Multiple Sources of Evidence for XML Retrieval

Börkur Sigurbjörnsson
borkur@science.uva.nl

Jaap Kamps[*]
kamps@science.uva.nl

Maarten de Rijke
mdr@science.uva.nl

Informatics Institute, University of Amsterdam
Kruislaan 403, 1098SJ Amsterdam, The Netherlands

ABSTRACT

Document-centric XML collections contain text-rich documents, marked up with XML tags. The tags add lightweight semantics to the text. Querying such collections calls for a hybrid query language: the text-rich nature of the documents suggest a content-oriented (IR) approach, while the mark-up allows users to add structural constraints to their IR queries. We will show how evidence for relevancy from different sources helps to answer such hybrid queries. We evaluate our methods using the INEX 2003 test set, and show that structural hints in hybrid queries help to improve retrieval effectiveness.

Categories and Subject Descriptors: H.2 [Database Management]: H.2.4 Query processing; H.2.8 Database Applications; H.3 [Information Storage and Retrieval]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software; H.3.7 Digital Libraries

General Terms: Experimentation.

Keywords: XML Retrieval, XPath, Content and structure.

1. INTRODUCTION

Document-centric XML documents contain text, marked up with XML tags, enriching the the text with lightweight semantics. The markup can be exploited in several ways. Retrieval engines can use specific tags to try to boost retrieval effectiveness, as illustrated by the effectiveness of using anchor text in web retrieval. If the user is aware of the structure she can try to add structural constraints to her *content*-query in order to help pinpointing her information need. The constraints can be both on the *granularity* of results (i.e., the requested unit of retrieval), and on the *environment* in which they appear.

We will focus only on elements fulfilling the granularity constraint, and investigate ways of ranking them w.r.t. how well they answer the information need expressed in the query. We base our ranking of an element on evidence for relevancy from various sources: (1) the content of the *element* itself; (2) the content of the surrounding *document*; and (3) the content and structure of the *environment* in which the element resides.

We evaluate our ranking methods using the INEX 2003 XML retrieval test-suite [2]. The INEX document collection contains over 12,000 computer science articles from 21 IEEE Computer Society journals, marked up with XML tags. On average a document contains 1532 elements and the average element depth is 6.9. The markup has around 170 tag names, such as articles ⟨article⟩, sections ⟨sec⟩, author names ⟨au⟩, affiliations ⟨aff⟩, etc. All our retrieval runs are created using a multinomial language model [1].

[*]Currently at Archives and Information Studies, Faculty of Humanities, University of Amsterdam.

2. CONTENT-AND-STRUCTURE

The INEX initiative combines database and information retrieval approaches for querying document-centric XML documents. The INEX test suite provides two sets of topics. Content-Only (CO) topics are like traditional IR topics. Content-And-Structure (CAS) topics express the information need as a combination of content and structural properties of relevant elements. The query language is a dialect of XPath [4], extended with a function for content-oriented search. The semantics of this query language is in the spirit of information retrieval. A human assessor decides about the relevancy of XML elements.

CAS topics contain the three traditional fields: title, description and narrative. The description and narrative describe the information need in natural language, while the language of the title is the XPath-dialect. As an example, let's look at a topic with the description field:

(Q) Find articles where the author is affiliated in California. From those articles, return sections about weather forecasting systems.

Its title field would be something like:

```
//article[about(./fm/au/aff,'California')]//sec[about(.,
     'weather forecasting systems')]
```

This query contains a mixture of content and structural constraints. The `about`-functions are seen as IR search functions. We refer to the path `//article//sec` as the *granularity constraint*. We refer to the about-functions as *environment constraint*; and to the paths `//article/fm/au/aff` and `//article//sec` respectively as the *path constraints* of the `about`-functions.

3. EXPERIMENTAL SETUP

In order to collect evidence from multiple sources, our retrieval engine will break each topic up in to two types of IR queries. The *full content* query merges the text from the title and description:

(Q1) california weather forecasting systems find articles where the author is affiliated in california from those articles return sections about weather forecasting systems

We use the full content queries both to rank documents, and to rank elements that fulfill the granularity constraint. For each about-function there is a *partial content* query containing the content part of the function:

(Q1a) california

(Q1b) weather forecasting systems

We use the partial content queries to rank elements that fulfill the path constraint of the respective about-function.

Since we want to use those queries to collect evidence from both the document and element level we build two indexes. We build a *document index* as used for standard document retrieval. Additionally we build an *element index*, where we index each XML element separately. For each element, all text nested inside it is indexed. Hence the indexing units overlap. Text appearing in a nested XML element is not only indexed as part of that element, but also as part of all its ancestor elements.

4. EXPERIMENTS AND DISCUSSION

As pointed out before, we look at three sources of evidence for relevancy, where in all cases we will only return elements fulfilling the *granularity constraint*. First, we consider evidence from the surrounding document alone. We create a *document-based run*, where we rank all elements using the score that the surrounding document gets when we retrieve from the document index. Here we use the *full content* query. This is the obvious retrieval strategy to follow if we only have a document retrieval system. Second, we look at evidence from the element that is to be retrieved. We create an *element-based run*, where we rank all elements using the score they get when we retrieve from the element index. Here we use the *full content* query. Third, the structural queries allow us to look at yet another source of evidence for relevancy, namely from the *environment* in which the result elements reside. For each about-function, we rank all elements that fulfill the respective *path constraint*, using the score they get when we retrieve from the element index. Here we use the respective *partial content* queries. We use these scores to measure how well elements meet the constraints put on their *environment*.

Run	MAP		P@10
Document-based run	0.2321		0.2240
Element-based run	0.2633	+13.4%	0.2640
Environment-based run	0.3090	+33.1%	0.2760
Mix-three-levels	0.3201	+37.9%	0.3000

Table 1: MAP and precision at 10 for the runs, the improvement is relative to the document-based run

The first three rows of Table 1 show the results of three runs, each based on a different source of evidence for relevancy. The document-based run is quite competitive, but the element-based and the environment-based runs give significantly better results. The environment-based run is 17.4% better than the element-based run, but the difference is not significant. Note that the environment-based run also considers the content of the retrieved element, since the topics usually have an about-function that is evaluated on the target element (such as the query Q1b above).

Our document-based run seems to be more competitive than a similar method reported in [3]. Note however that neither the collection nor the task is the same. Also, one-third of the topics asked for articles. The improvement comes thus only from the remaining two-thirds of the topics.

Looking at each source of evidence individually is not likely to be an optimal strategy. Thus we create a *mixed run* where we merge all the above sources of evidence. We simply rank elements by adding up the scores from each run. The result is shown in the last row of Table 1. The mixed run performs between 3.6% and 37.9% better than the underlying runs. The difference is significant over the document-based and the element-based runs.

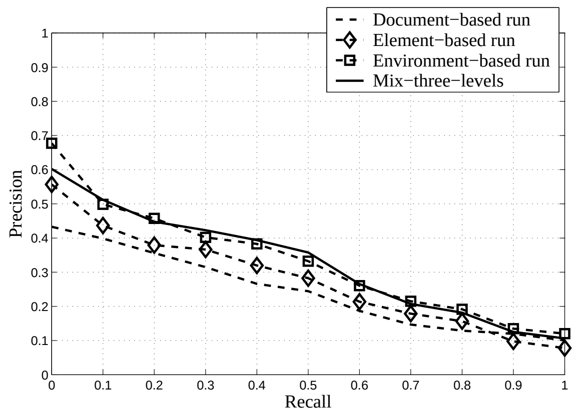

Figure 1: Precision-recall graphs for our runs

Figure 1 shows the precision-recall curves for our runs. The element-based and the environment-based runs have better initial precision compared to the document-based run. This is reasonable since the document-based run cannot distinguish between elements that appear in the same document.

5. CONCLUSIONS

We have seen that the structure of XML collections and queries can give natural ways to gather evidence for relevancy from multiple sources. The three different sources we explored here give different results, and are all of reasonably high quality. A simple combination of the sources gives further improvement, and outperforms the best official run at INEX 2003. Our results show that if users can add structural requirements to a content-based query in order to make a clearer description of their information need, a search engine should make use of those requirements to improve search results.

6. REFERENCES

[1] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, 2001.

[2] INitiative for the Evaluation of XML Retrieval, 2003. `http://inex.is.informatik.uni-duisburg.de:2003/`.

[3] R. Wilkinson. Effective retrieval of structured documents. In *SIGIR 1994*, pages 311–317, 1994.

[4] XML Path Language (XPath), 1999. `http://www.w3.org/TR/xpath`.