Module 3

XML Processing

(XPath, XQuery, XUpdate)

Part 5: XQuery + XPath Fulltext

Outline

- Motivation
- Challenges
- XQuery Full-Text – Language
- XQuery Full-Text – Semantics and Data Model

Motivation

- XML is able to represent a mix of structured and text information:
  - XML applications: *digital libraries, content management.*
  - XML repositories: *IEEE INEX collection, SIGMOD Record in XML, LexisNexis, the Library of Congress collection, HL7, MPEG7.*
- Need for a language to *search* XML documents

109TH CONGRESS
1ST SESSION

H. R. 2739

To address rising college tuition by strengthening the compact between the States, the Federal Government, and institutions of higher education to make college more affordable.

IN THE HOUSE OF REPRESENTATIVES

MAY 26, 2005

Mr. TIERNEY (for himself, Ms. MCCOLLUM of Minnesota, Mr. GEORGE MILLER of California, Mr. KILDEE, Mr. EMANUEL, Mr. BISHOP of New York, Mr. PAYNE, Ms. WOOLSEY, Mrs. MCCARTHY, Mr. WU, Mr. DAVIS of Illinois, Mr. GRIJALVA, Mr. MEEHAN, Mr. BECERRA, Mr. REYES, Mr. GONZALEZ, Ms. LINDA T. SÁNCHEZ of California, Mr. MCGOVERN, Ms. DELAURO, Mr. OWENS, Mr. HINOJOSA, Mr. KUCINICH, Mr. HOLT, Mr. CASE, Mr. VAN HOLLEN, Mr. ORTIZ, Mr. GUTIERREZ, Mr. CARDOZA, Mrs. JONES of Ohio, Ms. BALDWIN, Mr. WEXLER, Mr. BARROW, Mr. JEFFERSON, Mr. RYAN of Ohio, Ms. SOLIS, Ms. VELÁZQUEZ, and Ms. SCHAKOWSKY) introduced the following bill; which was referred to the Committee on Education and the Workforce

---

A BILL

To address rising college tuition by strengthening the compact between the States, the Federal Government, and institutions of higher education to make college more affordable.

Be it enacted by the Senate and House of Representatives of the United States of America in Congress assembled,

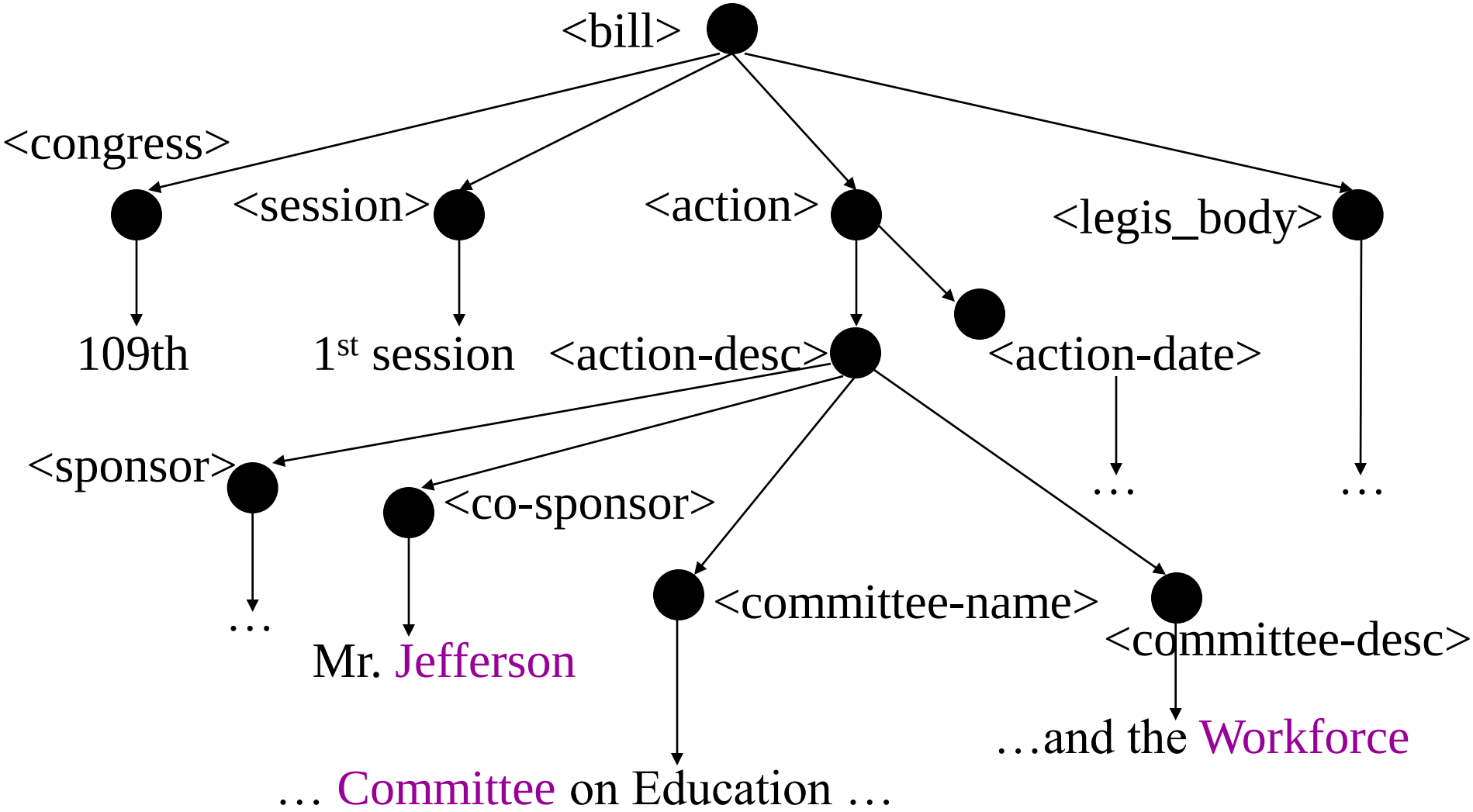
SECTION 1. SHORT TITLE; TABLE OF CONTENTS.

LoC XML Document

http://thomas.loc.gov/home/gpoxmlc109/h2739_ih.xml

```
<bill bill-stage = "Introduced-in-House">
  <congress> 109th CONGRESS </congress>
  <session> 1st Session </session>
  <legis-num> H. R. 2739 </legis-num>
  <current-chamber> IN THE HOUSE OF REPRESENTATIVES </current-chamber>
   <action>
     <action-date date = "20050526"> May 26, 2005 </action-date>
     <action-desc><sponsor name-id = "T000266"> Mr. Tierney </sponsor>   (for
       himself, and <cosponsor name-id = "M001143"> Ms. McCollum of  Minnesota
       </cosponsor>, <cosponsor name-id = "M000725"> Mr. George Miller of
       California </cosponsor>) introduced the following bill; which was referred to the
       <committee-name committee-id = "HED00"> Committee on Education and the
       Workforce </committee-name>
     </action-desc>
    </action>
   …
</bill>
```

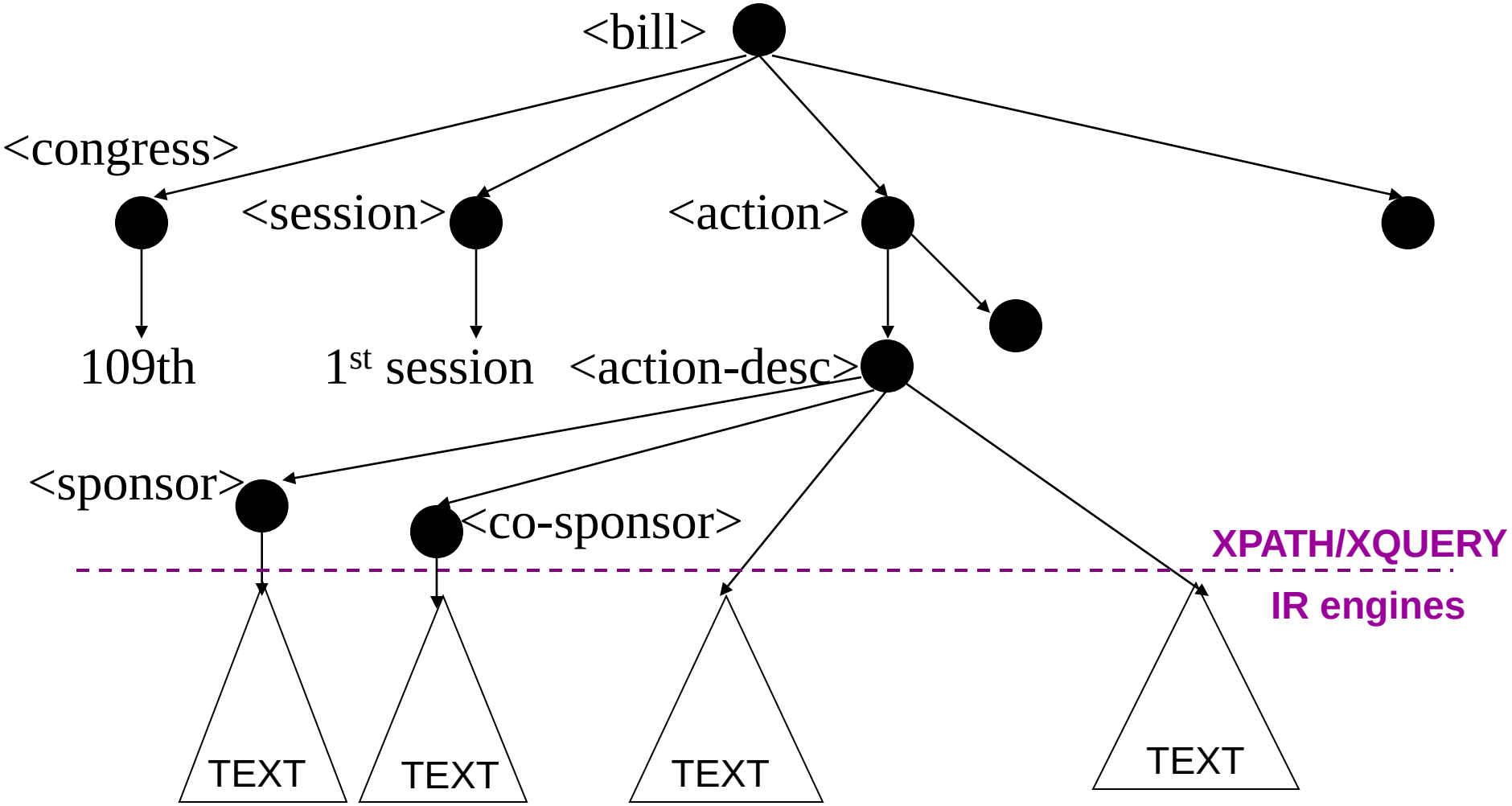
LoC Document Example



Outline

- Motivation
- Challenges
- XQuery Full-Text – Language
- XQuery Full-Text – Semantics and Data Model

Challenges: DB and IR



Challenges

- Searching over Structure+Text
  - *express complex full-text searches and combine them with structural searches*.
  - *specify a search context and return context.*
- Scores and Ranking
  - *Goal: find the most relevant results (remember how Google won over Altavista)*
  - *Typically assign a score value to each item of the result set, order by this value*
  - *In FT*
    - *specify a scoring condition,*
    - *possibly over both full-text and structured predicates*
    - *obtain k best results based on query relevance scores*

Motivation

- Current XML query languages are mostly “database” languages
  - Examples: XQuery, XPath
- Provide very rudimentary text/IR support
  - fn:contains(e, keywords)
  - Returns true iff element e contains keywords
- No support for complex IR queries
  - Distance predicates, stemming, …
- No scoring

W3C

- Full-Text Task Force (FTTF) started in Fall 2002 to extend XQuery with full-text search capabilities: IBM, Microsoft, Oracle, the US Library of Congress.
- First FTTF documents published on February 14, 2004. (public comments are welcome!): http://www.w3.org/TR/xmlquery-full-text-use-cases/

  http://www.w3.org/TR/xmlquery-full-text-requirements/
- XQuery Full-Text highly influenced by TeXQuery.
- Published a working draft describing the syntax and semantics of XQuery Full-Text on July 9, 2004.
- Now a standard:

  http://www.w3.org/TR/xpath-full-text-10/

Example Queries

- From XQuery Full-Text Use Cases Document
  - Find the titles of the books that contain the phrases “Usability” and “Web site” in this order, in the same paragraph, using stemming if necessary to match the tokens
  - Find the titles of the books that contain “Usability” and “testing” within a window of 3 words, and return them in score order
- Such queries are used, e.g. in legal applications

XML FT Search Definition

- *Context expression*: XML elements searched:
  - pre-defined XML elements.
  - XPath/XQuery queries.
- *Return expression*: XML fragments returned:
  - pre-defined meaningful XML fragments.
  - XPath/XQuery to build answers.
- *Search expression*: FT search conditions:
  - Boolean keyword search.
  - proximity distance, scoping, thesaurus, stop words, stemming.
- *Score expression*:
  - system-defined scoring function.
  - user-defined scoring function.
  - query-dependent keyword weights.

Outline

- Motivation
- Challenges
- XQuery Full-Text – Language
- XQuery Full-Text – Semantics and Data Model

Four Classes of Languages

- Keyword search
  "book xml"
- Tag + Keyword search
  book: xml
- Path Expression + Keyword search
  /book[./title about "xml db"]
- XQuery + Complex full-text search

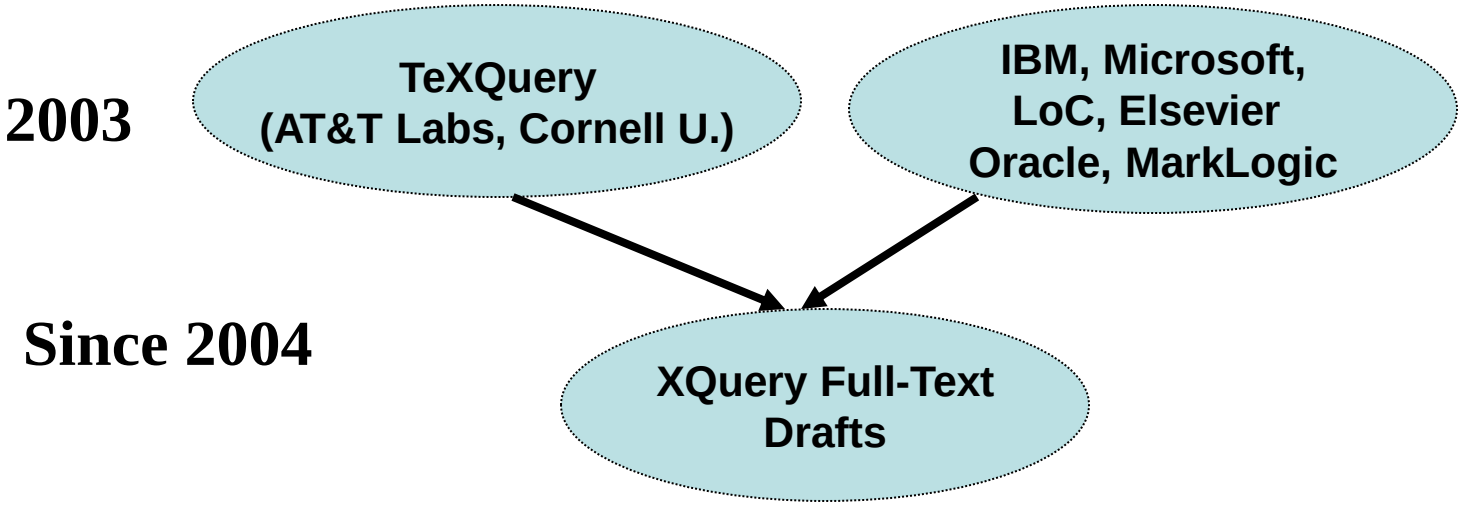
```
for $b in /book
let score $s := $b contains text "xml" ftand "db" distance
   at most 5 words
order by $b
return $b
```

XML Search Languages

- Keyword-only
  - Nearest concept (Schmidt, Kersten, Windhouwer, ICDE 2002)
  - XRank (Guo, Botev, Shanmugasundaram, SIGMOD 2003)
  - Schema-free XQuery (Li, Yu, Jagadish, VLDB 2003)
  - INEX Content-Only queries (Trotman, Sigurbjornsson, INEX 2004)
  - XKSearch (Xu & Papakonstantinou, SIGMOD 2005)
- Tag+Keyword
  - XSEarch (Cohen, Mamou, Kanza, Sagiv, VLDB 2003)
- Path+Keyword
  - XPath 2.0 (http://www.w3.org/TR/xpath20/)
  - XIRQL (Fuhr, Großjohann, SIGIR 2001)
  - XXL (Theobald, Weikum, EDBT 2002)
  - NEXI (Trotman, Sigurbjornsson, INEX 2004)

TeXQuery and XQuery Full-Text

- Extends XPath/XQuery with fully composable full-text primitives.
- Scoring and ranking on all predicates.



http://www.w3.org/TR/xquery-full-text/

Syntax Overview

One new XQuery construct, two extensions

1) FTContainsExpr
   - Expresses “Boolean” full-text search predicates
   - Seamlessly composes with other XQuery expressions
   - Integrates into grammar as comparison
2) Scoring Extensions
   - Extension to FLWOR expression
   - Possible at for and let
   - Can score FTContainsExpr *and* other expressions

FTContainsExpr and Scoring

- FTContainsExpr := RangeExpr (“contains text" FTSelection FTIgnoreOption?)?

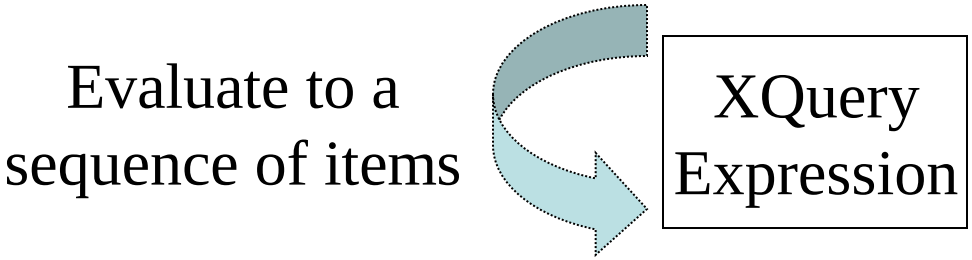
```
books//section [ . contains text ("usability" occurs exactly 4 times
    using stemming ftand "Software" using case sensitive) using
    stop words default window 4 words ordered]
```

- Scoring

```
for $b score $s in
    //books [ ./title contains text "XML" weight 0.4 and .//section
                contains text ("indexing" using stemming ftand
                "ranking" using thesaurus default)
                distance exactly 5 words and ./price < 50 ]
order by  $s
return <result score="{$s}">  {$b/title, $b//authors}  </result>
```

FTContainsExpr

- Like other XQuery expressions
  - Takes in sequences of items (nodes) as input
  - Produces a sequence of items (nodes) as output



- Can seamlessly compose with other XQuery expressions

FTContainsExpr

FTContainsExpr ::= RangeExpr ("ftcontains" FTSelection FTIgnoreOption?)?

- RangeExpression is search context
- FTSelection is search spec
- FTIgnore excludes certain nodes
- Returns true iff at least one node in ContextExpr satisfies the FTSelection

- Examples
  - //book contains text "Usability" ftand "testing"

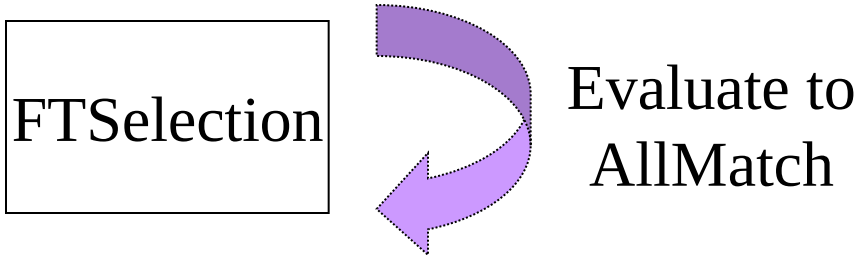
    distance at most 2 sentences
  - //book[./content contains text ‘Usability’ using stemming]/title
  - //book contains text {/article[author=‘Dawkins’]/title}

FTSelection (abbreviated)

- FTSelection := FTOr FTPosFilter* ("weight" RangeExpr)?
- FTOr := FTAnd ("ftor" FTAnd)*
- …
- FTPrimaryWithOptions ::= FTPrimary FTMatchOptions?
- FTPosFilter ::= FTOrder | FTWindow | FTDistance | FTScope | FTContent
- FTMatchOption ::= FTLanguageOption
  | FTWildCardOption
  | FTThesaurusOption
  | FTStemOption
  | FTCaseOption
  | FTDiacriticsOption
  | FTStopWordOption
  | FTExtensionOption

FTSelection

- Encapsulates all full-text conditions in FTContainsExpr
- Works in a new data model called AllMatch
  - Operates on positions within XML nodes (more fine grained than XQuery data model):
  - Fully composable; similar to composition of relational (and XML) operators!



FTSelection Composability

- ‘Usability’
- {/book[author=‘Dawkins’]/title}
- ‘Usability’ ftand {/book[author=‘Dawkins’]/title}
- (‘Usability’ ftand {/book[author=‘Dawkins’]/title}) same sentence
- (‘Usability’ ftand {/book[author=‘Dawkins’]/title}) same sentence window 5 words
- All of these evaluate to an AllMatch!
  - Allows arbitrary composition of full-text primitives

FTMatchOption

- Can be applied on any FTSelection to specify aspects such as stemming, thesauri, case, etc.
  - Fully composable with other context modifiers and FTSelections
- Examples
  - ‘Usability’ ftand ‘testing’ using stemming
  - ‘Usability’ ftand ‘testing’ using stemming using no stop words window 5 words
  - ‘Usability’ ftand ‘testing’ using stemming no stop words using case insensitive window 5 words using

Match Option Details

- Language Option: set language for processing expression, influences stemming, diacritics, …
- WildCardOption: consider wildcards in search term: “use.*”
- ThesaurusOption: use a thesaurus, e.g. for synonyms: “auto” -> “car”
- StemOption: search using the word stem: using and usability come from use
- CaseOption: specifiy how to consider cases, e.g. “Using” and “using”
- DiacriticsOption: consider diacritics, e.g should searching for Rene also return René?
- StopWordOption: words to ignore, typically things like “a”, “the”, …

Score Expressions

In any order

```
FOR  $v [score $s]? [AT $i]? IN Expr
LET   …
WHERE …
ORDER BY …
RETURN
```

Example

```
FOR $b score $s in
      /pub/book[. ftcontains "Usability" ftand "testing"]
ORDER BY $s
RETURN <result score={$s}> $b </result>
```

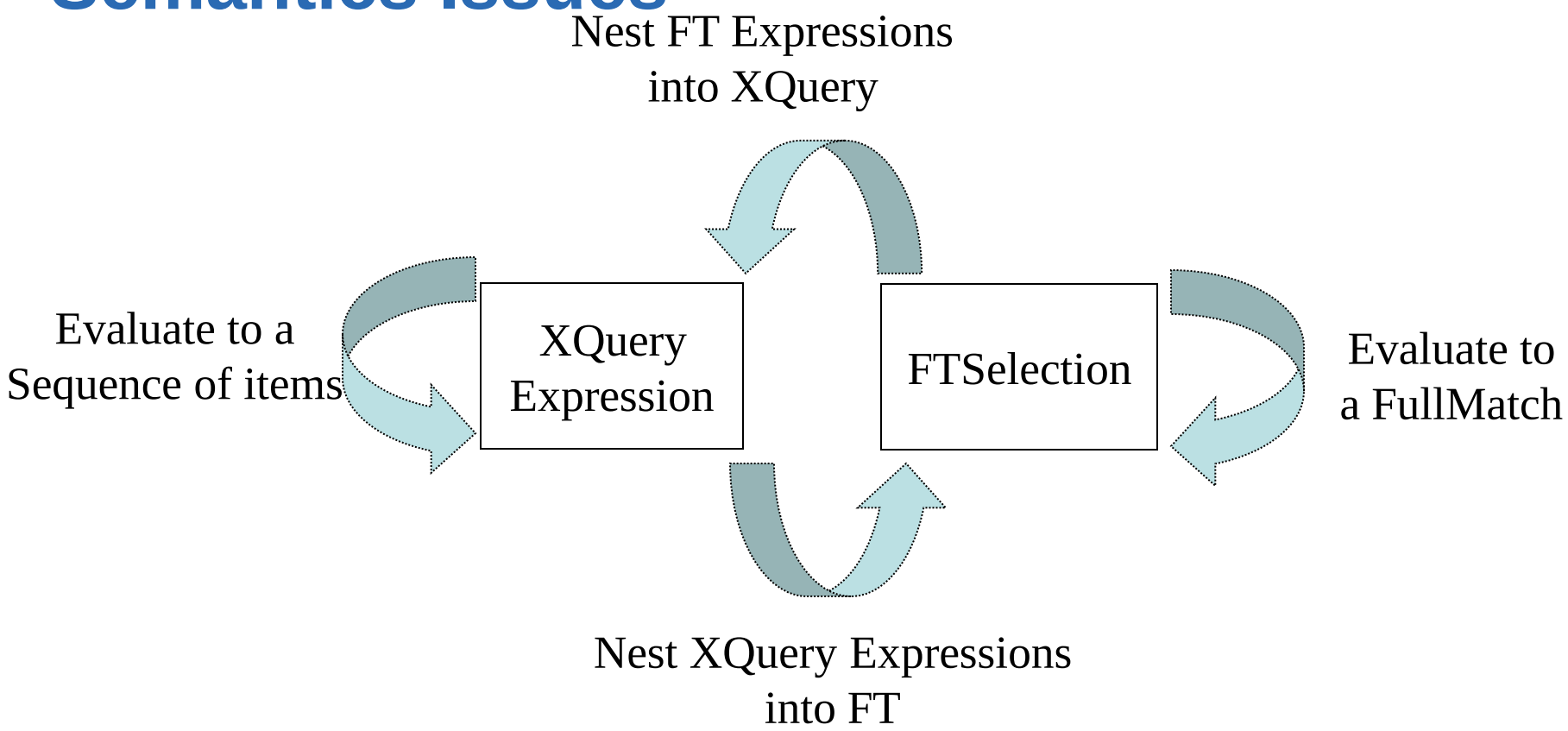
Scoring Notes

- Very much an open research problem
- Actual Scoring mechanism is implementation-defined
- Scoring approaches:
  - TF/IDF : results are relevant, if terms shows up a lot in the result, but not often in the overall document collection. Problems: does not capture structure/context – what is the document collection?
  - Google-Style link analysis: what are the links in a single document?
  - Vector Space Models
  - Structural Properties: words in certain tag or certain position have higher relevance

Outline

- Motivation
- Challenges
- XQuery Full-Text – Language
- XQuery Full-Text – Semantics and Data Model

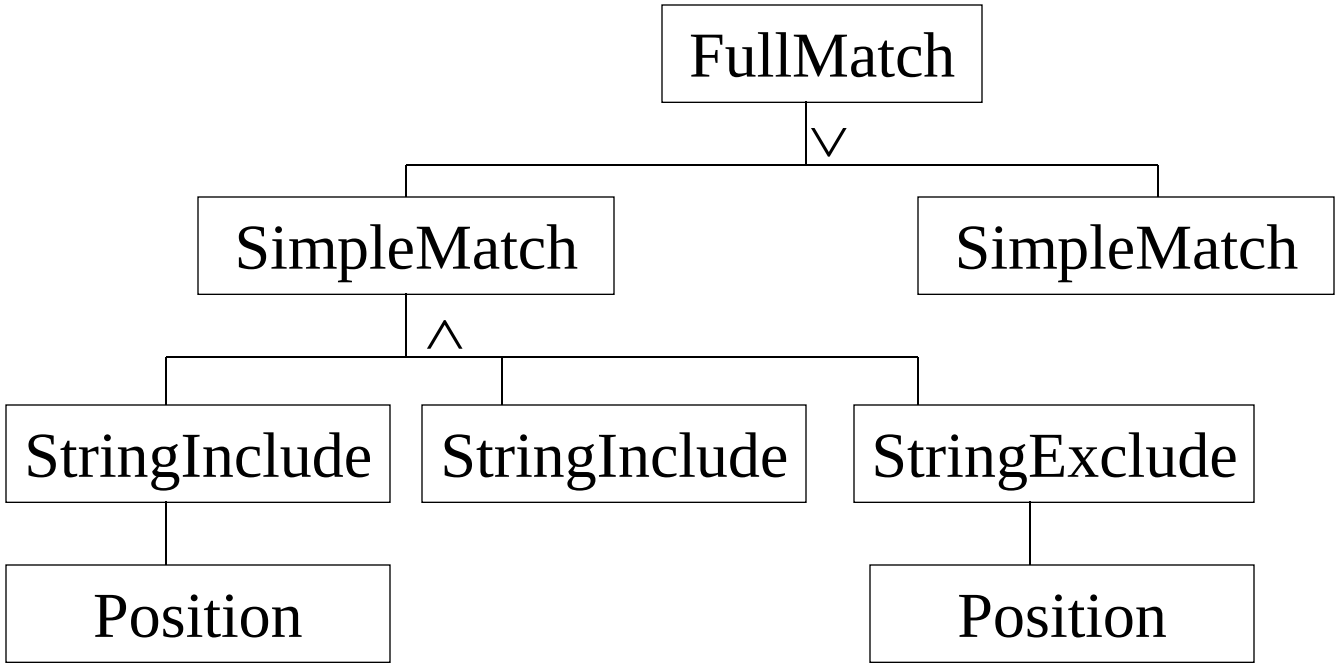
Semantics Issues



FullMatch Overview

- FTSelections are fully composable
- Extensible with respect to new FTSelections
  - Only have to define semantics w.r.t. FullMatch
- Clean way to specify semantics of FTSelections
  - Like specifying semantics of relational operators
- Provides basis for optimizing complex queries

FullMatch



- FullMatch can be interpreted as a propositional formula over word positions in DNF

Sample Document

```
<book(1) id(2)="1000(3)">
   <author(4)>Elina(5) Rose(6)</author(7)>
   <content(8)>
        <p(9)> The(10) usability(11) of(12) software(13)
               measures(14) how(15) well(16) the(17)
               software(18) provides(19) support(20) for(21)
               quickly(22) achieving(23) specified(24)
               goals(25). </p(26)>
        <p(27)>The(28) users(29) must(30) not(31) only(32)
               be(33) well-served(34), but(35) must(36)
               feel(37) well-served(38).</p(39)>
   </content(40)>
</book(41)>
```

N.B. Different document position numbering possible

Sample Query

```
$doc contains text
('usability' using stemming ftand
 'Rose')
 window 10 words
```

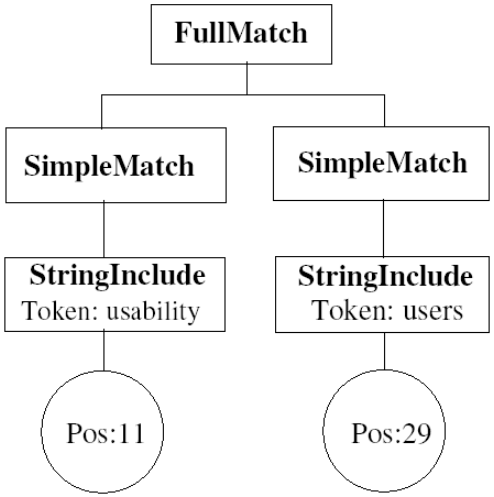
Sample FTSelection

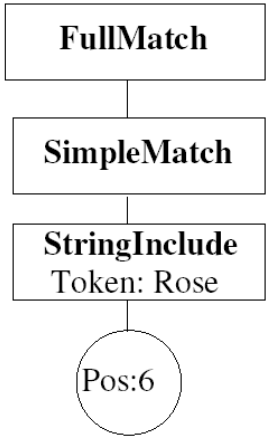
```
('usability' using stemming ftand
 'Rose')
 window 10 words
```

Semantics of FTPrimary

```
<book(1) id(2)=``1000(3)''>
   <author (4)>Elina(5) Rose(6)</author(7)>
   <content(8)>
        <p(9)> The(10) usability(11) of(12) software(13)
               measures(14) how(15) well(16) the(17)
               software(18) provides(19) support(20) for(21)
               quickly(22) achieving(23) specified(24)
               goals(25). </p(26)>
        <p(27)>The(28) users(29) must(30) not(31) only(32)
               be(33) well-served(34), but(35) must(36)
               feel(37) well-served(38).</p(39)>
   </content(40)>
</book(41)>
```

Semantics of FTPrimaryWithOptions





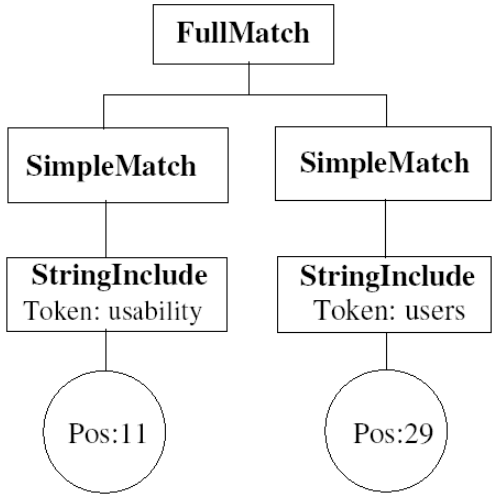
'usability' using stemming

'rose'

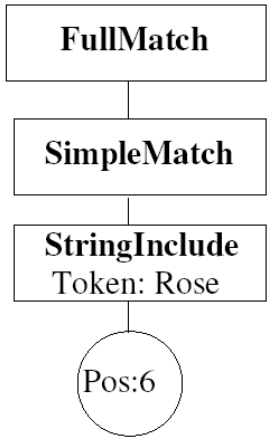
Sample FTSelection

```
('usability' using stemming ftand
 'Rose')
 window 10 words
```

Semantics of FTAnd



×

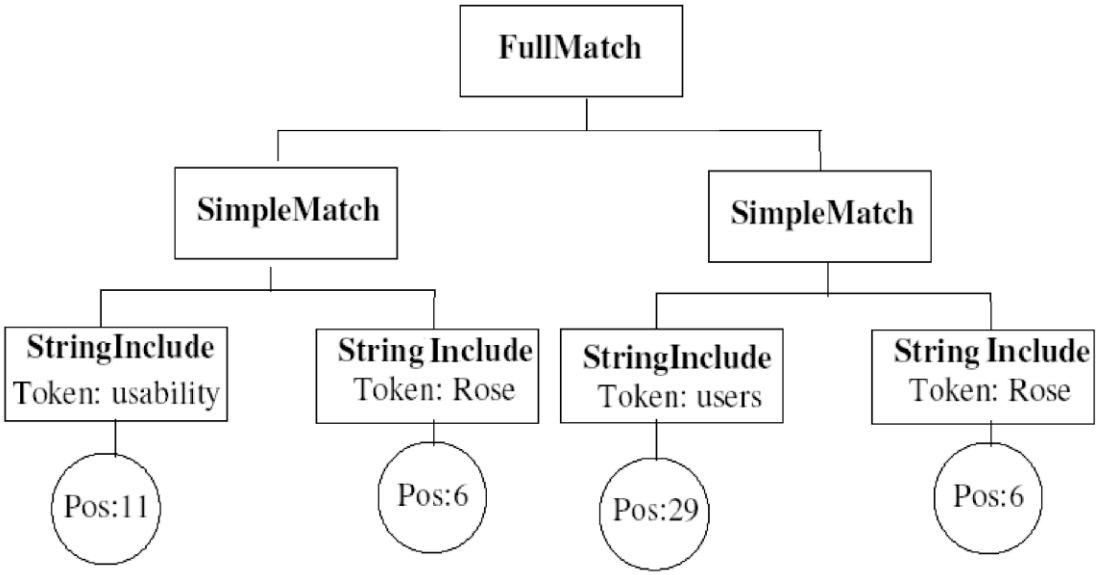


→

'usability' with stemming

'Rose'

Semantics of FTAnd




```
'usability' using stemmming ftand 'Rose'
```

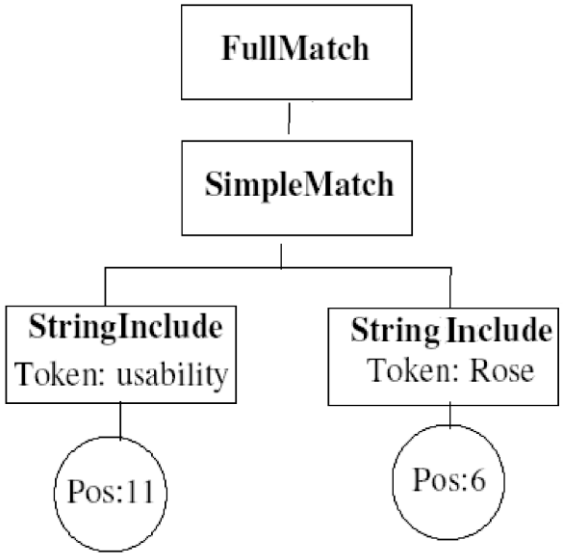
Sample FTSelection

```
('usability' using stemming ftand
 'Rose')
 window 10 words
```

Semantics of FTWindow

```
<book(1) id(2)=``1000(3)''>
   <author (4)>Elina(5) Rose(6)</author(7)>
   <content(8)>
        <p(9)> The(10) usability(11) of(12) software(13)
               measures(14) how(15) well(16) the(17)
               software(18) provides(19) support(20) for(21)
               quickly(22) achieving(23) specified(24)
               goals(25). </p(26)>
        <p(27)>The(28) users(29) must(30) not(31) only(32)
               be(33) well-served(34), but(35) must(36)
               feel(37) well-served(38).</p(39)>
   </content(40)>
</book(41)>
```

Semantics of FTWindow




```
('usability' using stemming ftand 'Rose')
Window 10 words
```

FullMatch Benefits

- FullMatch has a hierarchical structure
- Thus FullMatch can be represented as XML
- Semantics of FTSelections can be specified as transformation from input XML FullMatches to the output XML FullMatch
- Thus, semantics of FTSelections can be specified in XQuery itself!
- Full-text conditions and structural conditions represented in the same framework
  - Enables joint optimization and evaluation

Implemenations

- Galax/Galatex: original reference, not (publicly) updated since 2005
- MXQuery, BaseX, QizX: complete implementation for minimal compliance
- Zorba: support, but no results published

Summary

- Support for "search" and full-text operations in the context of XQuery
- Combine structured search with full-text operations
- Scoring algorithms still an open issue
- Not a replacement for Google-style IR, but a useful additions for large, structured document repositories (laws, patents, libraries)