A natural language ontology-driven query interface

A natural language ontology-driven query interface

Enrico Franconi and **Paolo Guagliardo** and **Sergio Tessaris**
KRDB Research Centre, Free University of Bozen-Bolzano, Italy
lastname@inf.unibz.it

Marco Trevisan
CELI Language & Information Technology, Torino, Italy
trevisan@celi.it

1 Motivations

Recent research showed that adopting formal ontologies as a means for accessing heterogeneous data sources has many benefits, in that not only does it provide a uniform and flexible approach to integrating and describing such sources, but it can also support the final user in querying them, thus improving the usability of the integrated system.

We introduce a framework that enables access to heterogeneous data sources by means of a conceptual schema and supports the users in the task of formulating a precise query over it. In describing a specific domain, the ontology defines a vocabulary which is often richer than the logical schema of the underlying data and usually closer to the user's own vocabulary. The ontology can thus be effectively exploited by the user in order to formulate a query that best captures their information need. The user is constantly guided and assisted in this task by an intuitive visual interface, whose intelligence is dynamically driven by reasoning over the ontology. The inferences drawn on the conceptual schema help the user in choosing what is more appropriate with respect to their information need, restricting the possible choices to only those parts of the ontology which are relevant and meaningful in a given context.

The most powerful and innovative feature of our framework lies in the fact that not only do not users need to be aware of the underlying organisation of the data, but they are also not required to have any specific knowledge of the vocabulary used in the ontology. In fact, such knowledge can be gradually acquired by using the tool itself, gaining confidence with both the vocabulary and the ontology. Users may also decide to just explore the ontology without actually querying the information system, with the aim of discovering general information about the modelled domain.

Another important aspect is that only queries that are logically consistent with the context and the constraints imposed by the ontology can be formulated, since contradictory or redundant pieces of information are not presented to the user at all. This makes user's choices clearer and simpler, by ruling out irrelevant information that might be distracting and even generate confusion. Furthermore, it also eliminates the often frustrating and time-consuming process of finding the right combination of parts that together constitute a meaningful query. For this reason, the user is free to explore the ontology without the worry of making a "wrong" choice at some point and can concentrate on expressing their information need.

Queries can be specified through a refinement process consisting in the iteration of few basic operations: the user first specifies an initial request starting with generic terms, then refines or deletes some of the previously added terms or introduces new ones, and iterates the process until the resulting query satisfies their information need. The available operations on the current query include addition, substitution and deletion of pieces of information, and all of them are supported by the reasoning services running over the ontology.

In this paper we summarise only the NL aspects of a tool based on those ideas, **Quelo**; for a complete picture of our ideas and of the tool refer to our papers (Franconi et al., 2011; Dongilli et al., 2004; Catarci et al., 2004; Catarci et al., 2005; Dongilli and Franconi, 2006; Franconi et al., 2010). Quelo relies on a web-based client-server architecture:

1. the tool logic, responsible of "reasoning" over the ontology in order to provide only relevant information w.r.t. the current query;

WS 2 Workshop Extended Abstracts, 9th International Conference on Terminology and Artificial Intelligence, TIA 2011, pages 43–46
Paris, 10 November 2011

2. the natural language generation (NLG) engine, that given a query and a lexicalisation map for the ontology produces an English sentence; the lexicon is automatically generated from the ontology;

3. the user interface (GUI), that provides visual access to the query and editing facilities for it, allowing to interact with the reasoning sub-system while benefiting from the services of the NLG engine.

An online fully functional demonstrator of Quelo is freely accessible at:
`http://krdbapp.inf.unibz.it:8080/quelo/`

2 Natural language aspects

The natural language interface of the tool masks the composition of a precise query as the composition of English text describing the equivalent information need. Interfaces following this paradigm are known as "menu-based natural language interfaces to databases" or "conceptual authoring" (see, most notably, (Hallett et al., 2007)). As we have seen before, the users of such systems edit a query by composing fragments of generated natural language provided by the system through *contextual* menus. In (Franconi et al., 2010) we describe how the natural language rendering of a query is achieved.

We start by defining a particular linear form of the query that satisfies certain constraints, necessary to represent the elements of the query using a linear medium, that is, text. The constraints are enforced at the API level to ensure that different graphical user interfaces represent the query in a homologous way. Moreover, a consistent ordering of the query elements needs to be preserved during the operations for query manipulation to avoid confusing the end user. The linearised version of the query is then used as a guide for the language generation performed by the tool's NLG engine.

The natural language interface (NLI) of the tool relies on a natural language generation (NLG) system to produce the textual representation of the query, following an idea presented in (Tennant et al., 1983) and refined in (Hallett et al., 2007).

For the tool's NLI to work with a specific knowledge base (KB) a lexicon and a template map must be provided for it. To ease the burden of developing these resources from scratch, we let the system generate them automatically. The functionality we implemented allows to produce all the resources necessary to configure our NLI for use with a new KB, using as a source of data the ontology itself.

2.1 Natural Language Generation module

NLG systems use techniques from artificial intelligence and computational linguistics to produce human-readable texts out of machine-readable data. The Query Tool uses NLG to represent the whole query, along with all the elements that the user can use to refine it, as English text. The generated text is enriched with links that connect it to the underlying logical form of the query. This allows the user to operate on the query simply by editing an English text.

Unlike most NLG systems, ours is built to let the user determine the structure of the generated text by inserting, replacing and removing snippets of it. While in the classic NLG pipeline the information to be conveyed in the text and its order is determined by the document planning module, in the Query Tool it is the user who decides both the information to be displayed and its arrangement.

As the Query Tool is not tailored to any specific domain, its NLG module is simple enough to be adopted in any context and it is not bundled with all the resources that are needed to generate text out-of-the-box. Therefore, in order to use it on a specific knowledge base, the system must be provided with a *lexicon* and a *template map*. The former contains the words to be used in the generated text; the latter is the bridge between the natural language and the knowledge representation language, associating each concept/role name with a generation template. Each such template contains the syntactic and lexical information necessary to generate a fragment of text representing the associated concept or role.

We selected the syntactic features available in the templates, hence supported by the generator, in order to keep the system simple while still being expressive enough. For this purpose, we collected and analysed a corpus of more than 12.000 unique relation identifiers and we partitioned them according to the recurring syntactic patterns. For each class of the partition, we then proposed a common natural language representation template. The result of this study is a set of simple

but effective templates for representing most ontology relations using natural language.

During the first stage (*microplanning*) of the generation, linguistic information stored in the template map and in the lexicon blends with the logic information encoded in the query into a single structure, known in the NLG literature as *text specification* and consisting of a list of syntactic trees with inflected lexemes on its leaves. The NLG system operates on this structure to aggregate groups of adjacent syntactic structures into single more complex structures, and to select and replace existing referring expressions with more appropriate ones. These two tasks are known in the literature as *aggregation* and *referring expressions generation*, respectively. At the same time, the system keeps track of which element of the text specification is associated with which element (either a node tag or an edge tag) of the query. An association holds when the syntactic element is the result of the instantiation of a template associated with the element of the query. These associations are used for enriching the generated text with links to the underlying query.

The linearisation of the query simplifies the effort required by the referring expressions generation, as referring expressions that need to be reworked always appear in subject position. Our algorithm replaces a subject with a pronoun whenever the previous sentence had the same subject, otherwise the subject is left unchanged. Although ambiguous expressions may occur, ambiguity is not a crucial issue as these expressions originate from user operations upon a selected element, which always becomes the target of the referring expression. Our aggregation module performs simple aggregation tasks such as aggregating sentences with the same subject, eliding the subject and parts of the verb if it is feasible.

Once these operations are completed, the text specification is ready to be transformed into the final text. This task, known as *surface realisation*, produces a list of text tokens, some of which are connected to edge or node labels. This list is finally fed to the GUI, that displays it to the user.

Elements populating the menu for addition and substitution operations undergo a similar processing. To produce the textual representation of such an element, the system makes a temporary copy of the portion of query affected by the operation. The operation is carried out on this portion and the resulting structure is fed to the generation pipeline used for entire queries. The outcome of this process is the text which will appear on the menu.

2.2 Generation of lexicon and template map

For the tool's NLI to work with a specific knowledge base (KB) a lexicon and a template map must be provided for it. Devising these resources requires an understanding of both the domain of interest and basic linguistic notions such as verb tenses, noun genders and countability. We briefly describe here how the system can generate these resources automatically. This technique follows an approach to domain independent generation proposed in (Sun and Mellish, 2006), after the learning of a rich corpus of relations. The functionality we implemented allows to produce all the resources necessary to configure our NLI for use with a new KB, using as a source of data the ontology itself. It has to be noted that the process is not completely reliable, therefore system engineers must review the result and make the necessary corrections.

The idea is based on the observation that KBs already contain some form of linguistic information. In real-world ontologies, every concept and relation has a unique identifier (ID), which most of the times is not just an arbitrary string, but a mnemonic chosen by the knowledge engineer to describe the intended meaning of the identified concept or relation. Moreover, within these IDs, certain syntactic patterns occur more frequently than others.

In our approach, each relation ID is first tokenized according to an algorithm that takes advantage of the naming conventions used by ontology engineers. Second, the tokenized ID is fed to a custom part-of-speech tagger built around QTAG (Tufis and Mason, 1998). The resulting tagged tokenized ID is then lightly preprocessed before being finally passed to a transformation rule, chosen among thirteen different ones, that produces a template for the template map of the NLG system.

For the design of the transformation rules, we analysed our corpus, containing more than 12.000 relation IDs, in order to devise a partition of the domain in terms of syntactic patterns. The classes defined in this partition are s.t. to each relation of the same class can be applied a simple transfor-

mation in order to obtain a template. Each such transformation is also a uniform interpretation of the intended meaning of each relation ID in the class. Some care is needed when giving a uniform interpretation to syntactic patterns, as there are situations in which the same syntactic pattern is to be interpreted differently. For instance, the relation IDs "country_of_nationality" and "language_of_country" share the same syntactic structure, but the first relation should be read as "the country of nationality of X is Y", while the second as "the language of X is Y". Each of the thirteen rules we defined corresponds to one class of the partition, and together they can handle 93% of the relations of the average ontology.

The system has been formally evaluated with some ontologies (e.g., (Ordnance Survey, ; Drummond et al.,)), contributing 64 unique relations in total. It is now available online and it has been used in many different contexts. From the IDs of these relations we automatically generated relation templates, which were then inspected in order to evaluate their usability in text generation. The result of the evaluation revealed that for 42 out of 64 relations (65%) the generated template is suitable for direct use with the Query Tool's NLI. The result suggests that although the generation of the template map is not totally reliable, it is nevertheless useful in that it speeds up the work of systems engineers, as they do not need to create the whole map from scratch, but only have to review the generated map and repair eventual errors. This improves the portability of the Query Tool's NLI, making it faster and easier to switch to a different knowledge base.

References

Tiziana Catarci, Paolo Dongilli, Tania Di Mascio, Enrico Franconi, Giuseppe Santucci, and Sergio Tessaris. 2004. An ontology based visual tool for query formulation support. In *Proc. of the 16th Eur. Conf. on Artificial Intelligence (ECAI 2004)*.

Tiziana Catarci, Paolo Dongilli, Tania Di Mascio, Enrico Franconi, Giuseppe Santucci, and Sergio Tessaris. 2005. Usability evaluation tests in the SeWAsIE (SEmantic Webs and AgentS in Integrated Economies) project. In *Proceedings of the 11th International Conference on Human-Computer Interaction (HCII 2005)*.

Paolo Dongilli and Enrico Franconi. 2006. An Intelligent Query Interface with Natural Language Support. In *Proc. of the 19th Int. Florida Artificial Intelligence Research Society Conference (FLAIRS 2006)*, Melbourne Beach, Florida, USA, May.

Paolo Dongilli, Enrico Franconi, and Sergio Tessaris. 2004. Semantics driven support for query formulation. In *Proc. of the 2004 Description Logic Workshop (DL 2004)*.

Nick Drummond, Matthew Horridge, Robert Stevens, Chris Wroe, and Sandra Sampaio. Pizza ontology. The University of Manchester.

Enrico Franconi, Paolo Guagliardo, and Marco Trevisan. 2010. An intelligent query interface based on ontology navigation. In *Proc. of the Workshop on Visual Interfaces to the Social and Semantic Web (VISSW 2010)*, February.

Enrico Franconi, Paolo Guagliardo, Marco Trevisan, and Sergio Tessaris. 2011. Quelo: an ontology-driven query interface. In *Proceedings of the 24th International Workshop on Description Logics (DL 2011)*.

Paolo Guagliardo. 2009. Theoretical foundations of an ontology-based visual tool for query formulation support. Technical Report KRDB09-5, KRDB Research Centre, Free University of Bozen-Bolzano. `http://www.inf.unibz.it/krdb/pub/TR/KRDB09-05.pdf`, October.

Catalina Hallett, Donia Scott, and Richard Power. 2007. Composing questions through conceptual authoring. *Computational Linguistics*, 33(1):105–133.

Ordnance Survey. Great britain's national mapping agency. `http://www.ordnancesurvey.co.uk/oswebsite/ontology/`.

Xiantang Sun and Chris Mellish. 2006. Domain independent sentence generation from RDF representations for the Semantic Web. In *Proc. ECAI'06 Combined Workshop on Language-Enhanced Educational Technology and Development and Evaluation of Robust Spoken Dialogue Systems*.

Harry R. Tennant, Kenneth M. Ross, Richard M. Saenz, Craig W. Thompson, and James R. Miller. 1983. Menu-based natural language understanding. In *Proc. 21st Annual Meeting of the Association for Computational Linguistics*, pages 151–158. Association for Computational Linguistics.

Marco Trevisan. 2009. A portable menu-guided natural language interface to knowledge bases. Master's thesis, University of Groningen.

Dan Tufis and Oliver Mason. 1998. Tagging Romanian texts: a case study for QTAG, a language independent probabilistic tagger. *Proc. 1st Int. Conf. on Language Resources and Evaluation (LREC'98)*, pages 589–596.