WP6 review presentation

- GATE ontology
- QuestIO - Question-based Interface to Ontologies

Funded by: European Commission – 6th Framework
Project Reference: IST-2004-026460
GATE Ontology – New/Changed Concepts

- Plugin – describes GATE plugins, which are sets of Resources
  - Key property: containsResource
- JavaClass – refers to the Java classes implementing the components
  - javaFullyQualifiedName
- Resources – new properties
  - Has<Init/Run>TimeParameter
  - resourceHasName, resourceHasComment
- ResourceParameter
  - parameterHasName, parameterHasDefaultValue
GATE knowledge base comprises:
- 42 classes
- 23 object properties
- 594 instances
Resource Instance Example
ANNIE Plugin Instance
Automatic Ontology Population from XML Config Files
Wrap-up

- New version of GATE ontology now distributed
- Most classes and properties same as before
- Some small changes detailed above, needed to model the data from the plugins configuration files
- Once mapping established from XML elements to ontology classes and properties, conversion was straightforward => ontology populated automatically
QuestIO: a Question-based Interface to Ontologies

Danica Damljanović
Valentin Tablan
Kalina Boncheva
University of Sheffield
d.damljanovic@dcs.shef.ac.uk

Funded by: European Commission – 6th Framework
Project Reference: IST-2004-026460
Content

- Objective and Motivation
- Problems and challenges
- Our Approach (how we do it?)
- Achievements (what we have done?)
- Evaluation
- What next?
- Questions?
Objective

- Developing a tool for querying the knowledge store using text-based Natural Language (NL) queries.
Motivation

Downsides of existing query languages (e.g., SeRQL, SPARQL):
- complex syntax,
- not easy to learn,
- writing queries is error-prone task,
- requires understanding of Semantic Web technologies.
Does it make sense?

“Java Class for parameters for processing resources in ANNIC?”

```
select c0,"[inverseProperty]", p1, c2,"[inverseProperty]", p3, c4,"[inverseProperty]", p5, i6
from {c0} rdf:type {<http://gate.ac.uk/ns/gate-ontology#JavaClass>}, {c2} p1 {c0}, {c2} rdf:type {<http://gate.ac.uk/ns/gate-ontology#ResourceParameter>}, {c4} p3 {c2}, {c4} rdf:type {<http://gate.ac.uk/ns/gate-ontology#ProcessingResource>}, {i6} p5 {c4}, {i6} rdf:type {<http://gate.ac.uk/ns/gate-ontology#GATEPlugin>}
where p1=http://gate.ac.uk/ns/gate-ontology#parameterHasType and p3=http://gate.ac.uk/ns/gate-ontology#hasRunTimeParameter and p5=http://gate.ac.uk/ns/gate-ontology#containsResource and i6=<http://gate.ac.uk/ns/gate-ontology#annic>
```
One year ago…

- A **Controlled** Language for Ontology Querying:
  - recognizing patterns in a text-based query and creating SeRQL queries accordingly;

- Limitations:
  - requires syntactically correct sentences;
  - cannot process concept-based queries such as ‘accommodation Rome’;
  - can process a limited set of queries.
Challenges

- to enhance robustness;
- to accept queries of any length and form;
- to be portable and domain independent.
From questions to answers

The text query is transformed into a SeRQL query using a set of *Transformers*. The input and an output for a *Transformer* is an *Interpretation*:

- **Interpretations** are used as a container for information.
- **Transformer** represents an algorithm for converting a type of interpretation into another.
From questions to answers

- Producing ontology-aware annotations
- Filtering annotations
- Identifying relations between annotated concepts
- Scoring relations
- Creating SeRQL queries and showing results
An Example

Result:

ANNIE Gazetteer --> hasRunTimeParameter --> document
ANNIE Gazetteer --> hasRunTimeParameter --> ANNIEANNIEGazetteerAnnotationSetName
ANNIE Gazetteer --> hasRunTimeParameter --> ANNIEANNIEGazetteerWholeWordsOnly
ANNIE Gazetteer --> hasRunTimeParameter --> longestMatchOnly
Scoring relations

We combine three types of scores:

- **similarity score** - using Levenshtein similarity metrics we compare input string from the user with the relevant ontology resource

- **specificity score** is based on the subproperty relation in the ontology definition.
*distance score is* inferring an implicit specificity of a property based on the level of the classes that are used as its domain and range.
What are the parameters of PR that is included in ANNIE?

```
select y,p,x from {x} rdf:type {http://gate.ac.uk/ns/gate-ontology#ResourceParameter}, {y} rdf:type {http://gate.ac.uk/ns/gate-ontology#ProcessingResource}, {y} <http://gate.ac.uk/ns/gate-ontology#hasInitTimeParameter> {x}, {y} p {x}, {z} <http://gate.ac.uk/ns/gate-ontology#containsResource> {y} where z=http://gate.ac.uk/ns/gate-ontology#ANNIE
```

```
select y,p,x from {x} rdf:type {http://gate.ac.uk/ns/gate-ontology#ResourceParameter}, {y} rdf:type {http://gate.ac.uk/ns/gate-ontology#ProcessingResource}, {y} <http://gate.ac.uk/ns/gate-ontology#hasRunTimeParameter> {x}, {y} p {x}, {z} <http://gate.ac.uk/ns/gate-ontology#containsResource> {y} where z=http://gate.ac.uk/ns/gate-ontology#ANNIE
```
What are the parameters of ANNIE POS Tagger OR Sentence Splitter?
Our achievements

- **Dynamically** generating SeRQL queries.
- **Unlimited number of concepts in a query.**
- Partially supporting **relative clauses:**
  - What are the parameters of the PR that is included in ANNIE plug-in?
- **Grouping identified concepts** to support more complex queries:
  - Which PRs are included in annic AND annie?
  - What are the parameters of POS Tagger OR Sentence Splitter?
- **Setting the environment for implementing user interaction:**
  - Tracking transformations from text to the SeRQL query so that user can be easily returned to the stage where he can change/refine his query.
Evaluation

We evaluated:
- coverage and correctness
- scalability and portability
Evaluation on coverage and correctness

We manually collected 36 questions posted by GATE users to the project’s mailing list in the past year, for example:

- Which PRs take ontologies as a parameter?
- Which plugin is the VP Chunker in?
- What is a processing resource?
Evaluation on coverage and correctness (2)

22 out of 36 questions were answerable (the answer was in the knowledge base):

- 12 correctly answered (54.5%)
- 6 with partially corrected answer (27.3%)
- System failed to create a SeRQL query or created a wrong one for 4 questions (18.2%)

Total score:

- 68% correctly answered
- 32% did not answer at all or did not answer correctly
Evaluation on scalability and portability

Sizes of the knowledge bases created based on:
- GATE ontology: http://gate.ac.uk/ns/gate-ontology
- Travel ontology: http://goodoldai.org.yu/ns/tgproton.owl

<table>
<thead>
<tr>
<th>GATE Knowledge Base</th>
<th>Travel Knowledge Base</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>Classes</td>
</tr>
<tr>
<td>42</td>
<td>318</td>
</tr>
<tr>
<td>Object Properties</td>
<td>Object Properties</td>
</tr>
<tr>
<td>23</td>
<td>86</td>
</tr>
<tr>
<td>Instances</td>
<td>Instances</td>
</tr>
<tr>
<td>594</td>
<td>2790</td>
</tr>
<tr>
<td>Total size (C + P + I)</td>
<td>Total size (C + P + I)</td>
</tr>
<tr>
<td>659</td>
<td>3194</td>
</tr>
<tr>
<td>Initialisation time</td>
<td>Initialisation time</td>
</tr>
<tr>
<td>19 seconds</td>
<td>109 seconds</td>
</tr>
</tbody>
</table>
Evaluation on scalability and portability

Query execution times:

<table>
<thead>
<tr>
<th>Query size</th>
<th>Execution (seconds)</th>
<th>Number of results</th>
<th>Actual query</th>
</tr>
</thead>
<tbody>
<tr>
<td>GATE Knowledge Base</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.148</td>
<td>15</td>
<td>“processing resources in ANNIE?”</td>
</tr>
<tr>
<td>2</td>
<td>0.234</td>
<td>37</td>
<td>“parameters for processing resources in ANNIE?”</td>
</tr>
<tr>
<td>3</td>
<td>0.298</td>
<td>37</td>
<td>“Java Class for parameters for processing resources in ANNIE?”</td>
</tr>
<tr>
<td>Travel Knowledge Base</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.013</td>
<td>52</td>
<td>“countries in asia”</td>
</tr>
<tr>
<td>2</td>
<td>2.030</td>
<td>52</td>
<td>“capitals of countries in asia”</td>
</tr>
<tr>
<td>3</td>
<td>3.307</td>
<td>52</td>
<td>“capitals of countries in global regions in asia”</td>
</tr>
</tbody>
</table>
What next?

- Using implemented transformations to employ user interaction:
  - When the system is not able to make decisions autonomously it will require additional input from the user.
- Improving the algorithms for generating SeRQL queries.
- Optimization of the tool initialization (scalability issues).
- More evaluation on scalability (with KIM).
- Evaluate its expressivity against that of SeRQL.
- Try technologies for soft matching and synonym retrieval, e.g., between hotel and accommodation.
Questions?

Thank you!