Vocabulary-independent methods for XML Information Retrieval

24 Aug 2006

Miro Lehtonen

University of Helsinki

Outline

- ➠ Introduction
- ➠ Methods for two challenges
  - → Detecting emphasis in XML documents
  - → Separating full-text content from data
- ➠ Conclusion

XML and Information Retrieval

- Advantages of XML
  - → Element-level search granularity
  - → Search for best entry points (in addition to whole documents)
  - → Queries on document structure
- Areas where XML has potential
  - → Metadata describing document content
  - → Indexing methods
  - → Query evaluation
  - → Ranking algorithms

Terms and definitions

- Vocabulary — *a set of words known to a person or other entity, or that are part of a specific language* [Wikipedia]
- XML vocabulary — a set of “XML Names” that are part of a specific XML language (a document type)
  - → Examples: XHTML, MathML, SVG
- XML Information Retrieval — like any IR, but with a focus on XML documents
  - → Focus in this talk: indexing and ranking methods
- Vocabulary-independence in indexing and ranking methods
  - → Names of XML elements and attributes are ignored

Emphasis on good search terms

- Authors often make the important content explicit to the readers by making it stand out from the surrounding content:

> ...difference in firing modes. Timed (stochastic) PN's use the *strong firing mode* in which a transition is forced to fire immediately after it is enabled...

- What kind of content is emphasised ?
  - Concepts that are essential in their context
    - Often followed by a definition in scientific literature
  - Phrases that are topical in a sentence
    - May disambiguate the meaning of the sentence
  - The same words that are entered as search terms!

Emphasised content in XML documents

- No changes in the typeface:

```
 ...difference in firing modes.  Timed (stochastic) PN’s
use the <it>strong firing mode</it> in which a transition
  is forced to fire immediately after it is enabled...
```

- How to find the emphasised content in XML documents ?
  - Find all `it` elements? At least three problems:
    - Element names depend on the XML vocabulary
    - The whole document could be written in italics
    - Other kind of emphasis is ignored
  - Detect temporary changes in the typeface by finding *inline elements*
    - Different kinds of emphasis are considered the same...
    - ...but inline elements occur in all XML documents with emphasised content!

Inline elements in document collections

- Test documents: 860 IEEE journals in an XML format
  - Inline elements with only one or two characters
  - Elements at the inline level that contain other elements
  - 544,495 inline elements containing at least three characters and no other elements
    - The most common content: “Fig. X” ($>$12%), “Figure X” ($>$6%), “Table X” ($>$3%)
    - Over 180,000 different phrases
  - A substantial amount of high quality index terms
- Other XML document types designed for full-text content look similar

Examples of useful inline elements

```
67 <it>deterministic</it>
65 <it>weight</it>
65 <it>internal</it>
65 <it>fixed</it>
64 <it>functionally redundant</it>
63 <it>capacity</it>
60 <it>minimum</it>
58 <ref>Lamport’s Algorithm</ref>
57 <b>Algorithm</b>
51 <tt>player</tt>
28 <it>sequentially redundant</it>
28 <ref>Sort Partition</ref>
27 <b>primitive</b>
24 <it>dependency relation</it>
21 <it>middle buffering</it>
19 <ref>Hybrid Partition</ref>
17 <b>architecture</b>
17 <it>shape space</it>
16 <it>input buffering</it>
16 <it>false sharing</it>
16 <it>critical path</it>
15 <it>useless shared copies</it>
15 <it>problem complexity</it>
15 <it>perceived usefulness</it>
```

Emphasised content in indexing and ranking methods

- Hypothesis: What is emphasised in the document should be emphasised in the index
  - → Vector Space Model: increase the weights of the emphasised terms
  - → Documents with emphasised search terms are ranked higher than those with unemphasised ones
- Simple and practical method: duplication of qualified inline elements
  - → Increases term frequencies (tf) by 1
  - → Improves phrase detection: typeface does not change mid-phrase
- Previously shown example modified accordingly:

```
 ...difference in firing modes.  Timed (stochastic) PN’s use the
<it>strong firing mode</it> <it>strong firing mode</it> in which
a transition is forced to fire immediately after it is enabled..
```

Duplicating inline elements has a positive effect on retrieval quality

- Retrieval precision improves most at low recall levels
  - Excellent if we only care about the first page of results!
- Further observations
  - Bigger documents benefit more than small ones
  - Triplicating inline elements does not help
  - Finding marginally relevant documents becomes more difficult

Heterogeneous XML documents

- XML as a document format is widespread over different application areas
- Traditional divide: document content (text) vs. database content
- Current and more accurate way of thinking:
  - → “*...there is no longer a difference in kind between the two, only a difference in degree*” [Goldfarb 2003]
  - → “*...difficult distinctions arise in the middle of the document type spectrum where documents contain both narrative and transactional features*” [Glushko and McGrath 2005]
  - → Most XML documents contain both data and full-text
- How to find the indexed content in arbitrary XML documents without losing the independence of XML vocabularies?

Selecting content to be indexed

- Content queried as data should not be indexed as full-text
  - Bibliographies, indices
  - Tables (of contents), charts
  - Other content should be indexed (probably)
- How to recognise data in arbitrary XML documents
  - By the names of the elements? Yes, but...
  - By the proportion of XML elements and text
  - T/E measure: the ratio of Text nodes and Element nodes in a document tree
    - T/E $<$1.00 $\Rightarrow$ data
    - T/E $\geq$1.00 $\Rightarrow$ full-text

An example of data: T/E = 14/21 = 0.67

```
<table border="1">
<caption><em>A test table with merged cells</em></caption>
<tr><th rowspan="2"/><th colspan="2">Average</th>
    <th rowspan="2">Red<br/>eyes</th></tr>
<tr><th>height</th><th>weight</th></tr>
<tr><th>Males</th><td>1.9</td><td>0.003</td><td>40%</td></tr>
<tr><th>Females</th><td>1.7</td><td>0.002</td><td>43%</td></tr>
</table>
```

```
                A test table with merged cells
          /-----------------------------------------\
          |          |      Average      |   Red    |
          |          |-------------------|  eyes    |
          |          |  height |  weight |          |
          |-----------------------------------------|
          |  Males   | 1.9     | 0.003   |   40%    |
          |-----------------------------------------|
          | Females  | 1.7     | 0.002   |   43%    |
          \-----------------------------------------/
```

An example of full-text: T/E = 14/9 = 1.56

```
PN’s use the <it>typeless enabling</it> and <it>firing rules</it>.
In contrast, for those nets following <it>typed enabling rules</it>,
the firing rules also have to be typed. The firing of a typed
transition, <math><tmath>$t_j$</tmath></math>, will remove
specific colored tokens from each input place of
<math><tmath>$t_j$</tmath></math> and add specific colored tokens
into each output place of <math><tmath>$t_j$</tmath></math>.
```

- The “mixed content model” is typical of full-text content, but it rarely occurs in data
- Adding one XML element in text content breaks up a Text node into three new ones
- The proportion of Text nodes increases in the presence of mixed content

Average T/E values by element type (IEEE journal collection)

Tag name	Count	Mean size	Median size	T/E
p	762,223	356.61	281	2.26
ss2	16,288	1,806.20	1,274	1.45
ss1	61,492	2,645.18	1,859	1.44
sec	69,735	4,820.91	2,949	1.43
article	12,107	32,555.31	26,816	1.18
journal	860	458,568.27	422,040	1.18
fm	12,107	756.55	578	0.84
bm	10,065	4,992.83	3,910	0.80
bibl	8,551	2,529.43	1,853	0.76
index	117	20,255.30	14,852	0.88

Advantages of not indexing data for full-text search

- The size of the index is reduced by 5-6%
- The discarded XML includes “noisy” content ⇒ Improved precision of information retrieval

Conclusion

- There are many ways to work with XML without looking at the tag names
  - Analysis of inline elements in detection of emphasis and phrases
  - Content analysis
  - Others, including intra-document link analysis
- The presented methods improve the quality of Information Retrieval in the test environment
- Vocabulary-independent methods apply to any kind of XML
- Questions, comments?