A Text-based Query Interface to OWL Ontologies

Danica Damljanovic, Valentin Tablan, Kalina Bontcheva
danica@dcs.shef.ac.uk

Objective

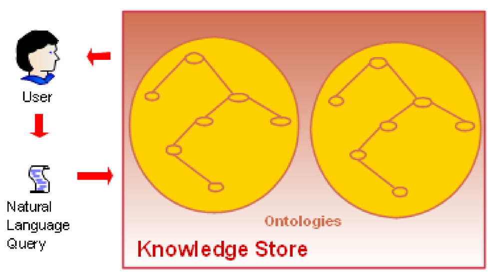

Figure 1: Objective

Developing a tool for querying the knowledge store using Natural Language (NL) queries; knowledge store includes ontologies and the knowledge base created based on them.

Motivation

Existing query languages (e.g., SeRQL, SPARQL) are complex. Writing queries:

- requires using the exact syntax that is not easy to learn,
- is error-prone task and
- requires understanding of Semantic Web technologies.

QuestIO: Question-based Interface to Ontologies

As shown in Figure 2, QuestIO is transforming text-based Natural Language queries into SeRQL queries. Main features of this system are:

- Portable without prior customisation.
- Easy to use: no training for the user.
- Allowing users to enter queries of any length and form.

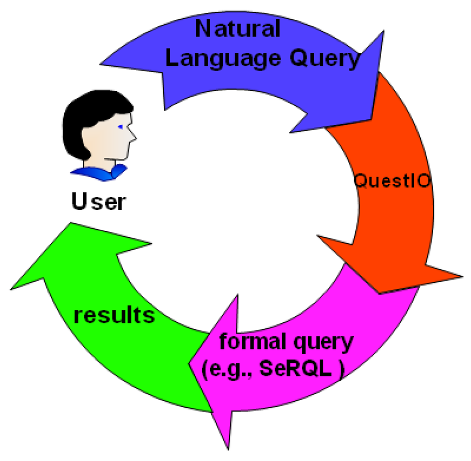

Figure 2: Querying the knowledge store

Extracting domain knowledge automatically

To initialise our system automatically we preprocess the ontology resources (e.g., classes, instances, properties and property values) and extract any human-understandable lexicalisations. Further on, we process these lexicalisations using *Onto Root Application* as shown in Figure 3, and extract roots i.e. lemmas which are then added to a dynamic gazetteer list. This list is used by *Key Concept Identification Tool (KCIT)* as explained later.

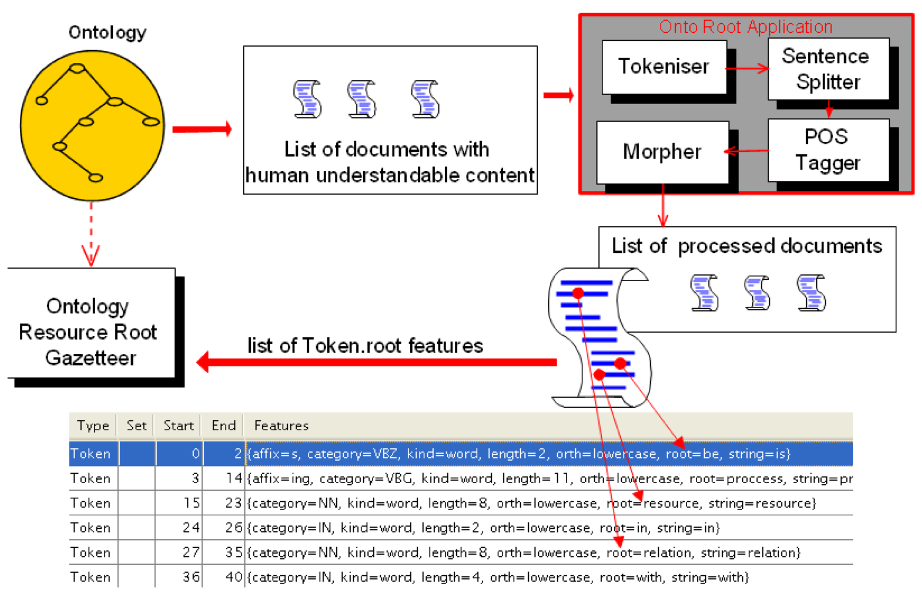

Figure 3: Building a Dynamic Gazetteer from the Ontology

Processing the query

As shown on Figure 4, after the query is being interpreted by *Query Interpreter*, it is being further analysed by two components. *KCIT* first recognises key concepts inside the query – those that are referring to the ontology resources (classes, instances, properties, property values). Next, *Context Collector* collects any other concepts that might be relevant for the further processing of the query, such as keywords (e.g., how many, count) or any chunks between recognised key concepts. Next, *Query Analyser* uses all collected concepts to perform appropriate transformations, formulate SeRQL queries, execute them and send them back to the *User Interface* where the *Result Formatter* renders them in a user-friendly manner.

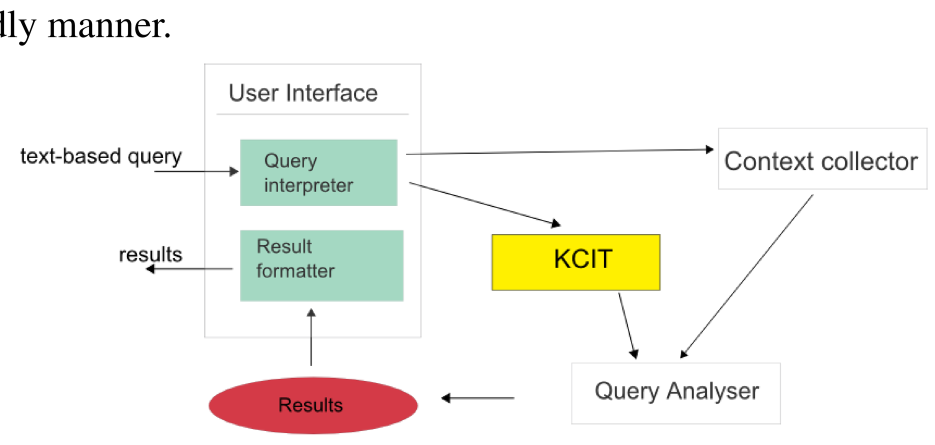

Figure 4: Component diagram

The detailed view of how *Query Analyser* uses collected concepts to create and execute SeRQL queries is shown in Figure 5.

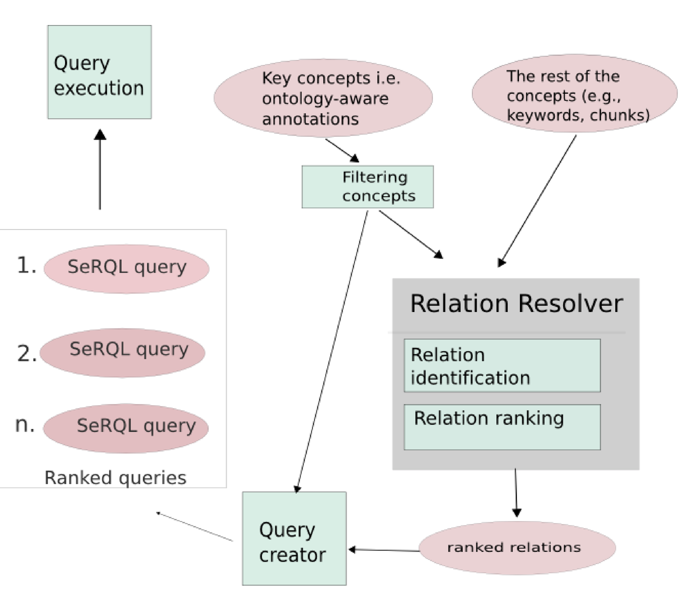

Figure 5: Query Analyser: detailed view

Demo

Demo available at: `http://www.gate.ac.uk/projects/tao/webpage/demos/movies/questio/questio.html`.

Acknowledgements

This research is partially supported by the EU Sixth Framework Program project TAO (FP6-026460): www.tao-project.eu.

Examples

Figures 6 and 7 depict example queries and how QuestIO interprets them to formulate SeRQL queries.

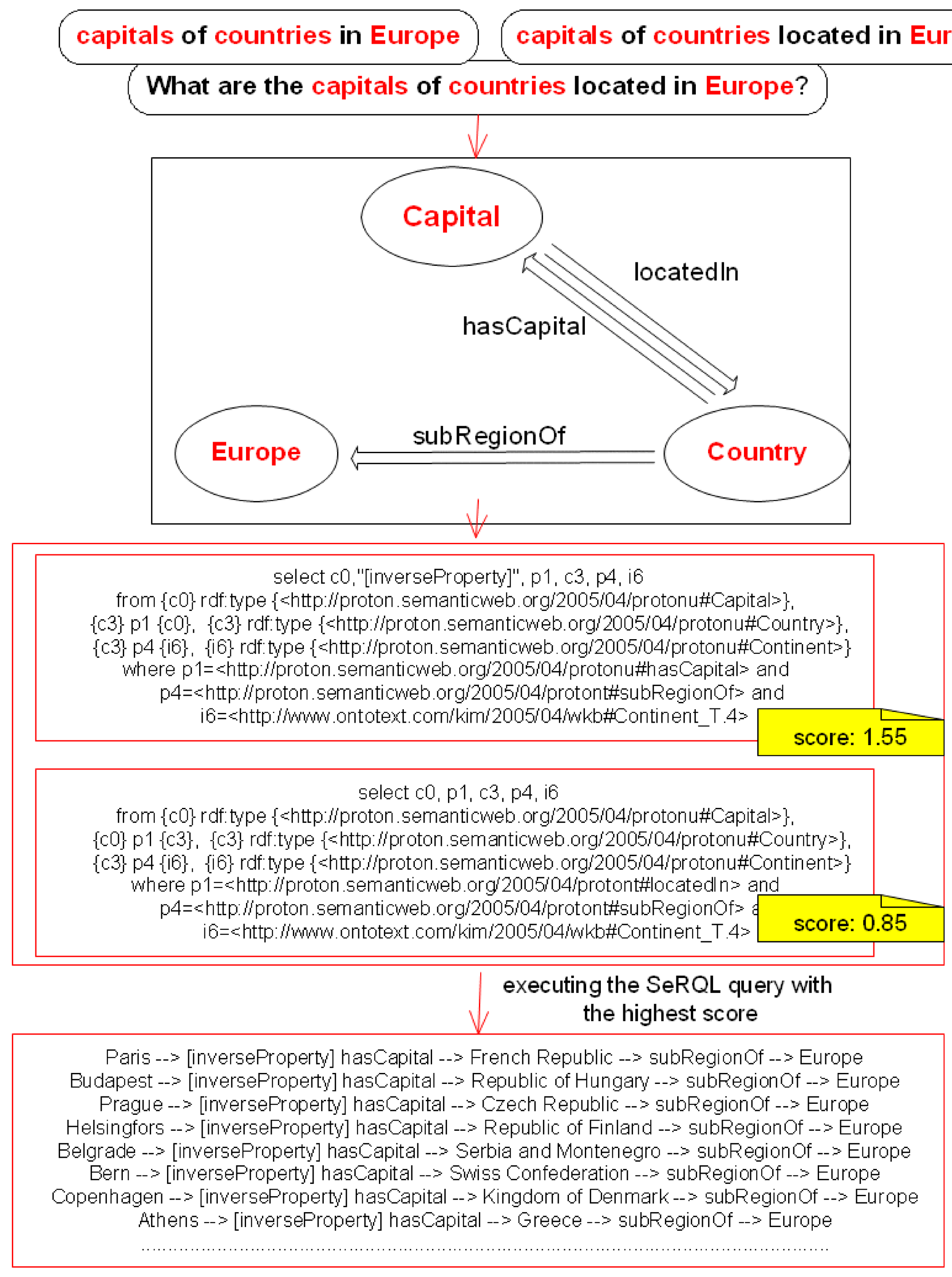

Figure 6: Handling queries with QuestIO

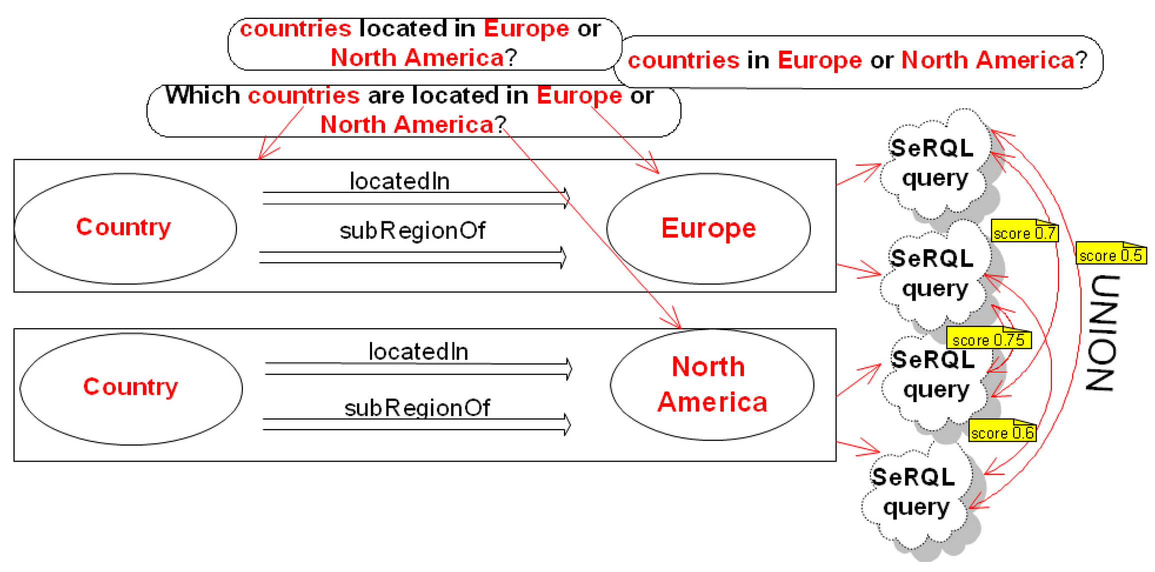

Figure 7: Handling disjunction queries with QuestIO

Evaluation

Coverage and correctness: 36 questions were collected from the GATE user mailing list where users are enquiring about various GATE modules and plugins. These questions were run against the GATE knowledge base: http:/gate.ac.uk/gate-kb, developed in the TAO project (http://www.tao-project.eu).

22 out of 36 questions were *answerable* (the answer was in the knowledge base). Further on, we eliminated 6 questions that were not supported by AquaLog system. We trialed both QuestIO and AquaLog with the rest of 16 questions. Results are shown in Table 1.

Table 1: Results of running the same set of queries with QuestIO and Aqualog: *c. correct - conditionally correct (correct after reformulated), p. correct - partially correct*

	QuestIO	*Correct*	Aqualog	*Correct*
correct	9 (56.25%)	56.25%	5 (31.25%)	31.25%
c. correct	0	(0%)	3 (18.75%)	18.75%
p. correct	5 (31.25%)	15.63%	3 (18.75%)	9.35%
failed	2 (12.5%)	0%	5 (31.25%)	0%
		71.88%		59.35%

Portability and Scalability: In this evaluation we show how our system scales when used with knowledge bases of different sizes, one being a subset of the other:

- KIM KB contains general data, specifically about organisations, persons, locations.
- Travel Guides Knowledge Base (KB) contains instances and relations between them from the Travel Guides (TG) Ontology: `http://goodoldai.org.yu/ns/tgproton.owl`. The core for Travel Guides KB contains geographical data such as those about cities, countries and continents . This core was exploited from KIM KB and has about 40 times less resources.

We prepared a set of queries enquiring about geographical locations and due to the overlap between the two knowledge bases, they give the same results for the same set of queries. Table 2 shows the the performance of the system in both cases.

Table 2: Initialization times for two knowledge bases of different sizes and execution times for running the same set of queries with QuestIO. Shown times are in seconds.

	TG	KIM
Initialization time	22.3	228
Queries	**Execution time**	
countries located in Asia	0.547	5.65
capitals of countries located in Asia	0.203	5.6
capitals of countries in southern Europe	0.109	5.4
which are the political regions in Europe	0.141	11.1
is London capital of any country?	0.625	111
capital country France	0.344	10.6