Advanced XML / Data on the Web

Lecture 4

Lars Birkedal [birkedal@it-c.dk]

The IT University of Copenhagen

Outline of this lecture

- Query Languages:
  - General desiderata for query languages
  - XML-QL (Read about Lorel and UnQL in ABS)
  - XQuery
- Readings:
  - ABS Chapters 4+5.
  - Deutsch. et. al.: A Query Language for XML.
  - XML Revolution Chapter 6

Desiderata for Query Languages

- Expressive Power — at least as expressive as SQL when restricted to relational data
- Semantics — need precise semantics to discuss transformation and optimizations
- Compositionality — output of a query as input for other queries
- Schema — exploiting schema information when available
- Program manipulation — simple core language is enough as most queries are written by other programs

XML-QL

Resources:

- ABS Section 5.1
- Deutsch. et. al.: A Query Language for XML.

XML-QL

- First declarative query language for XML
- Obtained by
  - assuming ssd data model
  - using features from earlier languages
    - patterns
    - templates
    - skolem functions
  - designing XML-like syntax

Patterns in XML-QL

Find all authors who published in Morgan Kaufmann:

```
where <book language = "french">
        <publisher>
          <name>Morgan Kaufmann</name>
        </>
        <author>$A</>
      </book> in "www.a.b.c/bib.xml"
construct <author>$A</a>
```

Abbreviation: </> closes any tag.

Patterns in XML-QL

Find all languages in which Jones' coauthors have published:

```
where <book language=$X>
        <author>$A</author>
      </book> in "www.a.b.c/bib.xml"
      <book>
        <author>$A</author>
        <author>Jones</author>
      </book> in "www.a.b.c/bib.xml"
construct <result>$X</>
```

Constructors in XML-QL

Find all authors and the languages in which they published:

```
where <book language=$L>
        <author>$A</author>
      </> in "www.a.b.c/bib.xml"
construct <result><author>$A$</><lang>$L</></>
```

Result is:

```
<result><author>Smith</author>
  <lang>English</lang></result>
<result><author>Smith</author>
  <lang>Mandarin</lang></result>
<result><author>Doe</author>
  <lang>English</lang></result>
...
```

Nested Queries in XML-QL

Find all authors and the languages in which they published, group by authors:

```
where <book.author>$A</> in "www.a.b.c/bib.xml"
construct <result><author>$A</>
  where <book.language=$L>
          <author>$A</>
        </> in "www.a.b.c/bib.xml"
  construct <lang> $L$ </>
          </>
```

Note: `book.author` is a (regular) path expression.

Nested Queries in XML-QL

Result is:

```
<result><author>Smith</author>
        <lang>English</lang>
        <lang>Mandarin</lang>
        <lang>...</lang>
        ...
</result>
<result><author>Doe</author>
        <lang>English</lang>
        ...
</result>
```

Skolem Functions in XML-QL

Same query, with Skolem functions:

```
where <book language=$L>
        <author>$A</>
      </> in "www.a.b.c/bib.xml"
construct <result id=F($A)>
            <author>$A</>
            <lang>$L</>
          </>
```

Assumptions:

- the ID attribute is always `id`
- default Skolem function for `author` is `G($A)`; for `lang` it is `H($A,$L)`.

Skolem Functions in XML-QL

Object fusion with Skolem functions and block structure — compile a complete list of authors, from two sources:

```
{where <book><author>$A</>
               <title>$T</>
       </> in "www.a.b.c/bib.xml"
 construct <person id=F($A)>
               <name id=G($A)>$A</>
               <booktitle>$T</>
             </>
}
```

Skolem Functions in XML-QL

```
{where <paper><author>$A</>
               <title>$T</>
               <journal>$J$</>
       </> in "www.d.e.f/papes.xml"
 construct <person id=F($A)>
             <name id=G($A)>$A</>
             <papertitle>$T</>
             <journaltitle>$J</>
           </>
}
```

Skolem Functions in XML-QL

Result (some have only book, others only papers, others have both):

```
<person><name>Smith</name>
        <booktitle>Book1</booktitle>
        <booktitle>Book2</booktitle>
</person>
<person><name>Jones</name>
        <booktitle>Book3</booktitle>
        <papertitle>Paper1</papertitle>
        <journaltitle>Journal1</journaltitle>
</person>
<person><name>Mark</name>
        <papertitle>Paper2</papertitle>
        <journaltitle>Journal2</journaltitle>
</person>
```

Skolem Functions in XML-QL

“Wrong” query:

```
where <book language=$L>
         <author>$A</>
       </> in "www.a.b.c/bib.xml"
construct <result id=F($A)>
             <author id=G($A)>$A</>
             <lang id=H($A)>$L</>
           </>
```

What is “wrong” here ?

Skolem Functions in XML-QL

Another “wrong” query:

```
where <book language=$L>
         <author>$A</>
       </> in "www.a.b.c/bib.xml"
construct <result id=F($A,$L)>
             <author id=G($A)>$A</>
             <lang id=H($A,$L)>$L</>
           </>
```

What is “wrong” here ?

Skolem Functions in XML-QL

Yet another “wrong” query:

```
{where <book language=$L> <author>$A</>
       </> in "www.a.b.c/bib.xml"
 construct <author id=F($A)>
             <lang id=H($A,$L)>$L</>
           </>}
{where <person><city>$C</>
                 <fluent-in>$X</>
       </> in "www.a.b.c/bib.xml"
 construct <location id=G($C)>
             <lang id=H($C,$L)>$L</>
           </>
}
```

What is “wrong” here ?

Constructing Trees Only

Three rules to construct only trees using Skolem functions

1 nested elements must have Skolem functions with arguments such that args1 $\subseteq$ args2

```
construct <tag1 id=F($args1)>
            <tag2 id=G($args2)>...</>
          </>
```

2 an element that has an atomic content must have a Skolem function with $X $\in$ args

```
construct ... <tag id=F(args)> $X</>...
```

Constructing Trees Only

3 if a Skolem function occurs in two different places then the following must hold:

- G = H
- args1 = args2
- tag1 = tag2

```
{construct <tag1 id=G(args1)>
           <tag id=F(args)>...</></>}
{construct <tag2 id=H(args1)>
           <tag id=F(args)>...</></>}
```

Break

20 minutes break

XQuery

XML Revolution Chapter 6

Open Issues

- Static type checking of queries: to what extent is it possible ?
- Query optimizations: for which fragments of XQuery, say, can you answer queries efficiently ?