Query Refinement with Ontologies and Users

Advanced Methods of Information Retrieval

Elena Demidova
SS 2018
Aims of the session

“Query refinement with ontologies and users”

- **Lecture:**
  - Understand the concepts and methods:
    - Probabilistic incremental query construction
    - Query interpretation space
    - Usage of ontologies in the interactive query construction process
    - Interaction cost
  - **Hands-on:**
    - Get practical experience with:
      - Creation of query hierarchy
      - Computation of query interpretation probability
      - Computation of information gain for query construction options
Database queries: expressiveness vs. usability

Usability

Easy to use

Complicated

Less expressive

More expressive

Expressiveness

Goal: Expressive AND Easy to use

IQ^P, FreeQ, SQAK

Structured queries

language, schema

(SQL, SPARQL, XQuery)

OBE (`75), NLQ (`99)

Keyword search

possibly imprecise results

BANKS, DBXPlorer, Discover (`02)

Keyword search

possibly imprecise results

BANKS, DBXPlorer, Discover (`02)

Elena Demidova: Advanced Methods of Information Retrieval

adapted from: [Tata et. al 2008]
Database queries: expressiveness vs. usability

- **Database queries:**
  - Knowledge of database schema
  - Knowledge of query language syntax

- **Keyword search:**
  - Easy-to-use but imprecise
  - Ambiguous: unclear information need

- **Automatic keyword query interpretation:**
  - Automatically translate keyword query in the *(most likely)* structured query (-ies) (CNs)
Recap: Candidate Network (CN)

A *Candidate Network (CN)* is a connected tree of keyword relations. Every two adjacent keyword relations $R_i, R_j$ in CN are joined based on the fk-reference in the schema $G_s$. 

![Diagram of Candidate Network (CN)]
Recap: CN ranking factors

Ranking can be performed at **CN and MTJNT levels**

**Typical ranking factors include:**
- Size of the CN tree – preference to the short paths
- IR-Style factors
  - Frequency-based keyword weights
  - Keyword selectivity (IDF)
  - Length normalizations
- Global attribute weight in a database (PageRank / ObjectRank)

Typically, the factors are combined

**Ranking result:** a list of ordered CNs
When ranking alone fails ...
Query interpretation techniques

- **Automatic keyword query interpretation:**
  - Automatically translate keyword query in the *(most likely)* structured query (-ies)
  - **No one size fits all** – no perfect ranking for every query and every user
  - If ranking fails, navigation cost can be unacceptable
    - too many interpretations / search results

- **Interactive query refinement**
  - **Goal:** Enable users to *incrementally refine* a keyword query into the *intended* interpretation on the target database in a *minimal* number of interactions
London... What do you mean?

A book title?

An author name?
The user enters a keyword query in the search box.

*IQ*P suggests a set of query construction options for the user to refine her query. By choosing the correct options, the user can shrink the number of possible interpretations quickly.

*IQ*P interprets the keyword query using database queries, and returns a list of possible interpretations to the user.

The user identifies the correct database query. After double clicking this query, the user is presented with the query results.
$IQ^P$: probabilistic incremental query construction enabling techniques

- A conceptual framework for probabilistic incremental query construction

- A probabilistic model to assess the probability of the possible informational needs represented by a keyword query

- An algorithm to perform the optimal query construction with minimal number of user interactions
Query interpretation

\[ K = \text{“hanks cruise 2001”} \]

\[ \sigma \text{ hanks} \in \text{name (Actor): hanks} \]
\[ \sigma \text{ cruise} \in \text{name (Actor): cruise} \]
\[ \sigma \text{ 2001} \in \text{year (Movie): 2001} \]

\[ T = \sigma \text{ ?} \in \text{name (Actor) \Join Acts} \Join \sigma \text{ ?} \in \text{year (Movie) \Join Acts} \Join \sigma \text{ ?} \in \text{name (Actor)} \]

\[ Q = \sigma \text{ hanks} \in \text{name (Actor) \Join Acts} \Join \sigma \text{ 2001} \in \text{year (Movie) \Join Acts} \Join \sigma \text{ cruise} \in \text{name (Actor)} \]

Partial interpretation of \( K \), sub-query of \( Q \)

Complete interpretation of \( K \) (structured query)

Interpretation space of \( K \)
Query hierarchy

$K = \text{“Tom Hanks 2001”}$
Query construction options (QCO)

Idea: use partial interpretations (sub-queries) as user interaction options (QCO)

Problem: large number of queries – and sub-queries (QCOs)

\[
\sigma_{2001 \in \text{year}(\text{Movie})} : 2001 \\
\sigma_{\text{hanks} \in \text{name}(\text{Actor})} : \text{hanks} \\
\sigma_{\text{cruise} \in \text{name}(\text{Actor})} : \text{cruise}
\]

\[
Q' = \sigma_{\text{hanks} \in \text{name}(\text{Actor})} \bowtie \text{Acts} \bowtie \sigma_{2001 \in \text{year}(\text{Movie})} \\
\sigma_{2001 \in \text{year}(\text{Movie})} \bowtie \text{Acts} \bowtie \sigma_{\text{cruise} \in \text{name}(\text{Actor})}
\]

How to select a QCO to present to the user?
Query construction plan (QCP) as a binary tree

**Idea:** use sub-query relations to organize the QCOs in a (binary) tree structure
The root node is the entire interpretation space

Remove queries conflicting with QCO₁

σₕanks ∈ name(Actor): hanks

QCO₁: σₖ₀₀₁ ∈ year(Movie): 2₀₀₁

Remove queries that subsume QCO₁

σₕanks ∈ name(Actor) ⋈ Acts ⋈ σₖ₀₀₁ ∈ year(Movie) ⋈ Acts ⋈ σₖ₉₃ ∈ name(Actor)

A leaf node is a single complete query interpretation

**Problem:** How to find an optimal QCP?
Defining a cost function for QCP

Idea: define a cost function
Take query probability into account
Construction of the most likely queries should not incur much cost

\[
Cost(QCP) = \sum_{\text{leaf } \in QCP} \text{depth}(\text{leaf}) \times P(\text{leaf})
\]

Given a keyword query \( K \), how to compute the probability of leaf nodes (i.e. complete query interpretations of \( K \))?  

\( K \) (a keyword query) = \{hanks, 2001, cruise\}  
\( Q \) (a leaf node of QCP) =  
\[\sigma_{\text{hanks } \in \text{name (Actor)}} \bowtie \text{Acts} \bowtie \sigma_{2001 \in \text{year (Movie)}} \bowtie \text{Acts} \bowtie \sigma_{\text{cruise } \in \text{name (Actor)}}\]

\( P(\text{leaf}) = P(Q|K) \): the conditional probability that, given \( K \), \( Q \) is the user intended complete interpretation of \( K \).
Query interpretation

A query interpretation consists of:

• A set of **keyword interpretations** $I$ that map a keyword to a value of an attribute (also interpretations as an attribute or table name are possible)

\[ \sigma_{2001 \in \text{year}(\text{Movie})}:2001 \quad \sigma_{\text{cruise} \in \text{name}(\text{Actor})}:\text{cruise} \]

\[ \sigma_{\text{hanks} \in \text{name}(\text{Actor})}:\text{hanks} \]

• A query template $T$

\[ T = \sigma_{? \in \text{name}(\text{Actor})} \bowtie \text{Acts} \bowtie \sigma_{? \in \text{year}(\text{Movie})} \bowtie \text{Acts} \bowtie \sigma_{? \in \text{name}(\text{Actor})} \]
Query interpretation: assumptions

**Assumption 1 (Keyword Independence):** Assume that the interpretation of each keyword in a keyword query is independent from the other keywords.

**Assumption 2 (Keyword Interpretation Independence):** Assume that the probability of a keyword interpretation is independent from the part of the query interpretation the keyword is not interpreted to.
Probability of a query interpretation

\[ P(Q \mid K) = P(I, T \mid K) \]

- \( I \) is the set of keyword interpretations \( \{A_i : k_i\} \) in \( Q \)
- \( T \) is the template of \( Q \)

\[ T = \sigma_{\text{name}(Actor)} \bowtie \text{Acts} \bowtie \sigma_{\text{year}(Movie)} \bowtie \text{Acts} \bowtie \sigma_{\text{name}(Actor)} \]

\[ P(Q \mid K) \propto \left( \prod_{k_i \in K} P(A_i : k_i \mid A_i) \right) \times P(T) \]

Estimates for \( P(T) \) and \( P(A_i : k_i \mid A_i) \)?
We model the formation of a query interpretation as a random process.

For an attribute $A_i$, this process randomly picks one of its instances $a_j$ and randomly picks a keyword $k_i$ from that instance to form the expression $\sigma_{k_i} \in A_i$.

Then, the probability of $P(\sigma_{k_i} \in A_i | \sigma_? \in A_i)$ is the probability that $\sigma_{k_i} \in A_i$ is formed through this random process.

Example:

$T = \sigma_? \in \text{name(Actor)} \bowtie \text{Acts} \bowtie \sigma_? \in \text{year(Movie)} \bowtie \text{Acts} \bowtie \sigma_? \in \text{name(Actor)}$

$Q = \sigma_{\text{hanks}} \in \text{name(Actor)} \bowtie \text{Acts} \bowtie \sigma_{2001} \in \text{year(Movie)} \bowtie \text{Acts} \bowtie \sigma_{\text{cruise}} \in \text{name(Actor)}$

$P(\sigma_{\text{hanks}} \in \text{name(Actor)} | \sigma_? \in \text{name(Actor)})$
Probability of a keyword interpretation

\[ P(\sigma_k \in A_i | \sigma \in A_j) \] can be estimated using Attribute Term Frequency (ATF):

\[
ATF(k_i, A_i) = \frac{TF(k_i, A_i) + \alpha}{N_{A_i} + \alpha \cdot B}
\]

ATF\((k_i, A_i)\) - the normalized keyword frequency of \(k_i\) in \(A_i\)

\(N_{A_i}\) – the number of keywords in \(A_i\)

\(\alpha\) - a smoothing parameter (typically \(\alpha = 1\): Laplace smoothing)

\(B\) – the vocabulary size
Probability of a query template

$$P(T) = \frac{\#\text{occurrences}(T) + \alpha}{N + \alpha \times B}$$

$\#\text{occurrences}(T)$ - number of queries in the log using $T$ as a template

$N$ - total number of queries in the log

$\alpha$ - smoothing parameter, typically set to 1

$B$ – a constant

When the query log is absent or is not sufficient, we assume that all query templates are equally probable.
Query construction algorithm

- Query hierarchy can become very large
- Use a greedy algorithm
- Expand query hierarchy incrementally
- Use a threshold to restrict the size of the top level
- Select the QCO to be presented to the user based on Information Gain (IG)
- IG can be computed using probability of query interpretation
IMDB: 7 tables, 10 million records, 108 queries

Evaluation on IMDB: interaction cost of IQ^P vs. ranking

- The median value of Rank (IQ^P) is 2
- Ranking has a high variance, ambiguous queries can receive ranks above 400
- The entire list can contain up to 3,500 queries
- The cost of construction stays within 15 options in the worst case
User study results: IQP interface vs. ranking

Example queries:

C₀: Find the role of Brad Pitt in the movie directed by Steven Soderbergh in 2004.

C₁₁: Find a movie directed by Blake Blue starring Conners Chad.

✓ 15 graduate CS students.
✓ 14 tasks in 7 complexity categories. Time-based competition. Category k: the correct query interpretation appears at the kth page of the ranking interface.
✓ IQP interface started to outperform the query ranking in C₃ (rank > 30).
✓ In C₁₁, IQP requires 4.3 less time.
IQ$^P$: discussion of results

✓ IQ$^P$ enables users to incrementally refine a keyword query into the intended interpretation.

✓ Interaction cost of IQ$^P$ has a much lower variance than the cost of ranking.

✓ Query construction outperforms ranking for the queries where the ranks of the intended query interpretations are above 30.

✓ Further experiments confirmed quality of the greedy algorithm and scalability for up to 100 tables.
Query construction for large-scale databases

- Freebase (in a relational representation):
  - 22 millions entities, more than 350 millions facts
  - more than 7,500 relational tables
  - about 100 domains
  - Wikipedia, MusicBrainz, ...
  - part of the LOD cloud

- Goal:
  - Enable efficient and scalable query construction solutions for large scale data
A film adaptation starring Tom Hanks was attempted […] after the actor's performances in The Terminal (2004).

Feng Zhenghu has been likened to the Tom Hanks character in The Terminal.

Tom Hanks' character Viktor Navorski is stuck at New York's JFK airport in the United terminal in The Terminal.
Structured MQL query for „Tom Hanks Terminal“

```
{
  "!pd:/film/actor/film": [{
    "name": "Tom Hanks",
    "type": "/film/actor"
  }],
  "film": [{
    "name": "The Terminal",
    "type": "/film/film"
  }],
  "character": {
    "name": null,
    "type": "/film/performance"
  }
}
```

http://www.freebase.com/query

Requires prior knowledge of:
✓ Schema: 7,500 relational tables
✓ Query language: MQL
1. The user enters a keyword query in the search box.

2. FreeQ interprets the keyword query using database queries, and returns a list of possible interpretations to the user.

3. FreeQ suggests a set of query construction options for the user to refine her query.

4. The user identifies the correct database query. After double clicking this query, the user is presented with the query results.
Limitations of the IQP query construction approach

- Inefficient QCOs
  - Too many keyword interpretations
  - A keyword interpretation subsumes a small proportion of the I-space
  - More general QCOs are needed

- Very large interpretation space
  - The number of sub-graphs of the schema graph grows very sharply with the size of the schema graph. The occurrences of keywords are more numerous in a larger database. Too many query interpretations.
  - Existing query interpretation approaches rely on a completely materialized interpretation space. This is no longer feasible.
  - Need to enable incremental materialization of the interpretation space
Query-based QCOs

- Keyword as schema terms or attribute values
  - `actor.name: hanks` (Hanks is in the actor’s name)

```
  hanks → actor.name, director.name, etc.
terminal → film.name, company.name, location.name, ...
```

- Joins using pk-fk relationships in the schema graph
  - `actor.name: hanks` – `acts` – `film.name: terminal`
Ontology-based QCOs

- Freebase domain hierarchy
  - Arts & Entertainment, Society
- External ontologies
  - E.g. YAGO+F mapping between YAGO and Freebase
  - Person, Location, Object
FreeQ query hierarchy example

Ontology-based QCOs:
- Arts & Entertainment: Tom Hanks
  - Actor: Tom Hanks

Query-based QCOs:
- Actor Name: Tom Hanks
- Acts

The arrows represent sub-query relationship
A measure of QCO efficiency

Entropy of the query interpretation space:

\[
H(\zeta) = - \sum_{I \in \zeta} P(I) \times \log_2 P(I)
\]

Expected information gain of a QCO as entropy reduction:

\[
IG(O) = H(\zeta) - H(\zeta|O)
\]

Entropy of the interpretation space given the QCO

\[
H(\zeta|O) = P(O) \cdot H(\zeta_O) + P(\neg O) \cdot H(\zeta_{\neg O})
\]

Probability of the QCO using probabilities of the subsumed query interpretations:

\[
P(O) = \sum_{I \in \zeta(O)} P(I)
\]
Efficient hierarchy traversal

✓ Query initialization:
  ✓ Path indexing: for each table, index all paths leading to keywords within radius $r/2$ (bi-directional):
  ✓ Is independent of keyword query length

$T \ast \text{avg}(E_t)^{r/2}$

✓ User interaction:
  ✓ Use path index to materialize QCOs and query interpretations incrementally by BF-k and DF-k
  ✓ Start expansion with the most probable QCOs
  ✓ Thresholds, time limits
Dataset and queries

Freebase data dump (June 2011). 7,500 tables, 20 mil. entities, 100 domains


<table>
<thead>
<tr>
<th>#Keywords</th>
<th>Avg. #nodes</th>
<th>Examples of keyword queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.00</td>
<td>location, book, event, disease, election</td>
</tr>
<tr>
<td>2</td>
<td>1.43</td>
<td>emperor album, hockey team, alpine skier</td>
</tr>
<tr>
<td>3</td>
<td>1.92</td>
<td>artist lived vancouver, founding figure kagyu</td>
</tr>
<tr>
<td>4</td>
<td>2.40</td>
<td>olympic athletes table tennis</td>
</tr>
<tr>
<td>5</td>
<td>2.11</td>
<td>canada hockey 2010 winter olympics</td>
</tr>
<tr>
<td>6</td>
<td>1.48</td>
<td>2011 san francisco international film festival</td>
</tr>
<tr>
<td>7</td>
<td>1.29</td>
<td>fictional character created by edgar allan poe</td>
</tr>
<tr>
<td>8</td>
<td>1.13</td>
<td>school type university of puerto rico at ponce</td>
</tr>
</tbody>
</table>

✔ higher complexity by [3-5] keywords
Interaction cost of query construction on Freebase

The mean and the standard deviation of the interaction cost

✓ Interaction cost can be significantly reduced and limited to a much smaller range using the ontology-based QCOs.

1 node: football game
2 nodes: ami suzuki album
3 nodes: location leonardo da vinci lived
Efficient hierarchy traversal

- Materializing all complete queries with an expansion radius $r$
  (e.g. Discover)
  \[ T_{k_i} \times \text{avg}(E_t)^r(K-1) \]

- Using path indexing and bi-directional search once at query init
  \[ T \times \text{avg}(E_t)^{r/2} \]

- Using path index to materialize partial and complete query interpretations incrementally and produce top-$k$ results quickly

- Information gain of QCOs can be estimated based on the materialized part of the hierarchy
Response time of query construction on Freebase

- Initial response time mostly stays within 2 seconds
- Interaction response time is always below 1 second

The mean and the standard deviation of the initial and interaction response times

- Initial response time mostly stays within 2 seconds
- Interaction response time is always below 1 second
FreeQ: discussion

- Enables efficient and scalable query construction solutions for large-scale Freebase data
- Involves ontologies e.g. Freebase hierarchy and YAGO+F to summarize database schema using abstract concepts
- Enriches schema with semantic information of YAGO ontology
- Uses efficient algorithms to materialize query interpretation space incrementally
Thank you!

Questions, Comments?

Dr. Elena Demidova

L3S Research Center
Leibniz University of Hannover

email: demidova@L3S.de
www: https://demidova.wordpress.com
Materials used in the slides

References and further reading

References:


Further reading:
