

XML Querying Using Ontological Information

Eric Svensson and Artur Wilk

PPSWR'06

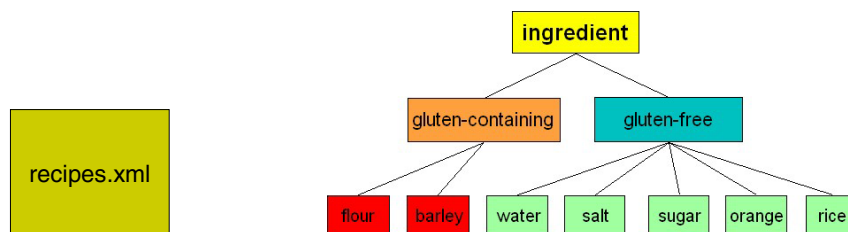
Budva, Montenegro

June 11th, 2006



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

June 11th, 2006

3

Outline



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

June 11th, 2006

4



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

June 11th, 2006

5



Xcerpt constructs

- Data terms
 - model XML documents

```
<CD price="$10.90">
  <title>Empire Burlesque</title>
  <artist>Bob Dylan</artist>
</CD>
```

```
CD[ attr{ price[ "$10.90" ] },
    title[attr{ }, "Empire Burlesque"],
    artist[ attr{ }, "Bob Dylan" ]
]
```

- Query terms

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

Query term	Data term	Matches?	Answers
a[X]	a["c"]	yes	{ X / "c" }
a[[X]]	a["b", "c"]	yes	{ X / "b" }, { X / "c" }

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,

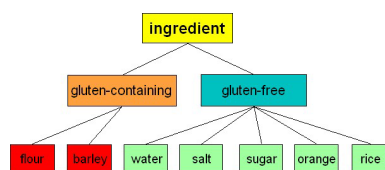
June 11th, 2006

7

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



June 11th, 2006



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

```
<children>
  <catom name="gluten-containing"/>
</children>
```

Ask

```
<conceptSet>
  <synonyms> <catom name="flour"/> </synonyms>
  <synonyms> <catom name="barley"/> </synonyms>
</conceptSet>
```

Response

9



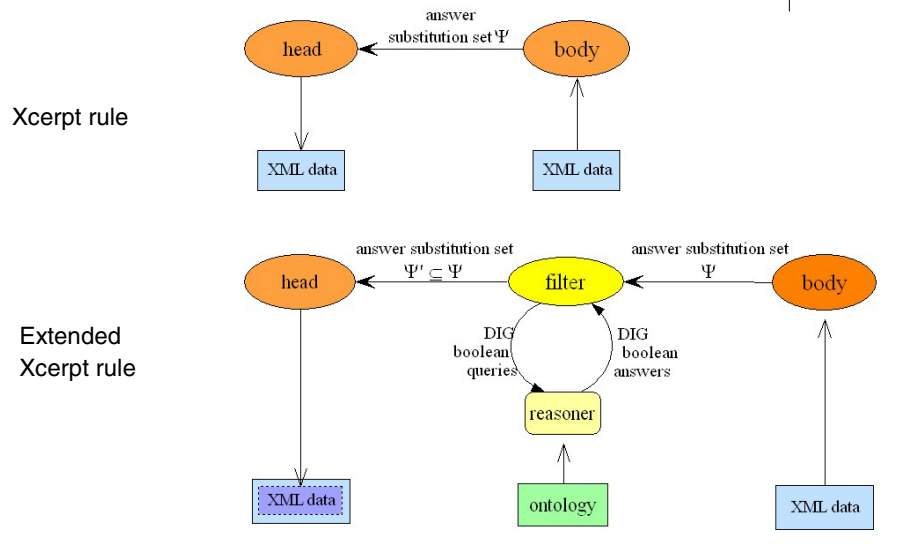
Extended Xcerpt

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

June 11th, 2006

10

Answer filtering



Answer filtering – example



Query: Find recipes with ingredients containing gluten

```
GOAL
names [ all var R ]
FILTER
!dig [
"http://localhost:14159/",
subsumes [
catom [ attr { name [ "gluten-containing" ] } ],
catom [ attr { name [ var N ] } ]
]
]
FROM
in [ resource [ "file:recipes.xml" ],
desc recipe [ [
name [ var R ],
desc ingr [ [ name [ var N ] ] ]
] ] ]
END
```

```
recipes[
recipe[
name[ "Recipe1" ],
ingredients[ ingr[ name[ "sugar" ],
ingr[ name[ "orange" ] ] ] ]
]
recipe[
name[ "Recipe2" ],
ingredients[ ingr[ name[ "flour" ],
ingr[ name[ "salt" ] ] ] ]
] ] ]
```

Answer substitutions:

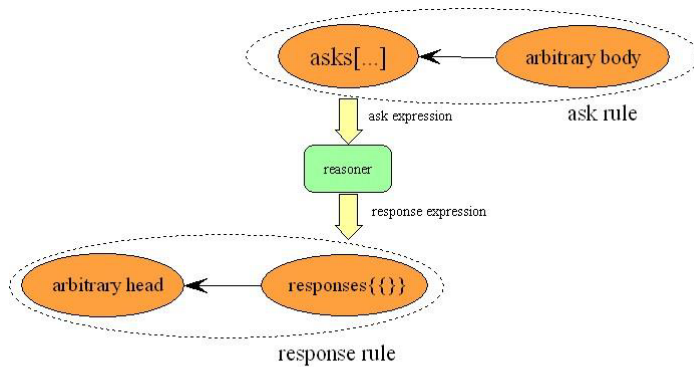
	R	N
before filtering	"Recipe1"	"sugar"
	"Recipe1"	"orange"
	"Recipe2"	"flour"
	"Recipe2"	"salt"
after filtering	"Recipe2"	"flour"

rule result: names ["Recipe 2"]

Ontological Information retrieval



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

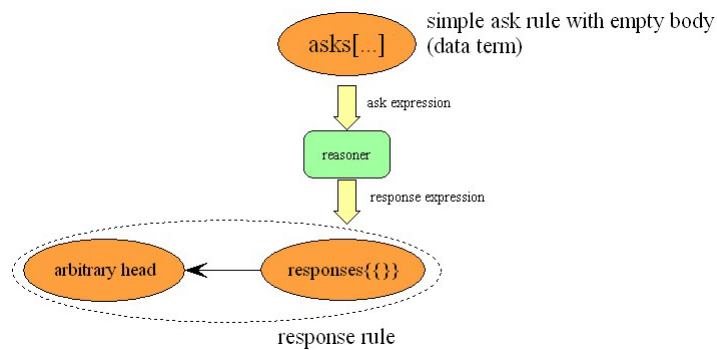


13

Ontological Information retrieval



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



14

Ontological information retrieval



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

$(h_r \leftarrow b_r) \leftarrow h_a =$

```
GOAL
  h_r
FROM
  in[ !dig[ URL, h_a ], b_r ]
END
```

June 11th, 2006

15

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.

`ask[instance[I, C]] ← q[I, C]`



`instance[..] ← response{{ desc "yes" }}`

June 11th, 2006

17



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**

19

Future work



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

June 11th, 2006

20