Why Using Structural Hints in XML Retrieval?

Karen Sauvagnat, Mohand Boughanem, and Claude Chrisment

IRIT - SIG
{sauvagna, bougha, chrisment}@irit.fr
118 route de Narbonne, F-31062 Toulouse Cedex 4

Abstract. When querying XML collections, users cannot always express their need in a precise way. Systems should therefore support vagueness at both the content and structural level of queries. This paper present a relevance-oriented method for ranking XML components. The aim here is to evaluate whether structural hints help to better answer the user needs. We experiment (within the INEX framework) with users needs expressed in a flexible way (i.e with ou without structural hints). Results show that they clearly improve performance, even if they are expressed in an "artificial way". Relevance seems therefore to be closely linked to structure. Moreover, too complex structural hints do not lead to better results.

1 Introduction

The growing number of XML documents leads to the need for appropriate retrieval methods which are able to exploit the specific features of this type of documents. Hierarchical document structure can be used to return specific document components instead of whole documents to users. The main challenge in XML retrieval is therefore to retrieve the most exhaustive and specific information units.

These last years, many methods were proposed in the literature for finding relevant elements. They can be divided into two main sub-groups, depending on the way they consider the content of XML documents. On one hand, the ***data-oriented approaches*** use XML documents to exchange structured data (as for example whole databases). XML documents are seen as highly structured data marked up with XML tags. The database community was the first to propose solutions for the XML retrieval issue, using the data-oriented approach. Unfortunately, the proposed approaches typically expect binary answers to very specific queries. In the Xquery language for example, proposed by the W3C [3], SQL functionalities on tables (tuples collection) are extended to support similar operations on forests (trees collection), as XML documents can be seen as trees. An extension of XQuery with full-text search features is expected [23]. On the other hand, the ***document-oriented approaches*** consider that tags are used to describe the logical structure of documents, which are mainly composed of text. The IR community has adapted traditional IR approaches to address the user information needs in XML collection. Some of these methods are based on the vector space model [8], [16], [9], or on the probabilistic model, either using

H. Legind Larsen et al. (Eds.): FQAS 2006, LNAI 4027, pp.197–209, 2006.

a relevance propagation method [6] or using language models [18], [10]. Such methods have been evaluated since 2002 in the framework of the INEX (Initiative for the Evaluation of XML Retrieval) campaign. This initiative provides an opportunity for participants to evaluate their XML retrieval methods using uniform scoring procedures and a forum for participating organisations to compare their results.

One of the major issue discussed by INEX participants is the users model used when expressing queries. In classical IR, users often have difficulties to express their queries, because they do not know the collection or because they cannot express their information need in a precise way. XML documents emphasize the issue: if users want to express very specific queries, they should not only know the content of the collection, but also how it is structured. This can be a problem when collections contain heterogeneous documents (i.e. documents following different DTD): users cannot express all the possible structure constraints. They should consequently be able to express their need in a flexible way, either by giving some keywords, or by giving some structural hints about what they are looking for. Such hints should not be satisfied strictly, but should be considered as clues on which types of elements are most probably relevant for users. Systems dealing with such needs are therefore necessary [15].

The use of structural hints has already been explored during the INEX 2004 campaign. Results showed that structure was not really useful to improve performance [5]. However, Kamps et al. [11] showed that the use of structure in queries functions as a precision enhancing device, even if it does not lead to improved mean average precision scores. It is however difficult to draw conclusions on INEX 2004 results, since the pool used for the assessments was not only obtained with the content conditions of queries (most of the results in the pool were obtained using structure conditions of queries and bias the pool).

In this paper, we use a relevance-oriented method, based on relevance propagation, to evaluate whether the introduction of structure in queries composed of simple keyword terms (also called Content-only (CO) queries) can improve performance. *In other words, does a relevant structure exist for each information need ? Is relevance closely linked to structure ?*

The rest of the paper is organised as follows. Section 2 presents our baseline model, which uses a relevance propagation method. The INEX 2005 test suite and the associated metrics are described in section 3. At least, section 4 presents experiments we've done on the introduction of structure.

2 Baseline Model

The model we used is based on a generic data model that allows the processing of heterogeneous collection (collections that contain documents following different DTD). We consider that a structured document sd_i is a tree, composed of simple nodes n_{ij}, leaf nodes ln_{ij} and attributes a_{ij}. Leaf nodes ln_{ij} are content-bearer (i.e. they are text nodes) whereas other nodes only give indication on structure.

During query processing, relevance values are assigned to leaf nodes and relevance score of inner nodes are then computed dynamically.

2.1 Evaluation of Leaf Nodes Weights

The first step in query processing is to evaluate the relevance value of leaf nodes ln according to the query. Let $q = t_1, \ldots, t_n$ be a query composed of simple keyword terms (i.e. a CO query). Relevance values are computed using a similarity function $RSV(q, ln)$.

$$RSV_m(q, ln) = \sum_{i=1}^{n} w_i^q * w_i^{ln} \quad (1)$$

Where:

- w_i^q is the weight of term i in query q
- w_i^{ln} is the weight of term i in leaf node ln

According to previous experiments [21], we choose to use the following term weighting scheme, which aims at reflecting the importance of terms in leaf nodes, but also in whole documents:

$$w_i^q = tf_i^q \qquad w_i^{ln} = tf_i^{ln} * idf_i * ief_i \quad (2)$$

Where tf_i^q and tf_i^{ln} are respectively the frequency of term i in query q and leaf node ln, $idf_i = log(|D|/(|d_i| + 1)) + 1$, with $|D|$ the total number of documents in the collection, and $|d_i|$ the number of documents containing i, and ief_i is the inverse element frequency of term i, i.e. $log(|N|/|nf_i| + 1) + 1$, where $|nf_i|$ is the number of leaf nodes containing i and $|N|$ is the total number of leaf nodes in the collection.

Inner nodes relevance values are evaluated using one or more propagation functions, which depend on the searching task. These propagation functions are described in the following sections.

2.2 Relevance Propagation for Content-Only Queries

In our model, each node in the document tree is assigned a relevance value which is function of the relevance values of the leaf nodes it contains. Terms that occur close to the root of a given subtree seem to be more significant for the root element that ones at deeper levels of the subtrees. It seems therefore intuitive that the larger the distance of a node from its ancestor is, the less it contributes to the relevance of its ancestor. This is modeled in our propagation formula by the use of the $dist(n, ln_k)$ parameter, which is the distance between node n and leaf node ln_k in the document tree, i.e. the number of arcs that are necessary to join n and ln_k. Moreover, it is also intuitive that the more relevant leaf nodes a node has, the more relevant it is . We then introduce in the propagation function the $|L_n^r|$ parameter, which is the number of n descendant leaf nodes having a non-zero score. The relevance value r_n of a node n is finally computed according to the following formula:

$$r_n = |L_n^r| \sum_{k=1..N} \alpha^{dist(n,ln_k)-1} * RSV(q, ln_k) \quad (3)$$

where $\alpha \in]0..1]$, ln_k are leaf nodes being descendant of n and N is the total number of leaf nodes being descendant of n.

2.3 Relevance Propagation for Content-Only Queries with Structural Hints

Content Only queries with structural hints (also called CO+S queries) are queries containing structural constraints that should be interpreted as vague conditions. Such constraints can simply be constraints on the type of the returned elements (example 1), or content restrictions on the environment in which the requested element occurs (descendants or ancestors): we talk about hierarchical queries (example 2). Here are some examples of CO queries with structural hints (expressed in the XFIRM query language [20]):

- Example 1: *te: sec[electronic commerce e-commerce]* : user is looking for information about "electronic commerce e-commerce" that can be found in section elements (that are target elements(indicated with the terminal expression *te*))
- Example 2: *//article[business strategies]//te: sec[electronic commerce e-commerce]*: user is also looking for information about "electronic commerce e-commerce" which is probably in section elements of an article about "business strategies"

The evaluation of a CO+S query is carried out as follows :

1. Queries are decomposed into elementary sub-queries ESQ, which are of the form: $ESQ = tg[q]$, where tg is a tag name, i.e. a structure constraint, and $q = t_1, ..., t_n$ is a content constraint composed of simple keywords terms.
2. Relevance values are then evaluated between leaf nodes and the content conditions of elementary sub-queries
3. Relevance values are propagated in the document tree to answer to the structure conditions of elementary sub-queries
4. Original queries are evaluated thanks to upwards and downwards propagation of the relevance weights [20]

In step 3, the relevance value r_n of a node n to an elementary subquery $ESQ = tg[q]$ is computed according the following formula:

$$r_n = \begin{cases} \sum_{ln_k \in L_n} \alpha^{dist(n,ln_k)-1} * RSV(q, ln_k) \ if \ n \ \in construct(tg) \\ 0 \ else \end{cases} \quad (4)$$

where the result of the $construct(tg)$ function is a set composed of nodes having tg as tag name, and $RSV(q, ln_k)$ is evaluated during step 2 with equation 1. The $construct(tg)$ function uses a *Dictionary* Index, which provides for a given tag tg the tags that are considered as equivalent.

For processing CO+S queries, as structural conditions should be considered as vague conditions, we use a dictionary index composed of very extended equivalencies. For example, a section node (*sec*) can be considered as equivalent to both a paragraph (*p*) and a body (*bdy*) node (see Appendix for the dictionary index used on the INEX collection). This index is built manually.More details about CO+S queries processing can be found in [20].

2.4 Discussion

Many relevance propagation methods can be found in the literature [1] [2] [8] [7] [19]. Our approach differs from these previous works on two main points. The first point is that all leaf nodes are indexed, because we think that even the smallest leaf nodes can be relevant or can give information on the relevance of its ancestors. Advantages of such an approach are twofold: first, the index process can be done automatically, without any human intervention and the system will be so able to handle heterogeneous collections automatically; and secondly, even the most specific query concerning the document structure will be processed, since all the document structure is kept.

The second point is that the propagation is made step by step and takes into account the distance that separate nodes in the document tree.

Our aim here is not to present a new propagation method, but to evaluate whether the introduction of structure can improve overall performance of systems.

3 INEX 2005 Evaluation Campaign

3.1 Collection and Topics

We used the well-known INEX framework to evaluate the use of structural hints. The 2005 test collection completes the one used during the last years and is composed of more than 17000 documents with extensive XML markup, extracted from IEEE Computer Society journals published between 1995 and 2004.

Experiments presented here are related to the Content-Only (CO) task and the Content-Only+Structure (CO+S) task of the INEX 2005 campaign. The 2005 CO-CO+S tasks are composed of 29 topics and of the associated relevance judgments. An example of such topic can be found in table 1. We use the *title* part of topic for CO queries, and the *castitle* part for CO+S queries.

Only 19 topics have structural hints. In order to compared CO and CO+S tasks however, the 2 sets of queries we use need to have the same number of topics. We consequently use for each task two sets of queries, respectively called

Table 1. Example of CO-CO+S query

<inex_topic topic_id="202" query_type="CO+S">
<InitialTopicStatement> [...]</InitialTopicStatement>
<title> ontologies case study </title>
<castitle> //article[about(.,ontologies)]//sec[about(.,ontoologies case study)]</castitle>
<description> Case studues in the use of ontologies </description>
<narrative> I'm writing a report on the use of ontnnologies. I'm interested in knowing how ontologies are used to encode knowledge in real world scenarios. I'm particularly interested in knowing what sort of concepts and relations people use in their ontologies [...] </narrative>
</inex_topic>

full-set and *partial-set*. In the full set, all 29 topics are used. For CO+S task, when no *castitle* part is available, we create it in an artificial way, by adding *section* as structural constraint (*section* elements are elements that are the most often returned by systems [4]). In the partial set, only the 19 topics containing a *castitle* part are used. The partial set is used for the official submissions at the INEX 2005 CO+S task, whereas the full set is used for official submissions at the CO task.

Relevance judgments for each query are done by the participants. Two dimensions of relevance are used: exhaustivity (e) and specificity (s). Exhaustivity is measured using a 4-level scale: highly exhaustive (e=2), somewhat exhaustive (e=1), not exhaustive (e=0), too small (e=?). Specificity is measured on a continuous scale with values in [0,1], where s=1 represents a fully specific component (i.e. one that contains only relevant information).

During the assessments, structural conditions were ignored. Judges assessed the elements returned for CO+S queries as whether they satisfied the information need with respect to the content criterion only.

3.2 Metrics

Whatever the metrics used for evaluating systems, the two dimensions of relevance (exhaustivity and specificity) need to be quantised into a single relevance value. Quantisation functions for 2 user standpoints are used:

- a strict quantisation to evaluate whether a given retrieval approach is able of retrieving highly exhaustive and highly specific document components

$$f_{strict}(e,s) = \begin{cases} 1 \; if \; e = 2 \; and \; s = 1 \\ 0 \; otherwise \end{cases} \tag{5}$$

- a generalised quantisation has been used in order to credit document components according to their degree of relevance

$$f_{generalised}(e,s) = e * s \tag{6}$$

Official metrics are based on the extended cumulated gain (XCG) [14] [13]. The XCG metrics are a family of metrics that aim to consider the dependency of XML elements (e.g. overlap and near misses) within the evaluation. The XCG metrics include the user-oriented measures of normalised extended cumulated gain (nXCG) and the system-oriented effort-precision/gain-recall measures (ep/gr). The xCG metric accumulates the relevance scores of retrieved documents along a ranked list. Given a ranked list of document components, xCG, where the element IDs are replaced with their relevance scores, the cumulated gain at rank i, denoted as xCG[i], is computed as the sum of the relevance scores up to that rank:

$$xCG[i] = \sum_{j=1}^{i} xG[j] \tag{7}$$

For example, the ranking $xG_q =< 2, 1, 0, 1, 0, 0 >$ produces the cumulated gain vector of $xCG =< 2, 3, 3, 4, 4, 4 >$.

For each query, an ideal gain vector, xI, can be derived by filling the rank positions with the relevance scores of all documents in the recall-base in decreasing order of their degree of relevance. The corresponding cumulated ideal gain vector is referred to as xCI. By dividing the xCG vectors of the retrieval runs by their corresponding ideal xCI vectors, the normalised xCG ($nxCG$) measure is obtained:

$$nxCG[i] = \frac{xCG[i]}{xCI[i]} \tag{8}$$

For a given rank i, the value of $nxCG[i]$ reflects the relative gain the user accumulated up to that rank, compared to the gain he/she could have attained if the system would have produced the optimum best ranking. For any rank the normalised value of 1 represents ideal performance.

Analogue to the definition of $nxCG$, a precision-oriented XCG measure, effort-precision ep, is defined as:

$$ep(r) = \frac{e_{ideal}}{e_{run}} \tag{9}$$

where e_{ideal} is the rank position at which the cumulated gain of r is reached by the ideal curve and e_{run} is the rank position at which the cumulated gain of r is reached by the system run. A score of 1 reflects ideal performance, where the user need to spend the minimum necessary effort to reach a given level of gain.

Effort-precision, ep, is calculated at arbitrary gain-recall points, where gain-recall is calculated as the cumulated gain value divided by the total achievable cumulated gain:

$$gr[i] = \frac{xCG[i]}{xCI[n]} \tag{10}$$

where n is the total number of relevant documents.

The meaning of effort-precision at a given gain-recall value is the amount of relative effort (where effort is measured in terms of number of visited ranks) that the user is required to spend when scanning a systems result ranking compared to the effort an ideal ranking would take in order to reach a given level of gain relative to the total gain that can be obtained. See [13] for more details.

4 Experiments

We experiment with two search strategies, which correspond to different user needs: find all relevant information in the collection or find only the most relevant information:

- **find all highly exhaustive and specific elements** (*thorough strategy*)**.** The nature of relevance in XML retrieval may imply overlapping elements (i.e. elements that are nested within each others) to be returned by systems. If a child element is relevant, so will be its parent, although to a greater or lesser extent. It is however a challenge to rank these elements appropriately, as systems that rank highly exhaustive and specific elements before less exhaustive and specific ones, will obtain a higher effectiveness performance.

– **find the most exhaustive and specific element in a path** (*focussed strategy*)**.** No overlapping elements are allowed: for a given document, only elements that are not nested within each others can be returned.

It seems to us important to evaluate the use of structural hints on these two strategies, since they could lead to contradictory behaviors.

To answer to the thorough strategy, weighted sub-trees (equation 3 and 4) are simply ordered and returned by the system. In order to remove nodes overlap (focussed retrieval), we use the following strategy: for each relevant path, we keep the most relevant node in the path. The results set is then parsed again, to eliminate any possible overlap among results components.

4.1 Experiments with a Thorough Strategy

According to previous experiments, we use $\alpha = 0.1$ in equation 3 and $\alpha = 0.5$ in equation 4 (these values are optimal for the considering sub-tasks).

Figures 1 and 2 show the evolution of the nxCG metric, for both the full set and partial set of queries. Table 2 shows the results for the ep/gr-MAP metric.

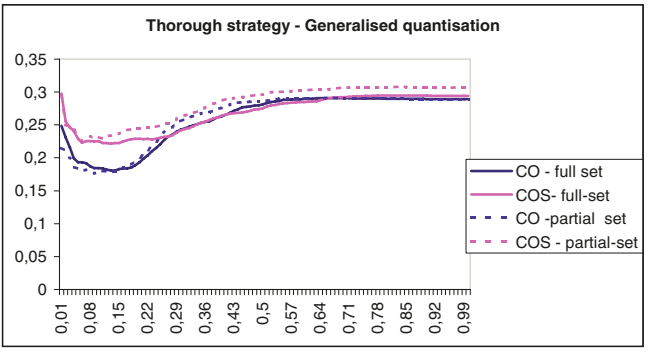

Fig. 1. nXCG evolution - Thorough strategy - Generalised quantisation

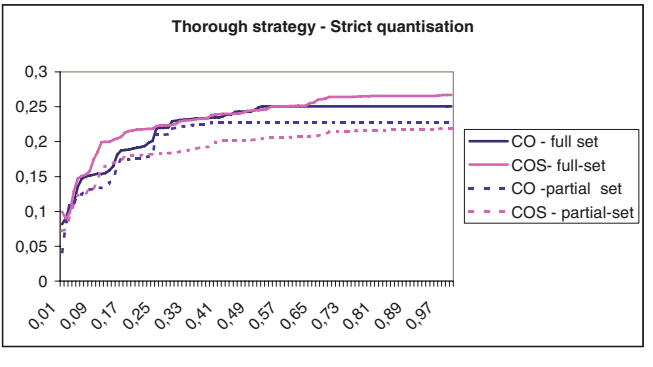

Fig. 2. nXCG evolution - Thorough strategy - Strict quantisation

Table 2. Comparison of CO/CO+S queries with MAP metric, Thorough strategy

		Generalised	Strict
Full-set	CO	0,0535	0,0234
	CO+S	0,0690	0,0240
	Gain	+29%	+3%
Partial-set	CO	0,0517	0,0158
	CO+S	0,0749	0,0163
	Gain	+45%	+3%

Results with CO+S queries are better for low levels of recall, and are comparable to those obtained for the CO strategy at other recall levels. However, if we considered average results (see table 2), the CO+S strategy is clearly preferable to the CO strategy.

If we now compare results for the full set and the partial set, we see that if we consider the generalised quantisation function, results are better with the partial set, whereas it is not the case when considering the strict quantisation function.

4.2 Experiments with a Focussed Strategy

For the focussed retrieval strategy, according to previous experiments, we use $\alpha = 0.1$ in equation 3 and $\alpha = 0.2$ in equation 4 (these values are optimal for the considering sub-tasks).

Figures 3 and 4 show the evolution of the nXCG metric, for both the full set and partial set of queries. Table 3 shows the results for the ep/gr-MAP metric.

At low levels of recall, results for the thorough strategy are comparable to results for the focussed strategy: CO+S queries allows to obtain better results than simple CO queries. However, we can observe reverse results for other recall levels. If we now consider average results, results obtained with CO+S queries outperformed again results obtained with CO queries.

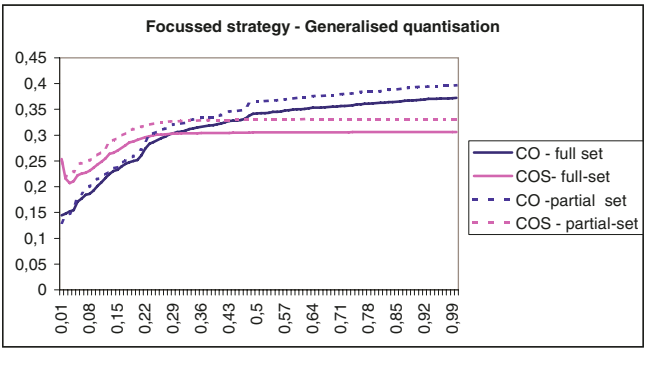

Fig. 3. nXCG evolution - Focussed strategy - Generalised quantisation

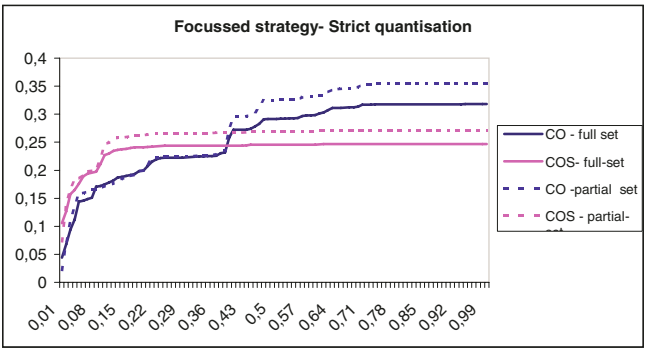

Fig. 4. nXCG evolution - Focussed strategy - Strict quantisation

Table 3. Comparison of CO/CO+S queries with MAP metric, Focussed strategy

		Generalised	Strict
Full set	CO	0,0538	0,0195
	CO+S	0,0782	0,0372
	Gain	+45%	+91%
Partial set	CO	0,0554	0,0167
	CO+S	0,0877	0,0353
	Gain	+58%	+111%

When comparing partial and full sets, we notice that results are in a general manner better for the partial set. This can be explained by the fact that the partial set only contains queries having a structural constraint defined by the user, whereas this constraints was artificially added for some queries in the full set. This is not surprising, since the user need is more clearly defined in queries of the partial set.

4.3 Discussion

For both retrieval strategies we see that results are significantly better when structural hints are used, for almost all recall levels and when considering mean average precision. This extends conclusions drawn in [11], in which authors showed that structured queries function as a precision enhancing device, i.e. are useful for promoting the precision in initially retrieved elements, but are not useful on mean average precision.

Improvements are more significant on the focussed strategy. This is not really surprising, since in CO+S queries the users needs are more clearly defined: they help finding the most specific element in a path.

Let us now consider results in depth, i.e. results for each query[1]. We see that CO+S queries improve results even for queries with an "artificial" structure con-

[1] Due to space limitation, results are not presented here.

straint, i.e. CO queries that do not have structural hints and for which we add the structural constraint (10 queries over 29). Queries for which no improvement is observed are queries having a too complex structure condition (i.e. having hierarchical conditions). This seems to show that only simple structural constraints are useful for improving overall performance, even if these constraints are artificial. In other words, it shows that relevant information seems to be located in some particular element types and that relevance is closely linked to structure. At last, we have to notice the relative high precision of our runs compared to INEX 2005 official submissions [5]. Most of our runs would have been ranked in the top ten for both quantisation functions. Best results are obtained for the CO-Focussed strategy: we would have been ranked first for generalised quantisation on nxCG[10], nXCG[25] and nXCG[50] metrics (official metrics at INEX 2005). We respectively obtain 0.2607, 0.2397 and 0.224, whereas best results were respectively 0.2181, 0.1918 and 0.1817, and were obtained by the University of Amsterdam with a language model-based method [22] and by IBM Haifa Research Lab [17] using the vector space model. We would also be in the top 5 for strict quantisation.

5 Conclusion

Content-oriented retrieval is one of the most challenging issue in XML retrieval, because queries do not contain indications on which type of elements should be returned to the user. In this paper, we present some experiments on the use of structural hints in queries (which allows the user to express queries in a more flexible way).

We can draw several conclusions from these experiments. First, the use of structural hints improve in a very significant way the system performance, both in average and at low levels of recall. This means that users have almost always an implicit structural need and that relevance is closely linked to structure. Moreover, structural hints have a higher impact for a focussed retrieval strategy than for a thorough retrieval strategy: they help the system to focus on the user information need. Third, the introduction of "artificial" structural hints also improve results: our system is consequently able to process queries in a flexible way. At last, no improvement is observed for structural queries having too complex structural conditions (i.e. hierarchical conditions): structure should consequently be only used as an indication of what type of elements should be retrieved.

In the future, we will apply these results for doing structured relevance feedback: as the use of structural hints improve significantly results, we will try to add them in CO queries when doing relevance feedback. The study of the type of relevant elements will allow us to better define the implicit need of the user and to express new queries with more relevant structural hints (than the artificial hints used in this article when none were available). Some preliminary work can be found in [12].

References

1. M. Abolhassani and N. Fuhr. Applying the divergence from randomness approach for content-only search in XML documents. In *Proceedings of ECIR 2004, Sunderland*, pages 409–419, 2004.
2. V. N. Anh and A. Moffat. Compression and an IR approach to XML retrieval. In *Proceedings of INEX 2002 Workshop, Dagstuhl, Germany*, 2002.
3. M. Fernandez, A. Malhotra, J. Marsh, M. Nagy, and N. Walsh. XQuery 1.0 and XPath 2.0 data model. Technical report, World Wide Web Consortium (W3C), W3C Working Draft, may 2003.
4. N. Fuhr, M. Lalmas, and S. Malik. INEX 2003 workshop proceedings, 2003.
5. N. Fuhr, M. Lalmas, S. Malik, and G. Kazai. INEX 2005 workshop pre-proceedings, 2005.
6. N. Fuhr, S. Malik, and M. Lalmas. Overview of the initiative for the evaluation of XML retrieval (INEX) 2003. In *Proceedings of INEX 2003 Workshop, Dagstuhl, Germany*, December 2003.
7. N. Gövert, M. Abolhassani, N. Fuhr, and K. Grossjohann. Content-oriented XML retrieval with hyrex. In *Proceedings of the first INEX Workshop, Dagstuhl, Germany*, 2002.
8. T. Grabs and H.-J. Scheck. Flexible information retrieval from xml with PowerDB XML. In *Proceedings in the First Annual Workshop for the Evaluation of XML Retrieval (INEX)*, pages 26–32, December 2002.
9. V. Kakade and P. Raghavan. Encoding XML in vector spaces. In *Proceedings of ECIR 2005, Saint Jacques de COmpostelle, Spain*, 2005.
10. J. Kamps, M. de Rijke, and B. Sigurbjornsson. Length normalization in XML retrieval. In *Proceedings of SIGIR 2004, Sheffield, England*, pages 80–87, 2004.
11. J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjornsson. Structured queries in XML retrieval. In *Proceedings of CIKM 2005, Bremen, Germany*, 2005.
12. M. B. Karen Sauvagnat, Lobna Hlaoua. Xfirm at inex 2005: ad-hoc and relevance feedback tracks. In *INEX 2005 Workshop pre-proceedings, Dagstuhl, Germany*, november 2005.
13. G. Kazai and M. Lalmas. Inex 2005 evaluation metrics. In *Pre-proceedings of INEX 2005, Dagstuhl, Allemagne*, November 2005.
14. G. Kazai, M. Lalmas, and A. P. de Vries. The overlap problem in content-oriented XML retrieval evaluation. In *Proceedings of SIGIR 2004, Sheffield, England*, pages 72–79, July 2004.
15. M. Lalmas and T. Rölleke. Modelling vague content and structure querying in xml retrieval with a probabilistic object-relational framework. In *Proceedings of FQAS 2004, Lyon, France*, june 2004.
16. Y. Mass and M. Mandelbrod. Component ranking and automatic query refinement for XML retrieval. In *Proceedings of INEX 2004*, pages 134–140, 2004.
17. Y. Mass and M. Mandelbrod. Experimenting various user models for xml retrieval. In *Pre-Proceedings of INEX 2005, Dagstuhl, Germany*, 2005.
18. P. Ogilvie and J. Callan. Using language models for flat text queries in XML retrieval. In *Proceedings of INEX 2003 Workshop, Dagstuhl, Germany*, pages 12–18, December 2003.
19. T. Roelleke, M. Lalmas, G. Kazai, J. Ruthven, and S. Quicker. The accessibility dimension for structured document retrieval. In *Proceedings of ECIR 2002*, 2002.
20. K. Sauvagnat, M. Boughanem, and C. Chrisment. Answering content-and-structure-based queries on XML documents using relevance propagation . In *Information Systems - Special Issue SPIRE 2004* . Elsevier, 2006.

21. K. Sauvagnat, L. Hlaoua, and M. Boughanem. XML retrieval: what about using contextual relevance? In *ACM Symposium on Applied Ccomputing (SAC) - IAR (Information Access and Retrieval) , Dijon*, April 2006.
22. B. Sigurbjörnsson, J. Kamps, and M. de Rijke. The university of Amsterdam at INEX 2005: Adhoc track. In *Pre-Proceedings of INEX 2005 workshop, Dagstuhl, Germany*, november 2005.
23. W3C. XQuery and XPath full-text use cases. Technical report, World Wide Web Consortium (W3C), W3C working draft, february 2003.

A Dictionary Index

The first tag in each line is considered to be equivalent to the following tags.

Table 4. Dictionary index

p,ilrj,ip1,ip2,ip3,ip4,ip5,item-none,p1,p2,p3,sec,ss1,ss2,ss3,abs,bdy,article
ip1,p,ilrj,ip2,ip3,ip4,ip5,item-none,p1,p2,p3,sec,ss1,ss2,ss3,abs,bdy,article
sec,ss1,ss2,ss3,p,ilrj,ip1,ip2,ip3,ip4,ip5,item-none,p1,p2,p3,abs,bdy,article
dl,l1,l2,l3,l4,l5,l6,l7,l8,l9,la,lb,lc,ld,le,list,numeric-list,numeric-rbrace,bullet-list
h,h1,h1a,h2,h2a,h3,h4
abs,fm,article,p,ilrj,ip1,ip2,ip3,ip4,ip5,item-none,p1,p2,p3,sec,ss1,ss2,ss3,bdy
fm,article
bdy,article
bb,bm,bib,bibl,article
bib,bibl,article,bb,bm
atl,tig,st
st,atl,tig
snm,au
au,snm
vt,bm,article
fgc,fig,p,ilrj,ip1,ip2,ip3,ip4,ip5,item-none,p1,p2,p3,sec,ss1,ss2,ss3,abs,bdy,article
article,fm,bm,bdy,abs,p,ilrj,ip1,ip2,ip3,ip4,ip5,item-none,p1,p2,p3,sec,ss1,ss2,ss3