Top-k Retrieval for Automated Human Resource Management

Umberto Straccia, Eufemia Tinelli, Tommaso Di Noia, Eugenio Di Sciascio, Simona Colucci
Outline

- The need for a semantic Human Resource Management
- Trade-off between scalability and expressivity
- Top-k retrieval for DLR-Lite
- A case study for top-k retrieval: HRM
- System description
Who needs a new HRM system?

- HRM is a “knowledge intensive” task
  - a lot of implicit information
- A pure keyword-based search is inadequate
- Information aggregation
Background tools and techniques

- DLR-Lite: a simple, but interesting Description Logic
  - Computationally tractable DL to query large databases
  - Sub-linear in data complexity
  - Good for very large database tables, with limited declarative schema design

- Top-k query answering over a DLR-Lite knowledge base
  - We extend the query formalism: conjunctive queries, where fuzzy predicates may appear
**Facts:** a finite set of expressions in the form $R(c_1, \ldots, c_n)$ where $R$ is a $n$-ary relation and each $c_i$ is a constant.

**Ontology:**

\[
\begin{align*}
Rr & \quad \rightarrow \quad A \mid \exists[i_1, \ldots, i_k] R \\
Rl & \quad \rightarrow \quad A \mid \exists[i_1, \ldots, i_k] R \mid \\
& \quad \quad \quad \exists[i_1, \ldots, i_k] R. (Cond_1 \sqcap \ldots \sqcap Cond_h) \\
Cond & \quad \rightarrow \quad ([i] \leq v) \mid ([i] < v) \mid ([i] \geq v) \mid ([i] > v) \mid \\
& \quad \quad \quad ([i] = v) \mid ([i] \neq v)
\end{align*}
\]

where $A$ is an atomic concept, $R$ is an $n$-ary relation with $1 \leq i_1, i_2, \ldots, i_k \leq n$, $1 \leq i \leq n$ and $v$ is a reference value for the concrete domain interpretation.
Top-k retrieval for DLR-Lite (2/2)

- $\exists[i_1, \ldots, i_k] R$ is the projection of the relation $R$ on the columns $i_1, \ldots, i_k$. Hence, $\exists[i_1, \ldots, i_k] R$ has arity $k$.

- $\exists[i_1, \ldots, i_k] R. (Cond_1 \cap \ldots \cap Cond_l)$ further restricts the projection $\exists[i_1, \ldots, i_k] R$ according to the conditions specified in $Cond_i$.

For instance $([i] \leq v)$ specifies that the value of the $i$-th column have to be less or equal than the value $v$. 
Query Language

\[
q(\vec{x})[s] \leftarrow \exists \vec{y} R_1(\vec{z}_1), \ldots, R_l(\vec{z}_l),
\]

\[
Order By (s = f(p_1(\vec{z}_1'), \ldots, p_h(\vec{z}_h')))
\]

- \(q\) is an \(n\)-ary relation (every \(R_i\) is an \(n_i\)-ary relation) whereas \(\vec{x}\) are the related \(n\) variables (distinguished variables).
- \(\vec{y}\) are the so-called non-distinguished variables.
- \(\vec{z}_i, \vec{z}_j'\) are tuples of constant or variable values in \(\vec{x}\) or \(\vec{y}\).
- \(p_j\) is an \(n_j\)-ary fuzzy predicate assigning to each \(n_j\)-ary tuple \(\vec{c}_j\) a score \(p_j(\vec{c}_j) \in [0, 1]\).
- \(f\) is a scoring function \(f : ([0, 1])^h \rightarrow [0, 1]\), which combines the scores of the \(h\) fuzzy predicates \(p_j(\vec{c}_j')\) into an overall score to be assigned to the rule head \(q(\vec{c})\).
Given a DLR-Lite knowledge base $\mathcal{K}$, and a query

$$q(\vec{x})[s] \leftarrow \exists \vec{y} \phi(\vec{x}, \vec{y})$$

- retrieve $k$ tuples $\langle \vec{c}, s \rangle$ instantiating $q$ with a maximal score
- rank them in decreasing order w.r.t. the score $s$. 

$$\text{ans}_k(\mathcal{K}, q) = \text{Top}_k \text{ ans}(\mathcal{K}, q).$$
The HRM domain
I.M.P.A.K.T.

Entry Points

Ontology browsing

Keyword-based search

Set fuzzy predicates (negotiable)

Set crisp predicates (strict)

The whole query
An example of query (1/2)

- **Strict:**
  - Engineering degree;

- **Preferences:**
  - Engineering degree final mark $\geq 103/110$;
  - Ph.D.;
  - Java programming and experience $\geq 6$ years;
  - Complex problem solving capabilities;
  - Good written English.
An example of query (2/2)

\[ q(id, \text{lastName}) \leftarrow \]

\[ \text{profileLastName}(id, \text{lastName}), \text{hasDegree}(id, \text{degreeId}, \text{mark}), \]
\[ \text{degreeName}(\text{degreeId}, \text{degreeName}), \text{Engineering\_Degree}(\text{degreeId}), \]
\[ \text{hasLevel}(id, \text{levelId}, \text{levelmark}), \text{levelName}(\text{levelId}, \text{levelName}), \]
\[ \text{knowsLanguage}(id, \text{lanID}, \text{Reading}, \text{Verbal}, \text{Writing}), \]
\[ \text{languageName}(\text{langID}, \text{langName}), \]
\[ \text{hasKnowledge}(id, \text{classID}, \text{years}, \text{type}, \text{level}), \]
\[ \text{knowledgeName}(\text{classID}, \text{hasKnowledge}), \]
\[ \text{hasComplementarySkill}(id, \text{classID2}), \text{skillName}(\text{classID2}, \text{capabilities}) \]

\[ \text{OrderBy}(s = \text{rs}(\text{mark}; 102, 110) \cdot 0.166 + \]
\[ \text{pref1}(\text{levelName}; \text{Doctoral\_Degree}) \cdot 0.166 + \]
\[ \text{pref1}(\text{langName}; \text{English}) \cdot \]
\[ \text{pref4}(\text{Writing}; \text{NotSpecified}/1.0, \text{Basic}/3.0, \text{Good}/6.0, \text{Excellent}/9.0) \cdot 0.166 + \]
\[ \text{rs}(\text{years}; 5, 10) \cdot \text{pref1}(\text{hasKnowledge}; \text{Java}) \cdot 0.166 + \]
\[ \text{pref1}(\text{capabilities}; \text{Complex\_Problem\_Solving}) \cdot 0.166) \]
Conclusion and Future Work

- Top-k retrieval for DLR-Lite
- An application for HRM

- Webify the whole system
- Design a (more) user friendly GUI
- Develop a general framework
Q & A