Query Reverse Engineering in the Context of the Semantic Web: A State-of-the-Art

DOI: 10.1007/978-981-16-1480-4_26
Handle: http://hdl.handle.net/1942/33837

Query Reverse Engineering in the context of the Semantic Web: a state-of-the-art

Leandro Tabares-Martín[1], Nemury Silega-Martínez[2], and Marc Gyssens[3]

[1] Hasselt University and University of Informatics Sciences
leandro.tabaresmartin@uhasselt.be, ltmartin@uci.cu
[2] University of Informatics Sciences nsilega@uci.cu
[3] Hasselt University marc.gyssens@uhasselt.be

Abstract. The Query Reverse Engineering problem tries to discover a query that satisfies a set of examples based on data stored in a data source. Notwithstanding this problem has been an active research field in the relational database's research community during several years, it is starting to be approached in the context of the Semantic Web. This study is the first of its kind providing a systematic review of the state-of-the-art with regard to Query Reverse Engineering in the context of the Semantic Web. Guided by a methodology for conducting systematic mapping studies, this paper provides insights about the existing approaches to the problem, as well as some of the remaining research opportunities in it.

Keywords: query by example, query by output, query reverse engineering, semantic web

1 Introduction

In its simplest form, the Query Reverse Engineering (QRE) problem can be defined as stated by Tran et al. [27]: "given a database D and a result table T, which is the output of some *known* or *unknown* query Q on D; the goal of QRE is to reverse engineer a query Q' such that the output of query Q' on database D is equal to T". While trying to fill the gap between non-specialized users and database technologies, researchers working in this problem have produced an array of related concepts and subfields which are not immediately easy to distinguish, such as *query-by-example*, *query-by-output*, and *query reverse engineering*, among others [13].

The principal aim of a systematic mapping study such as the one we are presenting here, is to provide an overview of a research field by classifying contributions pertaining to that field [24]. Such studies can help us in getting a significantly better insight in the subject at hand [11,8,9,15].

Such a systematic mapping methodology was developed originally by the "Evidence for Policy and Practice Information and Coordinating Centre" (EPPI) [23,22]. Later, the EPPI mapping methodology was modified by the "Social Care Institute for Excellence" (SCIE). This was motivated on the one hand by a shortage of empirical data necessary for answering certain questions using the

systematic review methodology, and on the other hand the need to describe the literature in a broad field of interest [9]. SCIE also introduced the term "systematic mapping methodology". They worked out detailed instructions for reviewers [9]. Currently, these kinds of studies are being used in various domains of knowledge following appropriately adapted methodologies [10,16,7,25].

With regard to Query Reverse Engineering, other systematic studies have been conducted [20]. However, these studies have not been focused on Semantic Web technologies. The aim of the current work is to make a systematic study of published literature about Query Reverse Engineering in the context of the Semantic Web.

The remainder of this paper is organized as follows: Section 2 gives some background of related studies in the field. Section 3 explains the methodological design of this study, and Section 4 describes our findings. Finally, Section 5 summarizes the main conclusions arrived after conducting the current study, as well as promising opportunities for future work.

2 Background and related studies

An important precursor to the field of Query Reverse Engineering is "Query By Example" (QBE). In the mid 1970s, Zloof [29] introduced this language to facilitate query writing in the relational model. It has been extensively investigated since then (see, e.g., [21], and references therein). The connection with the theme of this paper is that processing QBE has often been described as reverse engineering database queries from representative examples. There are actually two different aspects to this. The first aspect is to determine whether a query satisfying all examples provided actually exists. The second aspect is then finding such a query. These two aspects can be formalized as follows based on the work of [28]:

Suppose we are given a query family Q, a database D, and a set E of examples consisting of pairs of instances over D and corresponding outputs.

- **Problem 1 (Satisfiability)**. Determine whether there exists a query in Q that satisfies E.
- **Problem 2 (Learning)**. Find a query in Q that satisfies E, if such a query exists.

In the context of the Semantic Web the QRE has been approached by the studies of [6,12,5]. These studies have their roots in previous work for other data models [26].

In spite of the advantages presented by the QBE paradigm for non-specialized users of database technologies, to the best of our knowledge and as reported by [20] there have not been systematic studies approaching this paradigm from a Semantic Web perspective.

3 Method

3.1 Research questions

In accordance with this work's goal, the following research questions have been formulated:

- *RQ1*: Which approaches have been followed to conduct studies on Query Reverse Engineering in the context of the Semantic Web (QRESW)?
  - RQ1.1: Which papers have been published?
  - RQ1.2: In which manner has the number of publications evolved over time?
  - RQ1.3: In which venues papers about QRESW have been mostly published?
- *RQ2*: How has the QRESW process been performed?
  - RQ2.1: Which algorithmic approaches have been followed?
  - RQ2.2: Which algorithms can be seen as an evolution of previously existing algorithms?
  - RQ2.3: Which are the main technologies used to perform the QRESW process?
  - RQ2.4: Which are the principal datasets employed to validate the developed algorithms?
- *RQ3*: Which are the research opportunities in the existing literature with regard to QRESW?

3.2 Search

Several methodologies for systematic mapping research [19,24,18] agree in the PICO (Population, Intervention, Comparison and Outcomes) structure as key elements that need to be specified. Specifically for the software engineering area and this research in particular, these elements can be defined as follows:

- *Population*: In software engineering experiments, "population" might refer to a specific role, a category of engineers, an area of application, or an industry group. In this study's context, it is referred to papers regarding to Query Reverse Engineering in the context of the Semantic Web.
- *Intervention*: In software engineering, "intervention" might refer to a methodology, tool, technology or procedure addressing a particular issue. In our context it refers to the algorithms, technologies and datasets used to perform the Query Reverse Engineering process.
- *Comparison*: In software engineering, "comparison" might refer to a methodology, tool, technology, or procedure which which the intervention (see above) is being compared. We compare the different algorithmic approaches followed to perform the QRESW process, identifying similar aspects and studying their temporal evolution. On the other hand, we make a review of the main tools employed to codify them as computer programs.

– *Outcomes*: In our context, "outcomes" refers to the effort for a user to get a satisfactory query.

The PICO methodology allowed to identify the following keywords: "query reverse engineering", "query by output", "query-by-output", "query by example", "query-by-example", "query inference", "query learning", "SPARQL" and "RDF".

The databases have been selected according to the reports published in [14]. The SCOPUS database omits hyphens so we did not use keywords containing hyphens to search in it. Based on the identified keywords, Table 1 shows the search strings used and Table 2 contains the number of results per database after performing the search.

Table 1. Search strings used per database.

Database	Search
WoS	TS=(("query reverse engineering" OR "query by example" OR "query-by-example" OR "query by output" OR "query-by-output" OR "query inference" OR "query learning") AND ("SPARQL" or "RDF"))
SCOPUS	ALL (("query reverse engineering" OR "query by example" OR "query-by-example" OR "query by output" OR "query-by-output" OR "query inference" OR "query learning") AND ("SPARQL" OR "RDF"))
ACM	[[All: "query reverse engineering"] OR [All: "query by example"] OR [All: "query-by-example"] OR [All: "query by output"] OR [All: "query-by-output"] OR [All: "query inference"] OR [All: "query learning"]] AND [All: "sparql" or "rdf"]
ScienceDirect	(("query reverse engineering" OR "query by example" OR "query-by-example" OR "query by output" OR "query-by-output" OR "query inference" OR "query learning") AND ("SPARQL" or "RDF"))

Table 2. Number of search results per database.

Database	Search results
WoS	10
SCOPUS	354
ACM	102
ScienceDirect	200

3.3 Study selection

Some articles were excluded on the basis of titles and abstracts, while other articles were required full-text reading and quality assessment. "Backward snowball sampling" [17] was used to identify other studies that has not been previously included. The following inclusion criteria were applied to titles and abstracts:

- Studies in the field of Semantic Web technologies.
- Studies with regard to Query Reverse Engineering.
- Studies published between 2008 (the SPARQL language becomes a recommendation of the World Wide Web Consortium) and 2019 (the current study having been carried out in 2020).

The criteria below were used to exclude studies:

- Studies must present peer-reviewed materials.
- Studies must be presented in English or Spanish.
- Studies must be accessible as full text.
- Studies should not be duplicates of previous studies.

Figure 1 shows the number of articles included in the study at each stage of the selection process.

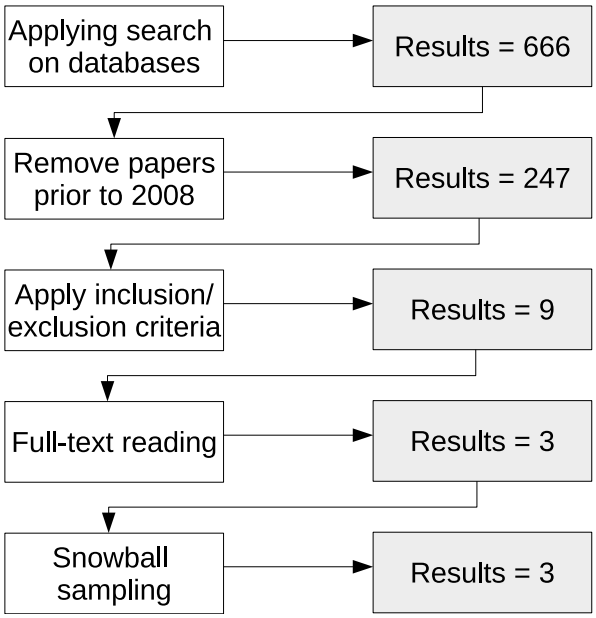

Fig. 1. Number of articles included in the study at each stage of the selection process.

Full-text reading allowed to identify further articles that should be removed as they were out of scope. We note that the remaining articles (3) were then

used for "backward snowball sampling" [17]. This allowed us to conclude that there are not other known studies regarding the topic.

3.4 Data extraction process

Table 3 shows the developed template to conduct the data extraction from the identified primary studies.

Table 3. Data extraction template.

Item	RQ
Study ID	
Article title	RQ1.1
Authors	
Publication year	RQ1.2
Publication venue	RQ1.3
Algorithmic approach	RQ2.1
Relations among studies	RQ2.2
Technologies used to develop the algorithms	RQ2.3
Datasets used to perform the reverse engineering process	RQ2.4
Remaining gaps in the field	RQ3

4 Results of the mapping

4.1 Papers with regard to QRESW (RQ1.1)

During the time frame considered in this study three papers have been published with regard to Query Reverse Engineering in the context of the Semantic Web. Their titles are as follows:

- "Reverse engineering SPARQL queries" [6].
- "SPARQLByE: Querying RDF data by example" [12].
- "Interactive inference of SPARQL queries using provenance" [5].

4.2 Temporal evolution of the QRESW (RQ1.2)

The temporal evolution of research about QRESW is shown in Figure 2. It is worthwhile to mention that the first two publications were partial results of a doctoral thesis presented in 2018 [13].

4.3 Publication Venues (RQ1.3)

In this study, we only considered peer-reviewed venues (not only journals, but also peer-reviewed workshops and conferences). In Table 4, it is shown how the articles are distributed over these venues. Some of the main venues in the research field have been selected to publish the studies, emphasizing the high importance of this research.

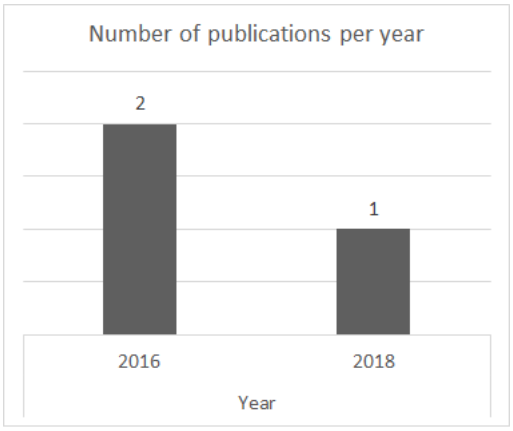

Fig. 2. Temporal evolution of the studies on QRESW.

Table 4. Publication venues.

Venue	Studies
VLDB 2016	[12]
WWW'16	[6]
ICDE 2018	[5]

4.4 Algorithmic approaches (RQ2.1)

All the studies on QRESW take a greedy approach to learn the queries. This kind of heuristics make at each stage a locally optimal choice with the intent of finding eventually a global optimum. While this approach allows to solve the problem in a reasonable amount of time, it does not guarantee that a global optimum is found, leaving space to new research with regard to query optimization.

4.5 Algorithmic evolution (RQ2.2)

Notwithstanding the novelty of the application field and the adaptations made, it could be possible to backtrack some of the ideas involved in the studies of [6,12] to the procedures followed by [26] during the implementation of the FOIL system.

4.6 Technologies (RQ2.3)

While conducting this study, it was noted that the Java language, the Apache Jena framework as well as the Virtuoso Open Source triplestore were involved in the totality of the papers identified. This allows to take them into account as references for future studies.

4.7 Datasets (RQ2.4)

Four main datasets were used in the identified studies. Table 5 shows the relation between the studies and the datasets:

- DBpedia and DBpedia Query Logs are general purpose datasets containing data extracted from Wikipedia. They offer a SPARQL endpoint available for querying [2] and its query logs are available online [3].
- The SP2B dataset is a benchmark [1] for SPARQL performance. It is integrated in the DBLP scenario comprising both a data generator to create DBLP-like queries of arbitrary size, as well as a collection of benchmark queries.
- The "Berlin SPARQL Benchmark" (BSBM) [4] provides a suite of benchmarks to compare performance of such systems across different architectures.

Table 5. Datasets used to conduct the identified studies.

Dataset	Studies
DBpedia	[5]
DBpedia query logs	[6,12]
SP2B	[5]
BSBM	[5]

4.8 Research opportunities

This study allowed us to identify the following existing research opportunities in the field:

- Reverse engineering SPARQL queries for a larger fragment of the language (e.g. involving more than the AND, OPTIONAL and FILTER operators).
- Derive instance equivalent queries with an increased load of meaningful information in the derived triple patterns.
- Ranking of the derived patterns to increase the expressiveness of the query.
- Usage of projections to discover queries from entities that are not directly related.
- Proposal of frameworks mixing QRESW with machine learning data completion algorithms to explore both explicitly stated and learnt data.

5 Conclusions and future work

Notwithstanding query reverse engineering on relational databases having been an active research field for several years, it has only started recently in the context

of the Semantic Web. To the best of our knowledge, only three studies within the time frame considered here have covered this topic in this context. They have been published in some of the top conferences in the fields of Databases and Web Research. These studies have approached the QRESW problem using greedy algorithmic approaches, leaving space for deeper research on query optimization. Moreover, some of the ideas behind these algorithms can be traced back to similar ones from the 1990, adapted to the new context.

In accordance with this new context, the technologies employed to implement the algorithms are Java as programming language, the Apache Jena as framework for semantic SPARQL queries as well as Virtuoso Open Source to store the RDF graphs. Besides that, the main data set used in the different studies to validate the developed algorithms is DBpedia, followed by the SP2B and BSBM data sets.

Query Reverse Engineering in the context of the Semantic Web remains as a promising research field. Among other ones, enlarging the fragments of the SPARQL language considered so far is an interesting area for further exploration.

6 Acknowledgments

The authors would like to acknowledge to BOF for its support to the current research through the Research Project R-10405.

References

1. http://dbis.informatik.uni-freiburg.de/index.php?project=SP2B/download.php
2. http://dbpedia.org/sparql
3. https://aksw.github.io/LSQ/
4. http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/
5. Abramovitz, E., Deutch, D., Gilad, A.: Interactive inference of sparql queries using provenance. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE). pp. 581–592. IEEE (2018)
6. Arenas, M., Diaz, G.I., Kostylev, E.V.: Reverse engineering sparql queries. In: Proceedings of the 25th International Conference on World Wide Web. pp. 239–249 (2016)
7. Barreiros, E., Almeida, A., Saraiva, J., Soares, S.: A systematic mapping study on software engineering testbeds. In: 2011 International Symposium on Empirical Software Engineering and Measurement. pp. 107–116. IEEE (2011)
8. Bates, S., Clapton, J., Coren, E.: Systematic maps to support the evidence base in social care. Evidence & Policy: A Journal of Research, Debate and Practice 3(4), 539–551 (2007)
9. Clapton, J., Rutter, D., Sharif, N.: Scie systematic mapping guidance. London: SCIE (2009)
10. Condori-Fernandez, N., Daneva, M., Sikkel, K., Wieringa, R., Dieste, O., Pastor, O.: A systematic mapping study on empirical evaluation of software requirements specifications techniques. In: 2009 3rd International Symposium on Empirical Software Engineering and Measurement. pp. 502–505. IEEE (2009)

11. Coren, E., Fisher, M.: The conduct of systematic research reviews for scie knowledge reviews (2006)
12. Diaz, G., Arenas, M., Benedikt, M.: Sparqlbye: Querying rdf data by example. Proceedings of the VLDB Endowment 9(13), 1533–1536 (2016)
13. Diaz-Caceres, G.: Increasing the usability of graph databases by learning SPARQL queries and RDF data. Ph.D. thesis, University of Oxford (2018)
14. Dyba, T., Dingsoyr, T., Hanssen, G.K.: Applying systematic reviews to diverse study types: An experience report. In: First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007). pp. 225–234. IEEE (2007)
15. Grant, M.J., Booth, A.: A typology of reviews: an analysis of 14 review types and associated methodologies. Health Information & Libraries Journal 26(2), 91–108 (2009)
16. Jalali, S., Wohlin, C.: Agile practices in global software engineering-a systematic map. In: 2010 5th IEEE International Conference on Global Software Engineering. pp. 45–54. IEEE (2010)
17. Jalali, S., Wohlin, C.: Systematic literature studies: database searches vs. backward snowballing. In: Proceedings of the 2012 ACM-IEEE international symposium on empirical software engineering and measurement. pp. 29–38. IEEE (2012)
18. James, K.L., Randall, N.P., Haddaway, N.R.: A methodology for systematic mapping in environmental sciences. Environmental evidence 5(1), 7 (2016)
19. Kitchenham, B., Charters, S., et al.: Guidelines for performing systematic literature reviews in software engineering version 2.3. Engineering 45(4ve), 1051 (2007)
20. Martins, D.M.L.: Reverse engineering database queries from examples: State-of-the-art, challenges, and research opportunities. Information Systems 83, 89–100 (2019), https://doi.org/10.1016/j.is.2019.03.002
21. Mottin, D., Lissandrini, M., Velegrakis, Y., Palpanas, T.: New trends on exploratory methods for data analytics. Proc. VLDB Endow. 10(12), 1977–1980 (Aug 2017), https://doi.org/10.14778/3137765.3137824
22. Oakley, A., Gough, D., Oliver, S., Thomas, J.: The politics of evidence and methodology: lessons from the eppi-centre. Evidence & Policy: A Journal of Research, Debate and Practice 1(1), 5–32 (2005)
23. Peersman, G.: A descriptive mapping of health promotion in young people. University of London (1996)
24. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology 64, 1 – 18 (2015), http://www.sciencedirect.com/science/article/pii/S0950584915000646
25. Qadir, M.M., Usman, M.: Software engineering curriculum: a systematic mapping study. In: 2011 Malaysian Conference in Software Engineering. pp. 269–274. IEEE (2011)
26. Quinlan, J.R.: Learning logical definitions from relations. Machine learning 5(3), 239–266 (1990)
27. Tran, Q.T., Chan, C.Y., Parthasarathy, S.: Query reverse engineering. The VLDB Journal 23(5), 721–746 (2014), https://doi.org/10.1007/s00778-013-0349-3
28. Weiss, Y.Y., Cohen, S.: Reverse engineering spj-queries from examples. In: Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. p. 151–166. PODS '17, Association for Computing Machinery, New York, NY, USA (2017), https://doi.org/10.1145/3034786.3056112
29. Zloof, M.M.: Query-by-example: The invocation and definition of tables and forms. In: Proceedings of the 1st International Conference on Very Large Data Bases.

p. 1–24. VLDB '75, Association for Computing Machinery, New York, NY, USA (1975), https://doi.org/10.1145/1282480.1282482