AquaLog: An ontology-driven Question Answering System to interface the Semantic Web

Vanessa Lopez Garcia
Knowledge Media Institute
The Open University.
Walton Hall, Milton Keynes,
MK7 6AA United Kingdom.
v.lopez@open.ac.uk

Enrico Motta
Knowledge Media Institute
The Open University.
Walton Hall, Milton Keynes,
MK7 6AA United Kingdom.
e.motta@open.ac.uk

Victoria Uren
Knowledge Media Institute
The Open University.
Walton Hall, Milton Keynes,
MK7 6AA United Kingdom.
v.s.uren@open.ac.uk

Abstract

The *semantic web* (SW) vision is one in which rich, *ontology-based semantic markup* will become widely available. The availability of semantic markup on the web opens the way to novel, sophisticated forms of question answering. AquaLog is a portable question-answering system which takes queries expressed in natural language (NL) and an ontology as input, and returns answers drawn from one or more knowledge bases (KB). AquaLog presents an elegant solution in which different strategies are combined together in a novel way. AquaLog novel ontology-based *relation similarity service* makes sense of user queries.

1 Introduction

AquaLog (Lopez, 2005) is a fully implemented ontology-driven Question Answering (QA) system[1], which takes an ontology and a NL query as an input and returns answers drawn from semantic markup (viewed as a KB) compliant with the input ontology. In contrast with much existing work on ontology-driven QA, which tends to focus on the use of ontologies to support query expansion in information retrieval (Mc Guinness, 2004), AquaLog exploits the availability of semantic statements to provide precise answers to complex queries expressed in NL.

AquaLog makes use of the GATE NLP platform, string distance metric, generic lexical resources, such as WordNet, as well as the structure of the input ontology, to make sense of the terms and relations expressed in the input query with respect to the target KB. Naturally, these terms and relations match the terminology familiar to the user rather than those used in the ontology.

We say that AquaLog is portable because the configuration time required to customize the system for a particular ontology is negligible. We believe that in the SW scenario it makes sense to provide a NL query interface *portable with respect to ontologies*, our AquaLog system allows to choose an ontology and then ask queries with respect to its universe of discourse. The reason for this is that the architecture of the system and the reasoning methods are domain-independent, relying on an understanding of general-purpose knowledge representation languages, such as OWL[2], and the use of generic lexical resources, such as WordNet.

Moreover, AquaLog learning mechanism ensures that, for a given ontology and a particular community jargon used by end users, its performance improves over time, as the users can easily correct mistakes and allow AquaLog to learn novel associations between the NL relations used by users and the ontology structure.

Approach. AquaLog uses a sequential process model (see Fig. 1), in which NL input is first translated into a set of intermediate representations – called Q*uery Triples*, by the Linguistic Component. The Linguistic Component uses the GATE infrastructure and resources (Cunningham, 2002) to obtain a set of syntactic annotations associated with the input query and to classify the query. Once this is done, it becomes straight-forward for the Linguistic Component to automatically create the Query-Triples. Then, these query triples are

[1] AquaLog is available as Open Source https://sourceforge.net/projects/aqualog

[2] A plug-in mechanism and a generic API ensure that different Knowledge Representation languages can be used.

Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume, pages 269–272,
New York City, June 2006.

further processed and interpreted by the Relation Similarity Service Component (RSS), which uses lexical resources and the ontology to map them to ontology-compliant semantic markup or triples.

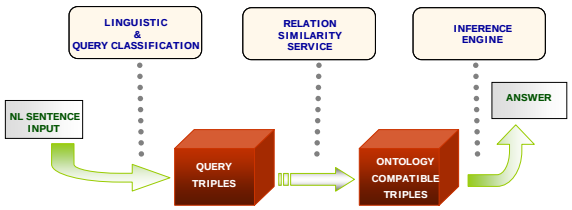

Fig. 1. AquaLog data model

2 Linguistic Component

The Linguistic Component's task is to map the NL input query to the Query-Triple. GATE (Cunningham, 2002) infrastructure and resources (e.g. processing resources like ANNIE) are part of the Linguistic Component.

After the execution of the GATE controller, a set of syntactic annotations associated with the input query are returned. These annotations include information about sentences, tokens, nouns and verbs. For example, we get voice and tense for the verbs and categories for the nouns, such as determinant, singular/plural, conjunction, possessive, determiner, preposition, existential, wh-determiner, etc. When developing AquaLog we extended the set of annotations returned by GATE, by identifying terms, relations, question indicators (which/who/when. etc.) and patterns or types of questions. This is achieved through the use of *Jape grammars*, which consist of a set of *phases*, that run sequentially, and each phase is defined as a set of pattern rules, which allow us to recognize regular expressions using previous annotations in documents.

Thanks to this architecture that takes advantage of the *Jape grammars*, although we can still only deal with a subset of NL, it is possible to extend this subset in a relatively easy way by updating the regular expressions in the *Jape grammars*. This ensures the easy portability of the system with respect to both ontologies and natural languages. Currently, the linguistic component, through the *Jape grammars*, dynamically identifies around 14 different linguistic categories or intermediate representations, including: basic queries requiring an affirmation/negation or a description as an answer; or the big set of queries constituted by a wh-question (such as the ones starting with: what, who, when, where, are there any, does anybody/anyone or how many, and imperative commands like list, give, tell, name, etc.), like "are there any PhD students in dotkom?" where the relation is implicit or unknown or "which is the job title of John?" where no information about the type of the expected answer is provided; etc.

Categories tell us not only the kind of solution that needs to be achieved, but also they give an indication of the most likely common problems that the system will need to deal with to understand this particular NL query and in consequence it guides the process of creating the equivalent intermediate representation. Categories are the driving force to generate an answer by combining the triples in an appropriate way. For example, in "who are the academics involved in the semantic web?" the triple will be of the form <generic term, relation, second term>, i.e. <academics, involved, semantic web>. A query with a equivalent triple representation is "which technologies has KMi produced?", where the triple will be <technologies, has produced, KMi>. However, a query like "are there any PhD students in akt?" has another equivalent representation, where the relation is implicit or unknown <phd students, ?, akt> . Other queries may provide little information about the type of the expected answer, i.e. "what is the job title of John?", or they can be just a generic enquiry about someone or something, i.e. "who is Vanessa?", "what is an ontology?"

At this stage we do not have to worry about getting the representation completely right as the interpretation is completely domain independent. The role of the triple-based intermediate representation is simply to provide an easy way to represent the NL query and to manipulate the input for the RSS. Consider the request "List all the projects in the knowledge media institute about the semantic web", where both "in knowledge media institute" and "about semantic web" are modifiers (i.e. they modify the meaning of other syntactic constituents). The problem here is to identify the constituent to which each modifier has to be attached. The RSS is responsible for resolving this ambiguity through the use of the ontology, or by interacting with the user. The linguistic component's task is therefore to pass the ambiguity problem to the RSS through the intermediate representation.

Nevertheless, a query can be a composition of two basic queries. In this case, the intermediate representation usually consists of two triples, one triple per relationship. There are different ways in which queries can be combined. Firstly, queries can be combined by using a "and" or "or" conjunction operator, as in "which projects are funded by epsrc and are about semantic web?". This query will generate two Query-Triples: <projects, *funded*, epsrc> and <projects, *?*, semantic web> and the subsequent answer will be a combination of both lists obtained after resolving each triple. Secondly, a query may be conditioned to a second query, as in "which researchers wrote publications related to social aspects?" which generates the Query-Triples <researchers, *wrote*, publications> and <*which are*, *related*, social aspects>,where the second clause modifies one of the terms in the first triple. In this example, ambiguity cannot be solved by linguistic procedures; therefore the term to be modified by the second clause remains uncertain.

3 Relation Similarity Service

This is the backbone of the QA system. The RSS component is invoked after the NL query has been transformed into a term-relation form and classified into the appropriate category. Essentially the RSS tries to make sense of the input query by looking at the structure of the ontology, string metrics[3], WordNet, and a domain-dependent lexicon obtained by the Learning Mechanism.

In any non-trivial NL system, it is important to deal with the various sources of ambiguity. Some sentences are structurally ambiguous and although general world knowledge does not resolve this ambiguity, within a specific domain it may happen that only one of the interpretations is possible. The key issue here is to determine some constraints derived from the domain knowledge and to apply them in order to resolve ambiguity. Whether the ambiguity cannot be resolved by domain knowledge the only reasonable course of action is to get the user to choose between the alternative readings. Moreover, since every item on the onto-triple is an entry point in the KB or ontology the user has the possibility to navigate through them. In fact, to ensure user acceptance of the system justifications are provided for every step of the user interaction.

[3] http://secondstring.sourceforge.net/

4 Related Work

This scenario is similar to research in NL queries to databases (NLIDB). However, the SW provides a new and potentially important context in which results from this research area can be applied. There are linguistic problems common in most kinds of NL understanding systems, see (Androutsopoulos, 1995) for an overview of the state of the art. In contrast with the latest generation of NLIDB systems (see (Popescu, 2003) for recent work) AquaLog uses an intermediate representation from the representation of the user's query (NL front end) to the representation of an ontology compliant triple, from which an answer can be directly inferred. It takes advantage of the use of ontologies in a way that the entire process highly portable and it is easy to handle unknown vocabulary. For instance, in PRECISE (Popescu, 2003) the problem of finding a mapping from the tokenization to the database requires that all tokens must be distinct, questions with unknown words are not semantically tractable and cannot be handled. In contrast with PRECISE, AquaLog interpret the user query by means of the ontology vocabulary and structure in order to make sense of unknown vocabulary which appears not to have any match.

Current work on QA is somewhat different in nature from AquaLog as they are open-domain systems. QA applications to text typically involve (Hirschman, 2001) identifying the semantic type of the entity sought by the question (a date, a person…); and determining key words or relations to be use in matching candidate answers. Moreover, as pointed by Srihari et al. (Srihari, 2004) Named Entity (NE) tagging is often necessary. The main differences between AquaLog and open-domains systems are: (1) it is not necessary to build hierarchies or heuristic to recognize NE, as all the semantic information needed is in the ontology. (2) AquaLog has mechanisms to exploit the relationships to understand a query. Nevertheless, the RSS goal is to map the relationships in the Query-Triple into an ontology-compliant-triple. Both AquaLog and open-domain systems attempt to find synonyms plus their morphological variants. AquaLog also automatically classifies the question before hand, based on the kind of triple needed, while most of the open-domain QA systems classify questions according to their answer target. The triple contains information not only about the an-

swer expected, but also about the relationships of the other terms in the query. To conclude, other QA systems also follow a relational data model (triple-based), e.g. the START "object-property-value" approach (Katz, 2002).

5 AquaLog in action: illustrative example.

For demonstration purposes AquaLog application is used with the AKT ontology in the context of the academic domain in our department (Lei, 2006), e.g., AquaLog translates the query "what is the homepage of Peter who has an interest on the semantic web?" into a conjunction of ontology-compliant non-ground triples: <what is?, has-web-address, peter-scott> & <person?, has-research-interest, Semantic Web area>.

Consider the query "what is the homepage of Peter?" on Fig. 2. Given that the system is unable to disambiguate between Peter-Scott, Peter-Sharpe, etc, user feedback is required. Also the user is call to disambiguate that "homepage" is the same that "has-web-address" as it is the first time the system came across this term, no synonyms have been identified, and the ontology does not provide further ways to disambiguate. The system will learn the mapping and context for future occasions.

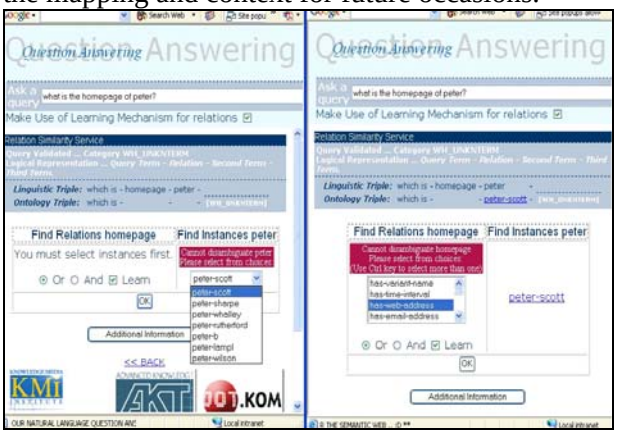

Fig. 2. Example of user disambiguation

On Fig. 3 we are asking for the web address of Peter, who has an interest in SW. In this case AquaLog does not need any assistance from the user, given that only one of the Peters has an interest in SW. Also the similarity relation between "homepage" and "has-web-address" has been learned by the Learning Mechanism. When the RSS comes across a query like that it has to access to the ontology information to recreate the context and complete the ontology triples. In that way, it realizes that "who has an interest on the Semantic Web" is a modifier of the term "Peter".

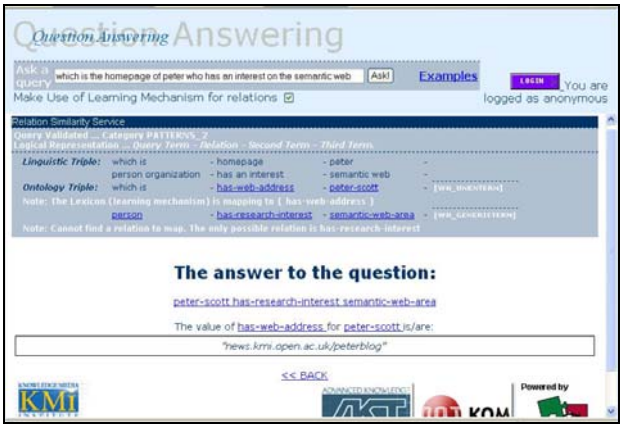

Fig. 3. Example of AquaLog disambiguation

References

Androutsopoulos, I., Ritchie, G.D., Thanisch P.: Natural Language Interfaces to Databases - An Introduction. *Natural Language Engineering*, 1(1) (1995) 29-81.

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *In Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (2002).*

Hirschman, L., Gaizauskas, R.: Natural Language question answering: the view from here. *Natural Language Engineering, Special Issue on Question Answering*, 7(4) (2001) 275-300.

Katz, B., et al.: Omnibase: Uniform Access to Heterogeneous Data for QA. In Proceedings of the 7th International Workshop on Applications of Natural Language to Information Systems (NLDB) (2002)

Lopez, V., Pasin, M., and Motta, E. AquaLog: An Ontology-portable Question Answering System for the Semantic Web. In proceeding of the ESWC (2005).

Mc Guinness, D.: Question Answering on the Semantic Web. IEEE Intelligent Systems, 19(1), 2004.

Popescu, A., M., Etzioni, O., Kautz, H., A.: Towards a theory of natural language interfaces to databases. *In Proceedings of the 2003 International Conference on Intelligent User Interfaces*, (2003) 149-157

Srihari, K., Li, W., Li, X.: Information Extraction Supported QA, In T. Strzalkowski & S. Harabagiu (Eds.), in *Advances in Open- Domain Question Answering*. Kluwer Academic Publishers (2004)

Lei, Y., Sabou, M., Lopez, V., Zhu, J., Uren, V. and Motta. E. An Infrastructure for Acquiring high Quality Semantic Metadata. *In proceedings of the 3rd European Semantic Web Conference.* Montenegro, (2006).