GLOO: A Graphical Query Language for OWL Ontologies

Amineh Fadhil and Volker Haarslev

Concordia University, Montreal, Quebec, Canada
{f_amineh,haarslev}@cse.concordia.ca

Abstract. The database usability experience has shown that visual query languages tend to be superior to textual languages in many aspects. By applying this principle in the context of ontologies, we present GLOO, a graphical query language for OWL-DL ontologies. GLOO maps diagrammatic queries to DL based query languages such as nRQL, which is offered by the OWL-DL reasoner Racer. GLOO hides the complexity of a DL query language from users and allows them to query OWL ontologies with less difficulty.

1 Introduction

Ontologies play an increasingly important role in the domain of knowledge management and information systems. Although progress has been made in supporting ontology editing and visualizing, not much has been accomplished for non-expert or naïve users wishing to query an ontology. Ideally, queries should be easy to design in order to facilitate the task for domain experts, who often lack necessary technical skills. However, the current state of the art requires one to submit a textual, description logic (DL) or SQL-like query to an OWL reasoner. The logic and syntax of these querying languages obviously necessitates a tedious effort from users before being able to reach a successful level to write query effectively. The importance of an effective and easy to use query system has already been recognized in the database area. Experimental evaluations of visually querying a database confirm this belief [1, 2]. In this paper, we propose a visual query system (VQS) equipped with GLOO, a formal graphical query language (VQL) for OWL-DL ontologies.

2 The Visual Query Language

2.1 Basic Visual Notation

The basic elements of GLOO are illustrated in Table 1. A concept (or class) is represented by its name in a filled oval, an individual by its name in a filled rectangle, a role (or property) by its name and an arrow pointing from one concept to another one.

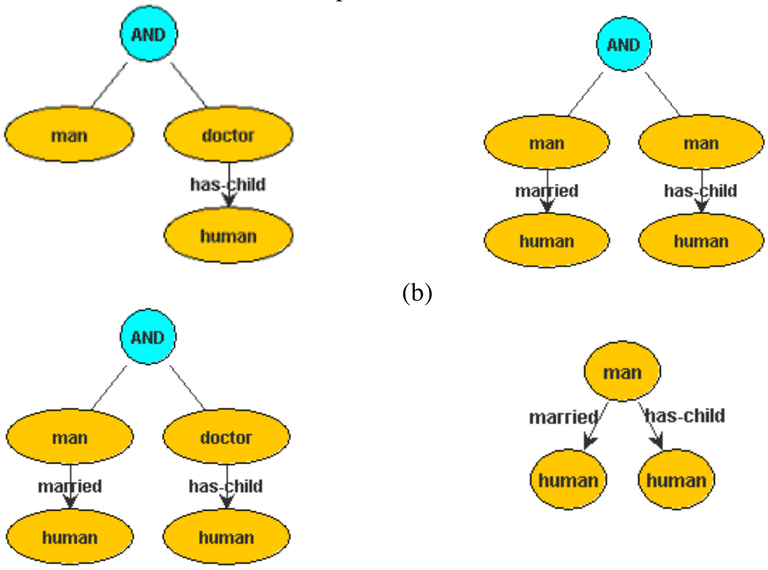
The applicable operators for these building blocks are negation, complement, conjunction (or intersection) and disjunction (or union). The difference between negation and complement relies on the fact that the former represents true classical negation of concepts whereas the latter designates negation as failure (NAF). The semantics of negation is to find individuals or pairs of individuals that are instances of the negated concept or role respectively. On the other hand, the semantics of complement is to find individuals or pairs of individuals that are not known to be instances of the complemented concept or role. On that account, negation is graphically illustrated by the "NOT" keyword while complement is distinguished by the "NOT KNOWN" keyword as shown in Table 1 for the visual elements <concept-negated>, <role-negated>, <concept-complemented>, and <role-complemented>. A role can have as domain and range any combination of the concept elements.

Table 1. Visualization of the basic elements of GLOO.

Basic Elements of GLOO	Visual Representation
<concept>	concept
<concept-negated>	NOT concept
<concept-complemented>	NOT KNOWN TO BE concept
<individual>	individual
<role>	role
<role-negated>	NOT role
<role-complemented>	NOT KNOWN TO role

2.2 Query Formulation with Conjunction and Disjunction

Basic operators that are commonly expected to be available for querying are conjunction and disjunction. In GLOO, they are graphically depicted by a special node labeled with the keyword 'AND' and 'OR' respectively. The 'AND' keyword unambiguously leads the user to the conclusion that the concepts connected to the 'AND' node are intersected. In fact, the diagram shown in Figure 1(a) illustrates a query asking for instances of man who are doctors and have a child. Users may combine two queries with these operators. For instance, one may first be looking for married men by constructing the 'man-married-human' query and then wishing to know those men who have a child by constructing the 'man-has_child-human' query and later on be interested in finding out those men who are also married and have a child by connecting the two previous queries with the 'AND' node as shown in Figure 1(b). Note that this query is intersecting two identical concepts and is equivalent to the simplified one shown in Figure 1(d) where these concepts were merged. Thus, this simplified query, although not explicitly using the 'AND' operator, is implicitly a conjunctive query. The merging is not possible for the query shown in Figure 1(c) because the intersection is performed over distinct concepts and thus the semantics of the query would be modified into asking for men who are doctors that happen to have a child and are married. These examples demonstrate that users can start by composing simple queries and then produce more complex, novel ones by connecting them with the intersection and union operators.



(a) (b) (c) (d)

Fig. 1. Visualization of conjunctive queries.

2.3 Mapping the VQL to nRQL

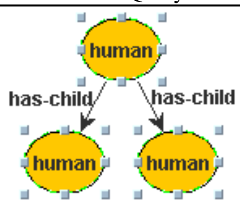
We claim that GLOO is formal because it is semantically and syntactically unambiguous given that its 'connectivity syntax' is formally defined by a grammar. The term 'connectivity syntax' is meant to refer to how the graph components are linked together. In fact, the visual layout of a query does not affect its semantics but the connectivity of its components directly impacts on its meaning.

The expressive power of GLOO can be informally described by being able to formulate queries on DL ABox elements (concept and role assertions) and make use of conventional operators (negation, complement, union and intersection) for building up more complex, refined queries. Our proposed VQL is independent of any OWL-query language and offers basic functionalities for querying OWL-DL ontologies. However, even though GLOO is mapped to nRQL, it does not match nRQL's full expressive power. In fact, our VQL hides the complexity of nRQL's syntax by mainly hiding query variables but this also causes the impossibility of unambiguously visualizing some queries.

nRQL [3] is implemented by an optimized OWL-DL query processor known to be highly effective and efficient. A nRQL query is composed of a query head and a query body. The query body consists of the query expression whereas the query head corresponds to the variables mentioned in the body that will be bound to the result.

Visually, the query head is symbolized by the selection of the concepts whose instances are to be part of the result. The building blocks of nRQL necessary for the semantic ‘translation’ of GLOO are ‘concepts query atoms’ and ‘role query atoms’. These query atoms can be observed in the query of Table 2 that depicts the conjunction of two role query atoms and three concept query atoms.

Table 2. Example of a graphical query and its equivalent in nRQL.

Visual Query	Corresponding nRQL query
	(Retrieve (x1 x2 x3) (and (x1 x2 has-child) (x1 x3 has-child) (x1 human) (x2 human) (x3 human)))

GLOO is particularly useful in representing queries searching for complex role filler graph structures in an ontology. For instance, the query shown in Table 2 is searching for children having a common parent.

3 Conclusion and Future Work

In conclusion, we proposed a VQL for OWL-DL ontologies hiding the complexity of a textual query language such as nRQL. The visual simplification is twofold, first by eliminating the textual syntax with the VQL and not allowing syntactic errors through the user interface (UI); second by assisting users in the querying process through the VQS features such as providing immediate feedback with result cardinalities. The implementation of the UI is underway. Part of the VQL has been already implemented. Work is still under progress for completing GLOO’s VQS.

References

1 Catarci, T., Santucci, G.: Are Visual Query Languages Easier to Use then Traditional Ones? An Experimental Proof. In Proceedings of the tenth Conference of the British Computer Society Human Computer Interaction Specialists Group-People and Computers X. August, 1995, Huddersfield, UK. P.323-338.

2 Catarci, T., Santucci, G.: Diagrammatic vs Textual Query Languages: A Comparative Experiment. In Proc. Of the IFIP W.G.2.6 Working Conference on Visual Databases, March 1995.

3 Haarslev, V., Möller, R., Wessel, M.: Querying the Semantic Web with Racer + nRQL. CEUR Workshop Proceedings of KI-2004 Workshop on Applications of Description Logics (ADL 04), Ulm, Germany, Sep 24 2004.