Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs

Esther Kaufmann, Abraham Bernstein, and Renato Zumstein
University of Zurich
Department of Informatics
Binzmuehlestrasse 14, CH-8050 Zurich, Switzerland
{kaufmann, bernstein}@ifi.unizh.ch

Abstract

The logic-based machine-understandable framework of the Semantic Web typically challenges casual users when they try to query ontologies. An often proposed solution to help casual users is the use of natural language interfaces. Such tools, however, suffer from one of the biggest problems of natural language: ambiguities. Furthermore, the systems are hardly adaptable to new domains. This paper addresses these issues by presenting *Querix*, a domain-independent natural language interface for the Semantic Web. The approach allows queries in natural language, thereby asking the user for clarification in case of ambiguities. The preliminary evaluation showed good retrieval performance.

1 Introduction

The logic-based underpinnings of the Semantic Web provide a stable scaffolding for machine-based processing. Common users, however, are typically challenged with formal logic. The result is a gap between the Semantic Web and the average user who is mostly unable to command formal logic. The gap manifests itself in a disconnection between the user's information needs and the query language with which the user tries to find the required information in ontologies [Spink *et al.*, 2001; Thompson *et al.*, 2005]. Nevertheless, querying is a major interaction mode with the Semantic Web; bridging it is, therefore, central for the success of the Semantic Web for end users.

This paper proposes to address the gap by presenting *Querix*, a domain-independent natural language interface (NLI) that uses clarification dialogs to query ontologies. The approach is simple and does not use any complex semantics-based technologies. Compared to a full natural language processing (NLP) engine, it does not try to resolve natural language (NL) ambiguities, but asks the user for clarification. The user acts the role of the druid Getafix (hence the name *Querix*) who is consulted by Asterix and the other villagers whenever anything strange occurs. The person composing a query benefits from the clarification dialog by better retrieval results and, even more important, by being relieved from the cognitive burden of learning a formal query language [Chakrabarti, 2004]. Querix does not claim to be "intelligent" by interpreting and understanding the input queries; it employs a reduced set of NLP tools and consults the user when hitting its limitations. It is the simplicity, however, that makes the interface completely portable.

2 The Querix System

The system consists of seven main parts: a user interface, an ontology manager, a query analyzer, a matching center, a query generator, a dialog component, and an ontology access layer.

The *user interface* allows the user to enter full NL queries and choose the ontology to be queried. After executing a query, it displays the results and the SPARQL query it generated to the user. When an ontology is chosen and loaded into Querix, the *ontology manager* enhances the resources' labels by obtaining synonyms from WordNet. The *query analyzer* employs two auxiliary components. The first component is the Stanford Parser [Klein and Manning, 2002] that provides a syntax tree for the NL query. From this syntax tree the query analyzer extracts the sequence of the main word categories *Noun* (N), *Verb* (V), *Preposition* (P), *Wh-Word* (Q), and *Conjunction* (C). Based on the extracted word categories a query skeleton is generated. Consider, for example, the query "What are the population sizes of cities that are located in California?" According to the query analyzer its query skeleton is `Q-V-N-P-N-Q-V-P-N`. The second query analyzer component is WordNet that provides synonyms for all nouns and verbs in the query's parse tree. (Note that we implemented a cost function in order to obtain the most appropriate synonyms as WordNet usually suggests too many.)

The *matching center* is the core component of Querix. It basically tries to match the query skeleton with the synonym-enhanced triples in the ontology. For each query, the following three steps are performed by the matching center:
(1) The matching center first tries to match the extracted query skeleton with a small set of heuristic patterns (e.g., `Q-V-N` representing "what are the population sizes," `N-P-N` representing "population sizes of cities," etc.). As such, the matching basically identifies subject-property-object patterns of a query. Valid patterns in the query skeleton have to overlap with regard to their first or last word category (e.g., "population sizes") to enable the joining of the triples in step 3.
(2) Next, the matching center searches for all matches between the synonym-enhanced nouns and verbs of the input

In proceedings of the 5th International Semantic Web Conference (ISWC 2006), Athens, GA, November 2006.

query with the resources and their synonyms in the ontology. Each possible match is stored including domain and range information.
(3) Finally, the triple patterns identified by step 1 and the resources found by step 2 are matched. This matching is enabled by storing each word category of the query skeleton together with its NL word form in step 1 and each noun as well as verb of the NL query with its synonyms from WordNet by the query analyzer. Additionally, the matching is controlled by the domain and range information of the ontology. After identifying all possible triples in the sentence skeleton and combining them to the ontology's resources, the *query generator* composes SPARQL queries from the joined triples. As each matching step comprises a cost function, the query generator procudes a ranked list of SPARQL queries.

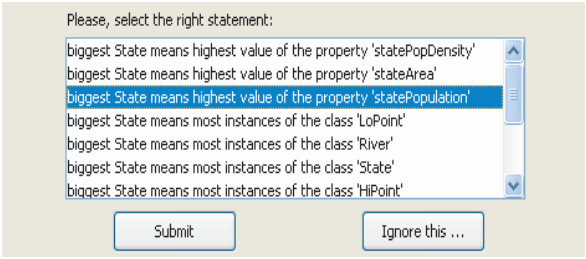

Figure 1: The Querix dialog component consulting the user for clarification

If Querix encounters ambiguities, i.e., several different solutions to a query with the same cost score, its *dialog component* consults the user by showing a menu from which the user can choose the meaning she/he intended (see Figure 1). In this way, the system retrieves the correct answer. The meanings offered by Querix are based on the possible triples identified by the matching center. For example, the compound "population size" can refer to two properties "population" and "populationDensity" in the ontology.

To execute the generated SPARQL query, Querix uses Jena as *ontology access layer* and the Pellet reasoner.

3 Preliminary Evaluation

To evaluate the performance of Querix, we implemented a protoype in Java. We translated a NL knowledge base [Tang and Mooney, 2001] that provides geographical information about the US into OWL and ran 215 queries. These queries syntactically represent the 879 queries that are provided by the data set. As many queries of the data set feature the same syntactic structure (e.g., "How many people live in Chicago?", "How many people live in New York?", ...), we reduced the queries such that each syntactic pattern appeared only once in the data set for our evaluation.

Querix successfully translated all 215 queries into SPARQL queries, thereby achieving 78.6% average recall and 77.67% average precision. The similarity of the values is due to the fact that if Querix generates an appropriate SPARQL query, the correct answer can be retrieved in most cases. After inspecting those queries that did not result in correct SPARQL queries, we found that 8 queries were not correctly analyzed by the Stanford parser, 8 queries were nonsense (e.g., "Which state lies in a city that ...?"), and 5 queries could not be answered due to mistakes in the knowledge base. After removing these queries, Querix achieved an average recall of 87.11% and an average precision of 86.08%. The remaining queries could not be translated appropriately, since we did not implement negation in order to avoid complex semantic analysis.

The results show that the approach is promising. As Querix does not exploit sophisticated logic-based or semantic techniques as typical full-fledged NLP systems do, some queries provided by the data set could not be answered. The dependencies between the words and phrases in the queries are identified by applying two auxiliary NLP tools and pattern matching algorithms that rely on the relationships that exist between the elements in the queried ontology. The approach, therefore, highly depends on the quality and choice of vocabulary of the ontology. This weakness is also Querix' major strength, as it does not need any adaptation for new ontologies, i.e., it is completely portable.

4 Conclusions

If the Semantic Web should be usable by casual users, its logic-based framework needs to be made accessible for querying. We propose that NLIs show a potential for end-user access to the Semantic Web but suffer from ambiguities, for which NL is notorious, and their inapplicability to new domains. To that end, we developed Querix, which is completely portable and asks the user for clarification if ambiguities occur. We believe that our simple, domain-independent approach provides a chance to offer the Semantic Web's capabilities to the general public.[1]

References

[Chakrabarti, 2004] Soumen Chakrabarti. Breaking through the syntax barrier: Searching with entities and relations. In *15th European Conference on Machine Learning (ECML 2004)*, pages 9–16, Pisa, Italy, September 2004.

[Klein and Manning, 2002] Dan Klein and Christopher D. Manning. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems (NIPS 2002)*, pages 3–10, 2002.

[Spink *et al.*, 2001] Amanda Spink, Wolfram Dietmar, Jansen Major B.J., and Saracevic Tefko. Searching the web: The public and their queries. *Journal of the American Society for Information Science and Technology*, 52(3):226–134, 2001.

[Tang and Mooney, 2001] Lappoon R. Tang and Raymond J. Mooney. Using multiple clause constructors in inductive logic programming for semantic parsing. In *12th European Conference on Machine Learning (ECML-2001)*, pages 466–477, Freiburg, Germany, 2001.

[Thompson *et al.*, 2005] Craig W. Thompson, Paul Pazandak, and Harry R. Tennant. Talk to your semantic web. *IEEE Internet Computing*, 9(6):75–78, 2005.

[1]This work was partially supported by the Swiss National Science Foundation (200021-100149/1).