Instance Query/Rules Interface

m.c. schraefel (ecs)
Taowei Wang
Oct. 2006
Fast overview

- These slides are a fast walkthrough of a design approach for a rules interface for querying ontology driven spaces
- Exemplar is the PIT or Profiles in Terror project, MindLab, UMD
## Running Use-case

### OBL

<table>
<thead>
<tr>
<th>Given Name</th>
<th>Osama bin Ladin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nickname</td>
<td>Abu Abdullah</td>
</tr>
<tr>
<td>Place of Birth</td>
<td>Riyadh, Saudi Arabia, Riyadh</td>
</tr>
<tr>
<td>Source</td>
<td><a href="http://www.tkb.org/KeyLeader.jsp?memID=6">http://www.tkb.org/KeyLeader.jsp?memID=6</a></td>
</tr>
<tr>
<td>affiliation</td>
<td>al-Qaida</td>
</tr>
<tr>
<td>has contact</td>
<td>Abu Doha -OBL meeting</td>
</tr>
<tr>
<td></td>
<td>Azzam is OBL’s teacher in Jephone battery</td>
</tr>
<tr>
<td>trained in</td>
<td>Osama’s Summer Camp</td>
</tr>
<tr>
<td></td>
<td>Some Mosque</td>
</tr>
<tr>
<td></td>
<td>Al-Qaeda School of Terrorism</td>
</tr>
</tbody>
</table>
What do Rules look like

If...

?x trainedIn ?z.
?y trainedIn ?z.
?x a TerroristLeader.
?y a TerroristLeader.

then...

?x trainedTogether ?y.

Body of the Rule
(Conditions)

Head of the Rule
(Conclusions)

- Can specify arbitrary AND/OR combinations of conditions
- Can have multiple conclusions
- Want to extract away this type of free-form rule-writing
  - Eliminate typing mistakes
- Reduce level of familiarity of ontology required to write rules
- Reduce amount of duplication needed for logic combinations
- Accessible from wherever there is a class-instance/property.
Variables and Variable Typing

-Types can be initially populated with a default value from the point where the user decides to create rules/queries.
  -This shows default “Terrorist Leader”, and up-down the hierarchy
  -OBL is shown because user came to the interface from OBL page.
Specifying And/Or combinations

- Use the orange circle to create boolean combination of class types
- Use Filter to textually filter available type in view
- Multiply select atomic types or combined ones (on right) to create more complex ones
- Select one type (atomic or complex) and press Done to finish
Property Filtering

- Given a variable type (or actual instance), we can filter and reorder the properties presented to the user
- Can also use frequency of usage (require queries to db)
Property Filtering

• Given a variable type (or actual instance), we can filter and reorder the properties presented to the user
• Can also use frequency of usage (require queries to db)
Graph Representation

- Re-enforce user’s cognitive map of the rules
- Immediate reflection to/from widgets
- Disjunctions => multiple graphs
- Once name of the result set is decided (say, “trained Together”), the graph displays “The result ‘trained Together’ is a set of relations between qualifying Terrorist Leaders”.
  - Unary results can say: “the results are a set of Schools”
- Graphical representation of result sets for future reference
Additional features

- Rubber band: put a rubber band around area of graph where you want to keep results of THAT PART of the query persistent

- Combination Of: rather than only Any or ALL can also have “any combination of X out of Y” conditions - upper range of 5 on y so not combinatorially, computationally nuts - for now.

- Plain language reflection of query in process: while the rule is being built up, see plain language feedback of how the system is interpreting the entries to validate the query.

- Query rules data results while building query: a question we have is is it useful to show the results for a rule as it is being built up?
Still more to ponder

• Can we have better representation of variables (than ?x)?
  – A richer annotation language for describing ontologies so interfaces can take advantage of these? (e.g. custom icons)

• How to best represent results? (can we perform analysis (given user’s rules/queries) to highlight ‘important’ portions of the results)