Reprinted from: H. L. Larsen, G. Pasi, D. Ortiz-Arroyo, T. Andreasen and H. Christiansen (eds), *Flexible query answering systems, Milan, June 2006*. Springer, 2006. (pp 121-132).

XML-structured documents: retrievable units and inheritance

Stephen Robertson[1], Wei Lu[2], and Andrew MacFarlane[3]

[1] Microsoft Research, 7 JJ Thomson Avenue, Cambridge, UK
and City University
ser@microsoft.com

[2] Center for Studies of Information Resources, School of Information Management, Wuhan University, China
and City University
sa713@soi.city.ac.uk

[3] Centre for Interactive Systems Research, Department of Information Science, City University, London, UK
andym@soi.city.ac.uk

Abstract. We consider the retrieval of XML-structured documents, and of passages from such documents, defined as elements of the XML structure. These are considered from the point of view of passage retrieval, as a form of document retrieval. A retrievable unit (an element chosen as defining suitable passages for retrieval) is a textual document in its own right, but may inherit information from the other parts of the same document. Again, this inheritance is defined in terms of the XML structure. All retrievable units are mapped onto a common field structure, and the ranking function is a standard document retrieval function with a suitable field weighting. A small experiment to demonstrate the idea, using INEX data, is described.

1 Introduction

In this paper, we explore the translation of some current ideas in text and document retrieval to an environment with XML-structured documents. Specifically, we consider various parts of the document (defined as elements in the XML structure) as retrievable units in their own right, in a similar fashion to the retrieval of passages from unstructured documents. However, the XML provides us with the means to define *inheritance*, whereby an element can inherit information from other elements. The use of inherited information in a ranking function requires a good way to weight the information from different sources; this is provided by a form of field-weighting used in more conventional text-retrieval settings.

To illustrate some of these ideas we make use of the INEX document collections and experiments. However, we first set the scene with an overview of a number of facets of retrieval in the domain of text documents, which may be unstructured or have minimal explicit structure.

2 Background

'Unstructured' text documents have been the subject of retrieval systems and experiments for many years. From experiments on small collections of scientific abstracts in the 1960s, through mid-size collections of news and news-wire materials to huge collections of heterogeneous web pages today, we have very considerable experience and understanding of how to retrieve from such collections. Although there are very many variants and alternatives, the dominant approaches may be summarised as follows:

- Unstructured text queries;
- 'Bag-of-words' indexing of documents;
- Statistically-based scoring functions (to score each bag-of-words document against the current query);
- Ranked output

This is a broad characterisation – maybe a caricature – but does capture some of the important features of the experience and understanding.[4] In particular, the very cavalier 'bag-of-words' approach, which ignores many intuitions and understandings about language and meaning in other contexts, nevertheless has been remarkably successful for search. On the other hand, the necessity for good scoring functions is paramount.

That said, in the following sections we explore some of the variants and alternatives which modify this dominant view, in preparation for discussing our approach to XML-structured documents.

2.1 Passage retrieval

The retrieval of (sub-document) passages has been an occasional interest within the document retrieval community for many years. One issue here has been that very few evaluation collections have relevance judgements at the passage level; however, passage retrieval has been tested as a method of doing document retrieval. Many of the problems and possibilities of using passage-level evidence, including all the ideas mentioned here, are discussed by Callan [1].

At the simplest level, passages may be defined more-or-less arbitrarily (for example in terms of fixed word-length windows on the text, or by means of relatively superficial parsing such as sentence or paragraph separation). Then each document is retrieved on the basis of the score of the best-matching passage within it, rather than on the basis of scoring the entire document. The model is effectively that the passage is the relevant part of the document, but since we have relevance judgements at document level only, we have to embed the task in a document retrieval one.

A variation on this theme is to mix the scores of the best-matching passage and the whole document. We might see this as based on the same model (the

[4] The notion that text itself is 'unstructured' is of course a gross over-simplification. However, we do not address the linguistic structure of text in the present paper.

passage is the relevant part), but with the additional assumption that the rest of the document also tells us something, provides some context, about the passage. Thus we evaluate the score of the passage, but allow it to *inherit* information from the document. This intuition informs the present paper.

2.2 Fields

One of the most obvious departures from the simplest bag-of-words model is the notion that some parts of the document/record are more informative than others. For a scientific paper, for example, we may guess that occurrences of words in the title or perhaps abstract are stronger indicators of the content of the paper than occurrences in the body text. We may operationalise this intuition by identifying a number of 'fields' or 'streams' in the documents – that is, a structure of fields that may be identified in or applied to every document in the collection. This is an important notion, whose value has been demonstrated both in formal experiments such as TREC (see e.g. [2]) and by the web search engines. In the latter case, the field structure must be applied to a very heterogeneous collection of documents. This implies a mapping of the features of each document type onto the common field structure. This mapping is typically a manual operation.[5]

The interaction of fields and bag-of-words scoring methods is not entirely trivial. A common approach has been to construct a separate bag-of-words for each field, derive separate scores for the query against each field, and then combine them (usually a weighted linear combination) into a single document score (see e.g. [4]). However, this approach conflicts with some characteristics of good bag-of-words scoring functions [5]. Instead, we need to construct a bag-of-words representation of the whole document which reflects the relative weights of the fields. Below, we use the BM25F method of doing this.

2.3 Inheritance between and within documents

One of the techniques which has amply proved its value in the web context is the inheritance of anchor text. Web search engines typically place heavy reliance on this technique. The principle is that an HTML link between documents (the `<a href="..."> ... </a>` tag) encloses a piece of text in the source document which describes the destination document. The technique is to associate this piece of text with the destination document. Thus each document contains a single field which is the accumulation of all the anchor text from incoming links.

Experiments on web data [6] indicate that (despite the noise it may contain) this is an extremely valuable clue: anchor text is typically weighted highly, considerably higher than body text. Clearly, it depends on using a good form of field weighting.

A number of different approaches have been used at INEX to tackle the issue of inheritance within documents. Arvola et al [7] use a contextualisation function which uses the sum of BM25 weights divided by number of elements in

[5] See for example the mapping in [3].

which a weight is generated for a given term. Both Ogilvie and Callan [8] and Sigurbjornsson et al [9, 10] generate language models either for each element of the XML tree or a chosen subset of elements and use linear interpolation on the models to combine the evidence. Mass and Mandelbrod [11] create a different index for each element type, and use the vector space model, simply combining the scores of the cosine correlation function. They revised this method using linear interpolation on scores for each element in [12] to achieve improvements of average precision in the range 30%-50%.

2.4 Tuning

Any scoring function that contains free parameters must be tuned. For example, if we distinguish five fields, then we have four or five free field weights to choose (we may fix one of the fields at weight 1). Most basic bag-of-words scoring functions also contain their own free parameters.

The usual way to do this is to use some set of evaluation data (queries and relevance judgements) as a training set, and discover the combination of values of the free parameters that optimises some evaluation measure on this training set. There is a whole host of issues associated with this approach [13], which will not be further explored in the present paper.

Some researchers working with INEX data have tunable parameters in their models; in some cases, they have undertaken tuning experiments. Arvola et als [7] contextualisation function allows for each element to be weighted in different ways. A parameter can be associated with each element's language model in the linear interpolation method used in [8–10, 12]. Mass and Mandelbrod [12] tune the parameters for the vector space linear interpolation method to good effect (see section 2.3 above).

3 Passages and inheritance in XML documents

We assume that we have text documents with XML structure. That is, the content of the documents is (largely or entirely) text, quite possibly including relatively large blocks of undifferentiated text like paragraphs, but with some overall XML structure. Blobs of text are held within XML elements.

3.1 INEX documents

We take the documents used in the INEX 2005 experiments to exemplify the idea; these are scientific papers and other contributions published in IEEE journals. They have XML structure that combines traditional structural elements (title, abstract, sections at various levels, bibliography etc.) with presentational matters (emphasis, font changes, mathematical material etc.). In fact structural and presentational matters are mixed with gay abandon: for example several different elements define 'paragraphs', either in particular presentation styles or with attributes which define the style.

The outline structure of an example INEX document is shown in Figure 1. In this outline, lower levels of the structure and all text have been removed; multiply-occurring elements have been reduced to one or two.

It seems likely that some structural elements would be very useful for part-document retrieval. We might reasonably define sections or some level of subsections as suitable units for retrieval; possibly also other elements such as bibliographic references or figures. However, it is equally clear that not all the XML elements are suitable for this purpose; consider the followin extreme example of a section title: `<st>C<scp>oncluding</scp> R<scp>emarks</scp></st>`. The `<scp>` tag represents a font (small capitals), and the enclosed text is a fragment of a word only, quite unsuitable as a retrievable unit.

Our choice is to make complete articles, sections, subsections at all levels and paragraphs into retrievable units. This is not to say that this choice is correct; it stresses the fact that this choice has to be made, and cannot simply be inferred from the XML structure. Retrievable units defined for our INEX 2005 experiments [14] included elements in the `<bm>` (back matter) tag and the abstract, but we might better consider these as ancillary matter, to be inherited as appropriate by other retrievable units, but not to be retrieved in their own right.

3.2 Inheritance

We have already discussed the notion of *fields* in a document. In this context, a very obvious field to consider is *title*. Complete articles have titles; sections and subsections also have or may have titles.

However, the interesting question arises concerning the relation between (say) the article title and the sections of the document. The article title describes the entire article, and therefore presumably also forms part of the description of each section or subsection within it. Similary, the main section title contributes to the description of the subsections.

Rather like the role of anchor text in web retrieval, we might reasonably assume that this information should be inherited appropriately. That is, if we consider a section as a retrievable unit, we should allow it to inherit information from outside itself that might be assumed to describe it. Thus sections should inherit article titles as well as their own; subsections should inherit both article and section titles. There may be other elements that should be inherited by the different levels: as indicated above, the abstract and bibliographic references are obvious examples.

The notion of inheritance may be compared to the mixing of whole-document score and passage score for passage retrieval; however, it is a more powerful and perhaps cleaner notion. It makes it quite clear that the passage (section, paragraph) is the retrievable unit; the role of the rest of the document is to tell us more *about the passage*. It also implies that if we are considering the passage for retrieval, we may choose to allow it to inherit selectively from the rest of the document, rather than simply inheriting its score.

```
<article>
   <fno/>
   <doi/>
   <fm>
      <hdr> info about journal in which article appears </hdr>
      <tig> the title </tig>
      <au sequence="first"> 1st author name, affiliation </au>
      <au sequence="additional"> ditto for 2nd author </au>
      ...
      <abs> abstract as paragraphs of text </abs>
      <kwd> list of keywords </kwd>
   </fm>
   <bdy>
      <sec>
         <st> section title </st>
         <p> etc -- paragraphs of text;
                    bulleted lists; figures
      </sec>
      <sec>
         ...
         <ss1> (subsection)
            <st> subsection title </st>
            <p> etc -- paragraphs of text;
                       bulleted lists; figures
         </ss1>
         <ss1> ... </ss1>
      </sec>
      ...
   </bdy>
   <bm>
      <footnote id="T02001aff"> paragraphs etc. </footnote>
      <bib>
         <bibl>
            <h> header </h>
            <bb id="bibT02001"> bibliographic details </bb>
            <bb id="bibT02002"> ditto </bb>
            ...
         </bibl>
      </bib>
      <vt id="T0200a1"> author 1 details and picture </vt>
      <vt id="T0200a2"> ditto author 2 </vt>
      ...
   </bm>
</article>
```

Fig. 1. Outline of an example INEX document. Lower levels of the structure (including all presentation elements) and all text have been removed; many multiply occurring elements have been reduced in number; notes indicate the content of some elements in this example.

3.3 Field structure

Following the discussion above, we would like to regard each retrievable item as a kind of field-structured document. Given a field structure, we have good methods for scoring the unit against a query. One issue, as indicated above, is that we need every retrievable item in the collection to be mapped onto *the same* field structure. Clearly, it is possible for a given field in a given item to be empty, but insofar as the different items will be competing against each other for retrieval, we need to ensure that items are balanced as far as possible.

In experiments for INEX 2005 [14], as an example of part of a mapping, we had a field for article title, for all retrievable elements. In other words, the article title was assumed to play the same role with respect to articles and to smaller units (sections and paragraphs). Section titles went to another field. Here we have reconsidered this mapping: a possible view is that a section title has the same relationship to the section in which it belongs as an article title has to the article. The relation of the article title to the section is different.

Another issue here is training. We need at the least a trainable weight for each separate field (possibly fixing one of them to 1). The more parameters, the more difficult training is and the more training data is needed. There is therefore good reason to start at least with a small number of fields.

Table 1 shows both the inheritance and field structure used in the present experiments. Again, it is not necessarily the best way to do it – it is offered as an example of how it might be done. Again, it is necessary to point out that this is a manually defined structure – we have no automatic methods of doing this.

	Field		
Retrievable element	Current title	Parent title(s)	Body text
article	article title	–	all text in article
section	section title (if present)	titles of article and all ancestor sections	all text in section
paragraph	–	ditto	all text in paragraph

(a dash indicates that the field is left empty)

Table 1. Retrievable elements mapped onto fields

4 Experiments

In this section, we describe a small experiment to illustrate some of the ideas presented above.

4.1 Data

The general XML structure of INEX documents has already been described. INEX 2005 provides both suitable test data (documents and topics) and a suitable methodology for trying out some of the above ideas. We use the CO (content only) topics. The data consists of 16819 articles, 40 topics and relevance judgements (see further below on the topics and judgements). Under our definition of retrievable unit (article, section, all levels of subsection and all types of paragraph), we identify 1529256 retrievable units in total.

We also need to perform some training in order to establish suitable values for the free parameters, particularly field weights. As is usual when training parameters, we would like to test the resulting trained algorithms on a different test set, to avoid overfitting. However, the set of topics for INEX 2005 CO is quite small (40 topics), and in fact only 29 of the 40 topics have relevance judgements. For these reasons we have abandoned the idea of a training-test split, and performed some training on the test set. Consequently, the results should be taken as indicative only. As a partial response to this problem, we have trained on one measure and evaluated on others, as discussed in section 4.3. Thus although it is likely that we have overfitted to some extent, accidental properties of the measures themselves will have been avoided. Note also that we are tuning only two parameters, which also reduces the danger of overfitting.

4.2 Ranking function and baselines

We use the BM25F (field-weighted extension of BM25) function mentioned above, as used in [5] (this reference also discusses the BM25 and BM25F parameters, which are therefore not further defined here). Essentially, BM25F combines information across fields at the term frequency level, and then combines information across terms. The BM25F parameters k_1 and b are taken as field-independent and fixed at $k_1 = 1.2$ and $b = 0.75$. BM25F also makes use of the average document length; in this context it would seem to make sense to interpret this as the average length of a retrievable unit, which will be much shorter than the average document. However, some preliminary experiments suggest that this does not work well; pending further investigation, we continue to use average document length.

We compare our present runs to the following:

- BM25 without field weights and without inheritance. For this we include the current title in with the section being retrieved – that is, we give the current title a weight of 1, the same as body text. Parent title is not included.
- Our INEX 2005 run (run 1). This was based on training of field weights at document level only. The field structures used in those experiments is slightly different from the present one.

4.3 Performance measures

All evaluation for our experiments is done with the EVALJ package developed for INEX. The following is a brief account of some of the metrics defined for INEX 2005. Further information about the metrics is available from [16].

INEX defines normalized extended Cumulated Gain measures for particular rank positions i (nxCG[i]), and an average up to a rank position, MAnxCG[i]. There is also a measure called effort precision which may be averaged over recall levels (MAep). Also, in order to measure the ability of systems to retrieve specific relevant elements, the relevance judgements include exhaustivity and specificity components. These can be used in different ways, referred to as quantization levels.

In our experiment we have used MAnxCG[50] as the criterion for optimisation, on the grounds that we would like a ranking which is effective wherever the user stops searching. This optimisation is done separately on two quantization levels, quant(strict) and quant(gen). We then evaluate the results with nxCG[10], nxCG[25], nxCG[50], and MAep. All training and evaluation is done with the overlap=off option, which means that overlap is tolerated in the evaluation.

4.4 Tuning

Using the very simple set of fields and inheritance indicated above, and also using the above standard values for the BM25F parameters k_1 and b, we train by a simple grid search on a range of values of each parameter. The two parameters we train are the weight of the 'Current title' field wf(cur) and that of the 'Parent title' field wf(par) (the weight of the 'Body text' field is set to 1). Thus we train over a 2-dimensional grid. The granularity of the grid is significant. We first tune over a relatively coarse grid, to obtain an approximate optimum. We then fine-tune within a smaller range, to obtain a better approximation. The full range is 0–3000 (wf(cur)) and 0–100 (wf(par)).

A 3-dimensional plot of the performance surface over this grid (for the quant(gen) measure) is shown in Figure 2. It can be seen that the surface shows a peak at a very high weight for the current title, and a much lower one for the parent title, but both much higher than body text.

4.5 Testing

Tuning on MAnxCG(50) produced the weightings in Table 2. As indicated above, the current tuning on the strict measure gives a significant weight to the parent title field (although much smaller than the current title field), suggesting that information in the parent title field has something significant to contribute to this task. The parent title weight for the gen measure is much smaller.

The performance of our tuned runs using the various other measures is shown in Table 3. For the strict measures, although we have not quite done as well as our very successful INEX submission on nxCG(25),[6] performance over the

[6] Another City INEX submission did slightly better on nxCG(25) and significantly better on nxCG(50), but even less consistently overall.

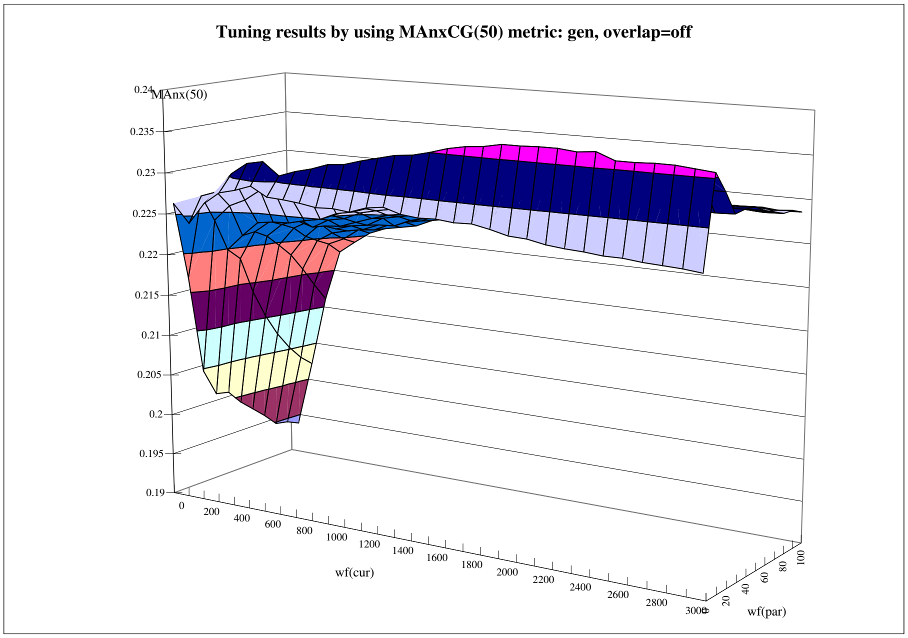

Fig. 2. Performance in the training set over a grid of values of the field weights (quant(gen)).

	wf(cur)	wf(par0
Unweighted	1	0
City-INEX	2356	22
Tuned quant(strict)	1900	56
Tuned quant(gen)	2100	8

Table 2. Field weights for the different runs.

	nxCG(10)	change	nxCG(25)	change	nxCG(50)	change	MAep	change
quant(strict)								
Unweighted	0.0588	–	0.0808	–	0.1338	–	0.0343	–
City-INEX	0.0538	-9%	0.1174	+45%	0.1539	+15%	0.0267	-22%
Tuned	0.0973	+65%	0.1037	+28%	0.1643	+23%	0.0401	+17%
quant(gen)								
Unweighted	0.2126	–	0.2327	–	0.2306	–	0.0624	–
City-INEX	0.1875	-12%	0.1747	-25%	0.1800	-22%	0.0445	-29%
Tuned	0.2321	+9%	0.2407	+3%	0.2382	+3%	0.0610	-2%

Table 3. Test results for different measures. Changes are relative to the baseline.

range of measures is much more stable; we consistently gain substantially on the baseline. For the gen measures, the tuning does not do quite so well, although it still outperforms our INEX submission and does about as well as or slightly better than the baseline.

These results are at least encouraging, although they must be taken with a pinch of salt, because of the overfitting issue.

5 Conclusions

We have presented a way of thinking about retrieval in structured (specifically XML-structured) documents. We propose to treat element retrieval as similar to document or passage retrieval, but to allow elements to inherit information from other elements. This can be done in the following way. First, the elements that are to be made retrievable must be selected. Next, a simple flat field-structure is defined, a single structure applicable to all retrievable elements. Next, a mapping is defined from the document elements to the field structure, for each type of retrievable element. This mapping defines the inheritance: that is, which elements will inherit information from other elements.

We may then use a standard document scoring-and-ranking function which is capable of differential weighting of fields. This is then likely to need tuning on a training set.

A small-scale experiment on INEX 2005 data has shown that this approach has some promise. First, it is possible to make a reasonable mapping of different elements onto a single flat field structure, and second, we have provided some experimental evidence that the inheritance idea is a useful one: in particular, it is useful for lower-level elements to inherit higher-level titles.

It is quite clear, however, that this test is a very partial one, and can be taken only as indicative in a small way. We expect in the future to conduct further and better-designed experiments to validate the ideas. These experiments should include much more detailed examination of the results, as well as being on a larger scale. Many questions remain, for example about how best both to choose inheritable elements and to map them onto a common field structure.

References

1. Callan, J.: Passage-level evidence in document retrieval. In Croft, W.B., van Rijsbergen, C.J., eds.: SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Springer-Verlag (1994) 302–310
2. Zaragoza, H., Craswell, N., Taylor, M., Saria, S., Robertson, S.: Microsoft Cambridge at TREC 2004: Web and HARD track. In Voorhees, E.M., Buckland, L.P., eds.: The Thirteenth Text REtrieval Conference, TREC 2004. NIST Special Publication 500-261, Gaithersburg, MD: NIST (2005) http://trec.nist.gov/pubs/trec13/t13_proceedings.html.

3. Amitay, E., et al.: Juru at TREC 2003 – topic distillation using query-sensitive tuning and cohesiveness filtering. In Voorhees, E.M., Buckland, L.P., eds.: The Twelfth Text REtrieval Conference, TREC 2003. NIST Special Publication 500-255, Gaithersburg, MD: NIST (2004) 276–282 `http://trec.nist.gov/pubs/trec12/t12_proceedings.html`.
4. Wilkinson, R.: Effective retrieval of structured documents. In Croft, W.B., van Rijsbergen, C.J., eds.: SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Springer-Verlag (1994) 311–317
5. Robertson, S., Zaragoza, H., Taylor, M.: Simple BM25 extension to multiple weighted fields. In Evans, D.A., Gravano, L., Hertzog, O., Zhai, C.X., Ronthaler, M., eds.: CIKM 2004: Proceedings of the 13th ACM Conference on Information and Knowledge Management, New York, ACM Press (2004) 42–49
6. Craswell, N., Hawking, D.: Overview of the TREC 2004 web track. In Voorhees, E.M., Buckland, L.P., eds.: The Thirteenth Text REtrieval Conference, TREC 2004. NIST Special Publication 500-261, Gaithersburg, MD: NIST (2005) 89–97 `http://trec.nist.gov/pubs/trec13/t13_proceedings.html`.
7. Arvola, P., Junkkair, M., Kekalainen, J.: Generalized contextualisation method for XML information retrieval. In Herzog, O., Schek, H., Fuhr, N., Chowdhury, A., Teiken, W., eds.: CIKM 2005: Proceedings of the 14th ACM Conference on Information and Knowledge Management, New York, ACM Press (2005) 20–27
8. Ogilvie, P., Callan, J.: Hierarchical language models for XML component retrieval. In Fuhr, N., Malik, S., Lalmas, M., eds.: Advances in XML Information Retrieval: Third International Workshop of the Initative for the Evaluation of XML Retrieval, INEX 2004. LNCS 3493, Heidelberg, Springer (2005) 224–237
9. Sigurbjornsson, B., Kamps, J., de Rijke, M.: An element-based approach to XML retrieval. In Fuhr, N., Malik, S., Lalmas, M., eds.: INEX 2003: Second International Workshop of the Initative for the Evaluation of XML Retrieval, INEX (2004) 19–26
10. Sigurbjornsson, B., Kamps, J., de Rijke, M.: Mixture models, overlap, and structural hints in XML element retrieval. In Fuhr, N., Malik, S., Lalmas, M., eds.: Advances in XML Information Retrieval: Third International Workshop of the Initative for the Evaluation of XML Retrieval, INEX 2004. LNCS 3493, Heidelberg, Springer (2005) 104–109
11. Mass, Y., Mandelbrod, M.: Retrieving the most relevant XML components. In Fuhr, N., Malik, S., Lalmas, M., eds.: INEX 2003: Second International Workshop of the Initative for the Evaluation of XML Retrieval, INEX (2004) 53–58
12. Mass, Y., Mandelbrod, M.: Component ranking and automatic query refinement for XML retrieval. In Fuhr, N., Malik, S., Lalmas, M., eds.: Advances in XML Information Retrieval: Third International Workshop of the Initative for the Evaluation of XML Retrieval, INEX 2004. LNCS 3493, Heidelberg, Springer (2005) 134–140
13. Taylor, M., Zaragoza, H., Craswell, N., Robertson, S.: Optimisation methods for ranking functions with multiple parameters. Submitted for publication (2006)
14. Lu, W., Robertson, S., MacFarlane, A.: Field-weighted XML retrieval based on BM25. INEX 2005; Submitted for publication (2006)
15. INEX: INitiative for the evaluation of XML retrieval. `http://inex.is.informatik.uni-duisburg.de/2005/` (Visited 13 February 2006)
16. Kazai, G., Lalmas, M.: INEX 2005 evaluation metrics. `http://inex.is.informatik.uni-duisburg.de/2005/inex-2005-metricsv6.pdf` (2005) (Visited 22 February 2006).