitly specified, but hold in the data. But we provide also data-driven patterns that do not have a schema-driven counterpart. The two types of patterns can be used in combination with additional semantic information from the ontology, for instance on how the data values from the DB translate into RDF literals. These considerations lead us to introduce also *pattern modifiers*. Moreover, some of our patterns come with accessory views defined over the DB-schema, which make explicit the presence of specific structures over the DB schema that are revealed through the application of the pattern itself. Such views can be used themselves, together with the original DB schema, to identify the applicability of further patterns.

2.2 Design Scenarios for VKG Mapping Patterns

The proposed patterns can be employed in a variety of VKD design scenarios, depending on which information artifacts are available, and which ones have to be produced. Specifically, we consider the following scenarios: *(i) Debugging of a VKG Specification*, which arises when a full VKG specification is already in place and must be debugged. *(ii) Conceptual Schema Reverse Engineering*, which aims at inferring a conceptual schema of the DB that represents the domain of interest by reflecting the content of a given full VKG specification. *(iii) Mapping Bootstrapping*, where the DB and the ontology are given, but mappings relating them are not, and patterns can be used to (semi-)automatically bootstrap an initial set of mappings. These can then be further refined and extended manually,

possibly exploiting again the patterns. *(iv) Ontology+Mapping Bootstrapping*, where neither the ontology nor the mappings are given as input, and have to be synthesized. This scenario can be reduced to the previous one by first inducing a baseline ontology mirroring the structure of the DB schema. *(v) VKG Bootstrapping*, where we just have a conceptual schema of the domain, and the goal is to set up a VKG specification. The conceptual schema can be then transformed into a normalized DB schema using well-established *relational mapping* techniques (e.g., [3]).

2.3 Analysis of Scenarios

In our work, we have analyzed the concrete mapping strategies arising from a number of VKG use cases in order to understand how patterns occur in practice, and with which frequency. To this purpose, we have gathered 6 different scenarios, coming either from the literature on VKGs, or from actual real-world applications, covering a variety of different application domains. So far, we have manually classified a total of 1582 mapping assertions, falling in 367 pattern applications. We have studied the coverage of mappings appearing therein in terms of our patterns, as well as on how many times the same pattern recurs. Our investigation has shown that only 3% of pattern applications fall outside of our categorization, and it also gives an interesting indication on which patterns are more pervasively used in practice.

3 Conclusions

The work carried out so far is only a first step, with respect to both categorization of patterns, and their actual use. Regarding the former, we are now exploring more in depth the interaction between patterns and pattern modifiers, such as value invention or identifier alignment. Regarding the latter, so far we have used patterns to investigate, and highlight, the specific problems to address when setting-up a VKG scenario. We are now investigating solutions to these problems, by exploiting approaches from other fields, e.g., schema matching.

Acknowledgements

This research has been partially supported by: the EU H2020 project INODE; the Italian PRIN project HOPE; the European Regional Development Fund (ERDF) Investment for Growth and Jobs Programme 2014-2020 through the project IDEE (FESR1133); the Free University of Bozen-Bolzano through the projects KGID, GeoVKG, OntoGeo, and STyLoLa; the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

1. Arenas, M., Bertails, A., Prud'hommeaux, E., Sequeda, J.: A direct mapping of relational data to RDF. W3C Recommendation, World Wide Web Consortium (Sep 2012), available at http://www.w3.org/TR/rdb-direct-mapping/
2. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer (2007)
3. Halpin, T., Morgan, T.: Information Modeling and Relational Databases. Morgan Kaufmann (2010)
4. Lenzerini, M.: Data integration: A theoretical perspective. In: Proc. of the 21st ACM Symp. on Principles of Database Systems (PODS). pp. 233–246 (2002). https://doi.org/10.1145/543613.543644
5. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. J. on Data Semantics **10**, 133–173 (2008)
6. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. Very Large Database J. **10**(4), 334–350 (2001)
7. Spanos, D.E., Stavrou, P., Mitrou, N.: Bringing relational databases into the Semantic Web: A survey. Semantic Web J. **3**(2), 169–209 (2012)
8. Xiao, G., Ding, L., Cogrel, B., Calvanese, D.: Virtual Knowledge Graphs: An overview of systems and use cases. Data Intelligence **1**(3), 201–223 (2019). https://doi.org/10.1162/dint_a_00011