How (Well) Do Datalog, SPARQL and RIF Interplay?

Axel Polleres

Siemens AG Österreich, Siemensstraße 90, 1210 Vienna, Austria
axel.polleres@siemens.com

Abstract. In this tutorial we will give an overview of the W3C standard query language for RDF – SPARQL – and its relation to Datalog as well as on the interplay with another W3C standard closely related to Datalog, the Rule Interchange Format (RIF). As we will learn – while these three interplay nicely on the surface and in academic research papers – some details within the W3C specs impose challenges on seamlessly integrating Datalog rules and SPARQL.

1 SPARQL Official Semantics vs. Academia

The formal semantics of SPARQL in its original recommendation in 2008 [19] has been very much inspired by academic results, such as by the seminal papers of Pérez et al. [13,14]. Angles and Gutierrez [1] later showed that SPARQL – as defined in those papers – has exactly the expressive power of non-recursive safe Datalog with negation. Another translation from SPARQL to Datalog has been presented in [15]. In the tutorial we will present the semantics of SPARQL, starting with the semantics as per [14] as well as its translation to Datalog; we will then discuss adaptions needed to be considered with regards to the official W3C specification, particularly:

1. SPARQL's multi-set semantics and solution modifiers
2. the treatment of complex expressions and errors in FILTERs
3. the treatment of FILTER expressions in OPTIONAL

Let us sketch briefly some reasons how these features affect the translation.

SPARQL's multi-set semantics. While Datalog is set-based, SPARQL queries – just like SQL – allow for duplicates in solutions. Since duplicates can stem from only certain patterns, the translation to Datalog can be "fixed" to cater for these; however, the translation becomes less elegant [18]. Likewise, solution modifiers such as ORDER BY and LIMIT/OFFSET in SPARQL have no straightforward equivalent within Datalog. Notably, multi-set semantics has been considered in some earlier works about Datalog [12].

Complex expressions and errors in FILTERs. Unlike the semantics given in [13], FILTERS in SPARQL have a 3-valued semantics for connectives such as "&&" and "||", to cater for errors. We will give some examples where these make sense and discuss how the translation to Datalog can be adapted.

P. Barceló and R. Pichler (Eds.): Datalog 2.0, LNCS 7494, pp. 27–30, 2012.

FILTERs in OPTIONALs. Another specialty of the SPARQL semantics, as noted by [1] outer joins in SPARQL – denoted by the OPTIONAL keyword – are not compositional due to the fact that certain "non-safe" FILTERs are possible, i.e., it is allowed that FILTERs refer to variables bound outside the OPTIONAL pattern. While a rewriting of SPARQL queries with safe FILTERS only is possible [1], a translation from SPARQL to Datalog could also cater for this semantics directly.

2 SPARQL in Combination with Rules

In the second part of the tutorial, we will have a closer look at using SPARQL itself as a rules language in the spirit of Datalog, from academic approaches [20,17] over practically motivated & implemented ones – such as SPIN [9] and R2R [4] – to combinations within the W3C standards themselves, namely SPARQL in combination with RIF. As for the latter, we will discuss both (i) whether or why not the translation from SPARQL to Datalog as per [15] works with RIF [5,16] and (ii) what a SPARQL query means in combination with a RIF rule set [8].

3 New Features in SPARQL1.1

Finally, we will discuss and sketch how translations to or a combinations with Datalog style rules could be extended to new features of the upcoming SPARQL 1.1 recommendation [21], namely:

Aggregate functions. Aggregate functions will allow operations on the query engine side such as counting, numerical min/max/average and so on, by operating over columns of results. This feature is commonly known from other query languages such as SQL, but also well investigated in terms of extensions of Datalog, cf. for instance [7]. A proposal to extend SPARQL with aggregates following these ideas for Datalog has been made in [17], whereas the SPARQL1.1 working group rather follows the SQL design.

Subqueries. This feature will allow nesting the results of a SPARQL query within another query. The SPARQL1.1 specification will only allow very limited subqueries, whereas a discussion of further options for subqueries within SPARQL has been presented by Angles and Gutierrez [2]; again we will discuss where Datalog fits in the picture.

Project expressions. This feature will allow one to compute values from expressions within queries, rather than just returning terms appearing in the queried RDF graph; built-ins within Datalog provide similar functionality.

Property paths. Many classes of queries over RDF graphs require traversing hierarchical data structures and involve arbitrary-length paths. While such queries over graph-based structures can be naturally expressed in languages like Datalog, is was not possible to express such queries using the original

SPARQL recommendation. The ability to formulate certain path queries has now been added in SPARQL1.1 and again the design choices have been influenced by discussions in academia [3,10].

Inferred results under different Entailment Regimes. The [8] specifies various entailment regimes for SPARQL, particularly for RDF Schema, OWL, and RIF; apart from the above-mentioned combination of SPARQL with RIF, we will particularly discuss those entailment regimes that are most closely related to Datalog, i.e. those based on the OWL fragments OWL 2 RL (which essentially includes RDF Schema [6]) and OWL 2 QL [11].

Acknowledgements. The presenter – who has been actively contributing to both the RIF and SPARQL specifications – would like to thank all the members of the RIF[1] and SPARQL[2] W3C working groups.

References

1. Angles, R., Gutierrez, C.: The Expressive Power of SPARQL. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 114–129. Springer, Heidelberg (2008)
2. Angles, R., Gutierrez, C.: Subqueries in SPARQL. In: Proceedings of the 5th Alberto Mendelzon International Workshop on Foundations of Data Management (AMW2011), Santiago, Chile. CEUR Workshop Proceedings, vol. 749. CEUR-WS.org (May 2011)
3. Arenas, M., Conca, S., Pérez, J.: Counting beyond a yottabyte, or how sparql 1.1 property paths will prevent adoption of the standard. In: Proceedings of the 21st World Wide Web Conference (WWW 2012), Lyon, France, pp. 629–638. ACM (April 2012)
4. Bizer, C., Schultz, A.: The R2R framework: Publishing and discovering mappings on the web. In: 1st International Workshop on Consuming Linked Data (COLD 2010), Shanghai, China (November 2010)
5. Boley, H., Hallmark, G., Kifer, M., Paschke, A., Polleres, A., Reynolds, D.: RIF Core Dialect. W3C recommendation, W3C (June 2010), http://www.w3.org/TR/rif-core/
6. Brickley, D., Guha, R., McBride, B. (eds.): RDF Vocabulary Description Language 1.0: RDF Schema. Technical report, W3C (February 2004), W3C Recommendation.
7. Faber, W., Leone, N., Pfeifer, G.: Recursive Aggregates in Disjunctive Logic Programs: Semantics and Complexity. In: Alferes, J.J., Leite, J. (eds.) JELIA 2004. LNCS (LNAI), vol. 3229, pp. 200–212. Springer, Heidelberg (2004)
8. Glimm, B., Ogbuji, C., Hawke, S., Herman, I., Parsia, B., Polleres, A., Seaborne, A.: SPARQL 1.1 Entailment Regimes. W3C working draft, W3C (January 2012), http://www.w3.org/TR/sparql11-entailment/
9. Knublauch, H., Hendler, J.A., Idehen, K.: SPIN - Overview and Motivation (February 2011); W3C member submission

[1] http://www.w3.org/2005/rules/wiki/RIF_Working_Group
[2] http://www.w3.org/2009/sparql/wiki/Main_Page

10. Losemann, K., Martens, W.: The complexity of evaluating path expressions in sparql. In: Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2012), Scottsdale, AZ, USA, pp. 101–112. ACM (May 2012)
11. Motik, B., Grau, B.C., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C., Calvanese, D., Carroll, J., De Giacomo, G., Hendler, J., Herman, I., Parsia, B., Patel-Schneider, P.F., Ruttenberg, A., Sattler, U., Schneider, M.: OWL 2 Web Ontology Language Profiles. W3C recommendation, W3C (October 2009), http://www.w3.org/TR/owl2-profiles/
12. Mumick, I.S., Shmueli, O.: Finiteness properties of database queries. In: 4th Australian Database Conference (1993)
13. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and Complexity of SPARQL. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 30–43. Springer, Heidelberg (2006)
14. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. ACM Transactions on Database Systems, 34(3), Article 16, 45 pages (2009)
15. Polleres, A.: From SPARQL to rules (and back). In: Proceedings of the 16th World Wide Web Conference (WWW 2007), Banff, Canada, pp. 787–796. ACM Press (May 2007)
16. Polleres, A., Boley, H., Kifer, M.: RIF Datatypes and Built-Ins 1.0. W3C recommendation, W3C (May 2010), http://www.w3.org/TR/2010/rif-dtb/
17. Polleres, A., Scharffe, F., Schindlauer, R.: SPARQL++ for Mapping Between RDF Vocabularies. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 878–896. Springer, Heidelberg (2007)
18. Polleres, A., Schindlauer, R.: dlvhex-sparql: A SPARQL-compliant query engine based on dlvhex. In: 2nd International Workshop on Applications of Logic Programming to the Web, Semantic Web and Semantic Web Services (ALPSWS 2007), Porto, Portugal. CEUR Workshop Proceedings, vol. 287, pp. 3–12. CEUR-WS.org. (September 2007)
19. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3c recommendation, W3C (January 2008), http://www.w3.org/TR/rdf-sparql-query/
20. Schenk, S., Staab, S.: Networked graphs: A declarative mechanism for sparql rules, sparql views and rdf data integration on the web. In: Proceedings WWW 2008, Beijing, China, pp. 585–594. ACM Press (2008)
21. Seaborne, A., Harris, S.: SPARQL 1.1 Query Language. W3C working draft, W3C (January 2012), http://www.w3.org/TR/sparql11-query/