The problem

- Combining querying of XML data with ontology queries
- Example
  - XML document containing recipes
  - Ontology classifying ingredients
  - Query: *Find gluten free recipes*
The approach

- XML query language: Xcerpt
- extended with ontology interface
  - DIG
- Ontology reasoner

Outline

- Preliminaries
  - Xcerpt
  - ontologies and DIG interface
- Extended Xcerpt
  - Answer filtering
  - Ontological information retrieval
- Conclusions
Xcerpt - Introduction

Xcerpt – query and transformation language for XML [Schaffert et al., 2004]

- inspired by logic programming
- uses pattern matching instead of path navigation

- programs consist of query rules \( c \leftarrow Q \)
  - the body \( Q \) used to extract XML data
  - the head \( c \) used to build new XML data

- Core constructs: data terms, query terms, construct terms

Xcerpt constructs

- Data terms
  - model XML documents

  \[
  \begin{aligned}
  <\text{CD} \text{ price}="$10.90"> \text{title}=&\text{Empire Burlesque}<\text{title}> \\
  \text{artist}=&\text{Bob Dylan}<\text{artist}> \\
  </\text{CD}>
  \end{aligned}
  \]

- Query terms
  - used to match data terms
  - successful matching results in variable bindings (answer substitutions), e.g.

<table>
<thead>
<tr>
<th>Query term</th>
<th>Data term</th>
<th>Matches?</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>a[ X ]</td>
<td>a[ &quot;c&quot;]</td>
<td>yes</td>
<td>{ X / &quot;c&quot; }</td>
</tr>
<tr>
<td>a[ X ] ]</td>
<td>a[ &quot;b&quot;, &quot;c&quot;]</td>
<td>yes</td>
<td>{ X / &quot;b&quot; }, { X / &quot;c&quot; }</td>
</tr>
</tbody>
</table>
Xcerpt constructs: Construct Terms and Query Rules

- Construct term
  - used to build data terms (by applying answer substitutions)
  - may contain
    - variables e.g. c[X]
    - grouping constructs all and some

- Query rule  \( c \leftarrow Q \)
  - \( Q \) – query terms
  - connected using and, or …
  - possibly associated with external resources
  - \( c \) – construct term

Ontologies

- Describe application domains
  - Concepts
  - Individuals: instances of concepts
  - Roles: binary relations between individuals

- Specified by special languages e.g. OWL

- Handled by ontology reasoners e.g. RacerPro
**DIG interface**

- ontology reasoner interface
- communication through HTTP POST requests
- XML encoded messages
  - **Tell**: managing the knowledge base
  - **Ask**: querying the knowledge base
  - **Response**: replying to the queries

```xml
<children>
  <catom name="gluten-containing"/>
</children>
<conceptSet>
  <synonyms>
    <catom name="flour"/>
  </synonyms>
  <synonyms>
    <catom name="barley"/>
  </synonyms>
</conceptSet>
```

**Extended Xcerpt**

- Xcerpt + ontology reasoner interface
- Communicates with a reasoner using DIG
- Two methods of interaction
  - answer filtering
  - ontological information retrieval
Answer filtering

Extended Xcerpt rule

Answer filtering – example

Query: Find recipes with ingredients containing gluten

Answer substitutions:

<table>
<thead>
<tr>
<th>R</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>before filtering</td>
<td></td>
</tr>
<tr>
<td>“Recipe1”</td>
<td>“sugar”</td>
</tr>
<tr>
<td>“Recipe1”</td>
<td>“orange”</td>
</tr>
<tr>
<td>“Recipe2”</td>
<td>“flour”</td>
</tr>
<tr>
<td>“Recipe2”</td>
<td>“salt”</td>
</tr>
<tr>
<td>after filtering</td>
<td></td>
</tr>
<tr>
<td>“Recipe2”</td>
<td>“flour”</td>
</tr>
</tbody>
</table>
Ontological Information retrieval

- DIGging rule: \( (h_r \leftarrow b_r) \iff (h_a \leftarrow b_a) \)

- simple DIGging rule:
  \( (h_r \leftarrow b_r) \iff h_a \) (fixed ask expression)

---

Ontological Information retrieval

- DIGging rule: \( (h_r \leftarrow b_r) \iff (h_a \leftarrow b_a) \)

- simple DIGging rule:
  \( (h_r \leftarrow b_r) \iff h_a \) (fixed ask expression)
Ontological information retrieval

- The prototype
  - simple DIGging rules incorporated into Xcerpt goal rules

\[
(h_r \leftarrow b_r) \iff h_a = \begin{array}{c}
\text{GOAL} \\
\text{FROM} \\
\text{in}[ \text{!dig} [ \text{URL}, h_a, b_r ] ]
\end{array}
\]

Ontological information retrieval - example

```
GOAL
  results [ all var C ]
FROM
  in [ !dig [ "http://localhost:14159/",
    asks [
      children [
        attr( id [ "q1" ] ),
        catom ([[ attr { name [ "gluten-free" ] } ]])
      ]
    ],
    responses {{
      conceptSet {{
        attr ( id [ "q1*" ] ),
        synonyms [[
          catom ([[ attr { name [ var C ] } ]])
        ]]
      }}
    ]}
  ]
END
```

Query result:

```
results [
  "water",
  "rice",
  "salt",
  "orange",
  "sugar"
]
```
**DIGging rules limitation**

- variable bindings cannot be passed between ask and response rule e.g.

```plaintext
ask[ instance[ I, C ] ] ← q[ I, C ]

instance[..] ← response{ desc “yes” }
```

**Summary**

- Extension of Xcerpt allowing to communicate with an ontology reasoner
  - answer filtering
    - prototype
      - no grouping constructs in filters
      - filters only in goal rules
  - ontology information retrieval
    - cannot be directly used for filtering (no variable passing)
    - prototype
      - simple DIGging rules
      - response rule only as a goal rule
Related work

- Datalog + DL with logical semantics
  - AL-log [98], CARIN[98], Rosati[05], Motik et al [05]
    - not applicable to Xcerpt + OWL
- Hybrid framework with fixpoint semantics [Assmann et al]
  - ontology reasoning after rule reasoning
- Our approach
  - required modification of Xcerpt system
  - ontology reasoning interleaved with rule reasoning
  - DIGging rules

Future work

- variable passing in DIGging rules
- hybrid implementation
  - DIGging rules handled by external application
- Xquery + ontologies?