Expressive OWL Queries: Design, Evaluation, Visualization

Petr Křemen, Department of Cybernetics, Czech Technical University in Prague, Prague, Czech Republic

Bogdan Kostov, Department of Cybernetics, Czech Technical University in Prague, Prague, Czech Republic

ABSTRACT

Web Ontology Language ontologies become more and more popular in complex domain modeling for their high expressiveness, flexibility and well defined semantics. Although query languages adequate in expressiveness to OWL reasoning capabilities were introduced before, their implementations are rather limited. In this paper, the authors study SPARQL-DL^NOT, an extension of one of these query languages, SPARQL-DL, and present novel evaluation and optimization techniques for efficient SPARQL-DL^NOT execution. As queries become complex easily, they also present a novel graph-based visualization that simplifies query construction and maintenance. Presented techniques and algorithms were implemented in the Pellet reasoner and in their novel Protégé plug-in OWL2Query.

Keywords: Conjunctive Queries, Evaluation, Ontology, Optimization, Visualization, Web Ontology Language - Second Edition (OWL2)

INTRODUCTION

Nowadays, semantic web ontologies, and Web Ontology Language OWL 2 (Motik, Patel-Schneider, & Parsia, 2009) in particular, are first-choice means for expressing complex structured knowledge on the semantic web as well as domain applications. They bring a different view to the knowledge representation than traditional relational database technology, namely open world reasoning (i.e. whatever is not known to be true, is not necessarily false) and expressive meta-modeling (i.e. class/property taxonomies, class/property constructors, transitivity, inverse roles, etc.).

DOI: 10.4018/jswis.2012100104

Unfortunately, although OWL (Patel-Schneider, Hayes, & Horrocks, 2004), standardized as a W3C Recommendation in 2004, and OWL 2, standardized as a W3C Recommendation in 2009, stemmed from a relatively rapid standardization process based on current research in the field of expressive description logics (Horrocks, Kutz, & Sattler, 2006), this was not the case for queries to OWL. Thus, although ontology designers spent significant effort to represent the domain knowledge precisely using expressive ontological constructs, standardized means for retrieving the knowledge efficiently were limited to simple queries, like consistency checks, subsumption checks, instance checks or instance retrievals. Expressive query languages with varying features were introduced during
last few years. In this paper, we extend one of these languages, SPARQL-DL (Sirin & Parsia, 2007) with additional OWL 2 constructs and negation as failure (feature that allows to use the query language not only for query answering but also for integrity constraints checking (Tao, Sirin, Bao, & McGuinness, 2010) and present evaluation and visualization technique.

Section Related Work reviews state-of-the-art query languages for OWL and relates them to our approach. Section Preliminaries reviews the OWL, OWL 2 and SPARQL-DL languages to the extent necessary to understand section SPARQL-DL languages in which our extension to SPARQL-DL is introduced. In section Evaluating SPARQL-DL, we present a query evaluation algorithm for SPARQL-DL together with a novel dynamic query reordering optimization. Easy-to-understand graph-based query visualization of SPARQL-DL is presented in Section Visualization. Both the query engine and query visualization were implemented in our novel OWL2Query system.

RELATED WORK

During last years several conjunctive query and meta-query languages were introduced and implemented on top of existing OWL reasoners. Most of the efforts have been spent on conjunctive ABox queries (i.e. queries that retrieve only ABox individuals). Authors of (Horrocks & Tessaris, 2000) propose methods for conjunctive ABox query answering in the ALC language (Baader, Calvanese, McGuinness, Nardi, & Patel-Schneider, 2003) that transforms a conjunctive query answering problem to instance checking or instance retrieval problem. Although for ALC the proposed technique works fine, the authors noticed that its generalization is problematic for expressive description logics backing OWL or OWL 2 and is possible only for queries that do not contain cyclic structures of undistinguished variables (see section Preliminaries for more details). Evaluation and optimization of conjunctive ABox queries in expressive description logics are discussed also in Sirin and Parsia (2006), Ortiz, Calvanese, and Eiter (2006) and in Dolby et al. (2008).

Conjunctive ABox queries allow retrieving individuals and literals from an OWL ontology, thus being similar to SQL query language for relational databases. However, OWL ontologies contain also significant amount of knowledge in TBox and RBox to model taxonomies and complex characteristics of classes (disjointness, equivalence, etc.) or properties (transitivity, functionality). To address this issue, query languages SPARQL-DL, SQWRL (O’Connor & Das, 2009) and OWL-SAIQL (Kubias, Schenk, Staab, & Pan, 2007) appeared during last years that allow evaluating mixed ABox, TBox and RBox queries to retrieve individuals, classes, and properties.

SQWRL is a SWRL-based query language (Horrocks et al., 2004) that allows posing queries by combining any OWL axiom or SWRL atom, in which individual/class/property names can be replaced by distinguished variables. Comparing to SPARQL-DL and OWL-SAIQL, SQWRL allows also formulating aggregate queries.

OWL-SAIQL allows for retrieving also named classes as well as complex class descriptions, comparing to SQWRL and SPARQL-DL.

SPARQL-DL allows for both distinguished variables and undistinguished variables in queries and provides native SPARQL (Prud’hommeaux & Seaborne, 2008) syntax. Distinguished variables bind individuals/classes/properties that are returned in the result set, while undistinguished variables match domain elements that need not be interpretations of individuals in the queried ontology. Since undistinguished variables are not necessarily bound to existing individuals in the knowledge base, their match is not part of the result set.

Example 1. The need for undistinguished variables is demonstrated in (Křemen & Kouba, 2011) by $Q_4$ of the following form1
Domain-Specific Ontologies Trading for Retrieval and Integration of Information in Web-Based Information Systems
www.igi-global.com/chapter/domain-specific-ontologies-trading-retrieval/52867?camid=4v1a

A URI is Worth a Thousand Tags: From Tagging to Linked Data with MOAT
www.igi-global.com/article/uri-worth-thousand-tags/37499?camid=4v1a