CWI at INEX2002

Johan List and Arjen P. de Vries

Center for Mathematics and Computer Science (CWI)
P.O.Box 94079, 1090GB Amsterdam, The Netherlands
{j.a.list, a.p.de.vries}@cwi.nl

Abstract

This paper describes our participation in INEX (the Initiative for the Evaluation of XML Retrieval) and discusses several aspects of our XML retrieval system: the retrieval model, the document indexing and manipulation scheme and our preliminary evaluation results of the submitted three runs.

In our system, we have used a probabilistic retrieval model where we map *dimensions of relevance* to (possibly structural) properties of documents and use these dimensions of relevance for retrieval purposes. The study concentrates on *coverage*, a measure reflecting how focused the component is on the given topic while considering that it should serve as an informative unit to be retrieved by itself. We also discuss an efficient and database-independent indexing scheme for XML documents, based on text regions and discuss region operators for selection and manipulation of XML document regions.

1 Introduction

This paper describes our participation in INEX (the Initiative for the Evaluation of XML Retrieval). We participated with our XML retrieval system, built on top of a research database kernel, MonetDB.

The primary goals for participation in the XML Retrieval Initiative were 1) to gain experience in information retrieval of documents possessing various degrees of semantic structure, 2) to look for possibilities to introduce structural properties of documents into probabilistic retrieval models and 3) to examine whether the use of structure information can improve retrieval performance.

The construction of any information retrieval system (and as such an XML retrieval system) can be thought of to address three components: document representation, the retrieval model and query formulation. Document representation defines the logical and physical representation of documents in a retrieval system. 'Flat' documents are mostly represented with techniques such as inverted lists, but in the case of structured documents we need to represent the structural aspects of documents as well.

The use of structure plays a possible role as well in addressing the second component, the definition of the retrieval model. The basis for our model is a probabilistic retrieval model, the statistical language model developed by Hiemstra [11].

The third component deals with query formulation. The extra dimension of structure in XML documents plays a role here as well: how is structural information integrated in the query possibilities and in what sense do query formulation possibilities depend on user knowledge of the structure(s) present in the collection?

The main contributions of this paper are twofold. We present an efficient and database-independent indexing scheme for XML documents based on *XML document regions*. We then describe a probabilistic retrieval model where we map (structural) properties of documents to dimensions of relevance and use these dimensions of relevance for retrieval purposes. The study concentrates on *coverage*, a measure describing how much of the document component is relevant to the topic of request while also considering that it should serve as an informative unit to be retrieved by itself.

2 The Retrieval Model

Research in the user modeling and concept of relevance areas (see e.g. [3, 4, 5, 2]) suggests that relevance is a multidimensional concept of which *topicality* (i.e. content-based relevance) is only a single one. Mizarro [16] names other, possible non-topical dimensions *abstract characteristics of documents* constructed

independently from the particulars of the database or collection at hand. In other words: other, non-topical dimensions are constructed independently from the language models present in the documents of a collection, suggesting orthogonality between the topicality dimension and any additional dimensions. Examples of other, non-topical dimensions include comprehensibility (style or difficulty of the text) and quantity (how much information does the user want; measured by e.g. the size of documents and the number of documents returned to the user).

Additional dimensions of relevance become more important for structured document retrieval. Retrieval units can vary in granularity and hence vary in the amount of information offered to the user. This varying amount of information highly likely causes a user to judge the relevance of document components on more properties besides topicality alone.

We model dimensions of relevance with a set of independent probabilities (assumed independent given a document instantiation) in a probabilistic retrieval model. The research question is whether we can effectively map dimensions of relevance to document properties (structural or otherwise) that in turn can be represented by (probabilistic) entities in the retrieval model. The results reported here investigate a combination of quantity and topicality, visualized in Figure 1; aiming to capture the notion of coverage used in the evaluation.

2.1 A Motivating Example

In INEX, retrieval results are judged on two aspects: relevance and coverage. Relevance is aimed to reflect how exhaustively a topic is discussed within a document component; coverage reflects how focused the component is on the given topic, considering that it also serves as an informative unit. The INEX relevance assessment guide [1] defines relevance and coverage on a four degree scale: relevance levels of 0 (irrelevant), 1 (marginally relevant), 2 (fairly relevant), and 3 (highly relevant), and coverage of N (no coverage), E (exact), S (too small) and L (too large). With the combination of these measures it is possible to identify document components that satisfy both topicality and quantity.

Consider the example document in Figure 2. Say that the system that estimates topicality identifies one relevant subsection in the first section and one relevant subsection in the second section. The open question is then whether to return the two separate subsections, or the separate sections or single body containing these as well as the remaining (possibly irrelevant) subsections (i.e. what is the retrieval unit?). The additional context

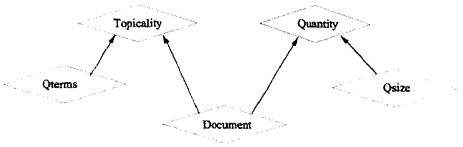


Figure 1: Encoding of additional relevance dimensions. Note that *Qterms* and *Qsize* denote information given by the query (query terms and preferred component size).

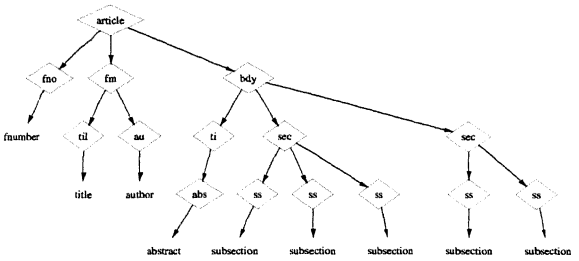


Figure 2: Running example XML syntax tree.

provided by the full sections or body may be more desirable for a user than the individual two subsections in isolation.

We approach the problem of chosing the best acceptable retrieval unit by optimizing on both topicality and size of document components:

- the shorter the document component, the more likely it will not contain enough information to fulfill the information need (the component may be less exhaustive, e.g. relevance level 1 or 2, and 'too small', coverage grade S);
- the longer the document component, the more likely that distilling the topically relevant information will take substantial more reader effort (the component may be more exhaustive, e.g. relevance level 3, but 'too large', grade L on the INEX coverage scale).

We therefore rank the documents in a collection against a combination of topicality and quantity (where the user uses document component size as a representation of quantity). In probabilistic terms, we calculate the probability of complete relevance of a document component, given its probability of relevance on both the topicality and the quantity dimensions.[1]

[1] Here, 'complete relevance' covers all dimensions of relevance, unlike the 'exhaustiveness only' notion of relevance used in INEX.

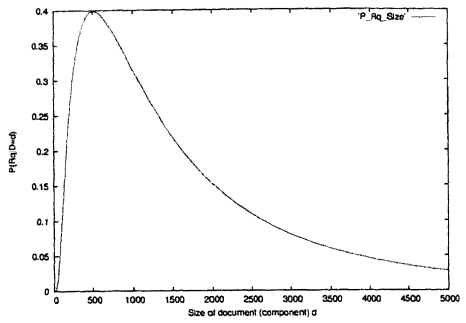


Figure 3: The log-normal distribution used for modeling the quantity dimension

2.2 Modeling Relevance Dimensions

The model in Figure 1 leads to the following. When $P(R_t|D_d)$ is the probability of topical relevance given document d and $P(R_q|D_d)$ is the probability of quantity relevance given document d, then we can calculate a joint probability of 'complete' relevance or user satisfaction as:

$$P(D_d,R_t,R_q,Q_{terms},Q_{size}) = P(R_t|D_d,Q_{terms})P(R_q|D_d,Q_{size})P(D_d)$$

Looking at the motivating example in subsection 2.1 and especially the user reasoning for modeling the quantity dimension, we decided to use a log-normal distribution as in Figure 3. The steep slope at the start reflects the pruning we want to model for (extremely) short document components since short components are unlikely to be good retrieval units. The long tail reflects that we do want to prune out very long document components, but not as rigorously as extremely short ones. Long components might be useful, even while taking more reader effort to distill the relevant information.

We also need a modeling parameter for the distribution itself. We have chosen component size, but other possibilities include:

- the depth of the document component in the tree structure, where we want to penalize components present deep in the trees (generally small components and too specific) or components present high in the trees (generally large components and too broad);
- the number of children of a document component. A short document component containing a large amount of children highly likely contains a diversified mix of information and a could be less desirable for a user than a more homogeneous component.

2.3 Modeling Topicality

The model used for describing topicality of documents is a probabilistic model, the statistical language model described by Hiemstra [11]. The main idea of this model is to extract and to compare document and query models and determine the probability that the document generated the query. In other words, the statistical language model extracts linguistic information and is suited for modeling of the topicality dimension of the information need.

In deriving document models for all of the documents in the collection, we regarded every subtree present in the collection as a separate document. The probability of topical relevance $P(R_t|D_d,Q_{terms})$ where Q_{terms} consists of the set of query terms $\{T_1,\cdots,T_n\}$ is calculated with:

$$\begin{aligned} P(R_t|D_d,Q_{terms}) &= P(R_t|D_d,T_1,\cdots,T_n) \\ &= P(D_d)\prod_{i=1}^{n} P(I_i)P(T_i|I_i,D_d) \end{aligned}$$

where $P(I_i)$ is the probability that a term is important (the event I has a sample space of $\{0,1\}$).

We follow the reasoning of Hiemstra [11] to relate the model to a weighting scheme (tf.idf-based). After some manipulation of the model we get:

$$P(D_d,T_1,\cdots,T_n) \propto P(D_d)\prod_{i=1}^{n}(1+\frac{\lambda P(T_i|D_d)}{(1-\lambda)P(T_i)})$$

As estimators for $P(D_d)$, $P(T_i|D_d)$ and $P(T_i)$ we used:

$$P(D_d) = \frac{1}{n} \tag{1}$$

$$P(T_i|D_d) = \frac{tf_{i,d}}{\sum_i tf_{i,d}} \tag{2}$$

where n is the number of documents, $tf_{i,d}$ is the term frequency of term i in document d and $\sum_i tf(i,d)$ is the length of document d.

For $P(T_i)$ we used:

$$P(T_i) = \frac{df_i}{\sum_i df_i} \tag{3}$$

where df_i is the document frequency of term i.

Filling in the likelihood estimators gives us the following model for topicality (with a constant λ for all terms):

$$P(R_t|D_d, Q_{terms}) = P(R_t|D_d, T_1, \cdots, T_n)$$
$$\propto \sum_{i=1}^{n} \log(1 + \frac{\lambda}{1-\lambda} \frac{tf_{i,d}}{\sum_i tf_{i,d}} \frac{\sum_i df_i}{df_i})$$

We used a very simple query model resulting in query term weights represented with $tf_{i,q}$, the term frequency of term i in query q.

3 XML Document Indexing and Manipulation

3.1 Document Model

Generally, XML documents are represented as rooted (syntax) trees and indexing schemes focus on the storage of the edges present in the syntax tree, combined with storage of the text present. One of these approaches is described by Schmidt [17], which we used as a starting point for our own indexing scheme. In Schmidt's approach, each unique path is stored in a set of binary relations where each binary relation represents an edge present in the path. Furthermore, multiple instances of the same path (even if they are present in different syntax trees) are stored in the identical set of relations. The system also maintains a schema of the paths present and their corresponding relations: the *path summary*.

The advantage of Schmidt's approach is that the execution of pure path queries can be performed efficiently; selecting the nodes belonging to a certain path prevents a forced scan of (large) amounts of irrelevant data, requiring only a fast lookup in the path summary to get to the relation required. The disadvantage is that the generation of the transitive closure of a node is an expensive operation. In database terms: the transitive closure is the union of the separate paths present in the component. The reconstruction of each path is performed with join operations, where the number of join operations depends on the number of steps present in the path.

Since we need fast access to the component text for determining statistics, we pursued another approach. Instead of seeing an XML document instance as a syntax tree, we see each XML document instance as a linearized string or a set of *tokens* (including the document text itself). Each component is then a text region or a contiguous subset of the entire linearized string. The linearized string of the example document in Figure 2 is shown below:

```
<article><fno>fno</fno><fm><til>Til</til>
<au>Author</au></fm><bdy><abs>Abs</abs>
<sec>Sec</sec></bdy></article>
```

A text region a can be identified by its starting point s_a and ending point e_a within the entire linearized string. Figure 4a visualizes the start point and end point numbering for the example XML document and we can see, for example, that the *bdy*-region can be identified with the closed interval [12..37]. We have visualized the complete region set of the example XML document in Figure 4b. The index terms present in the content text of the XML document are encoded as text regions with a length of 1 position and stored in a separate relation, the word index $\mathcal{W}$.

For completeness, we give the formal definition for an XML data region as used in our system below.

Definition 3.1. An XML data region $\mathbf{r}$ is defined as a five-tuple $(o_r, s_r, e_r, t_r, p_r)$, where:

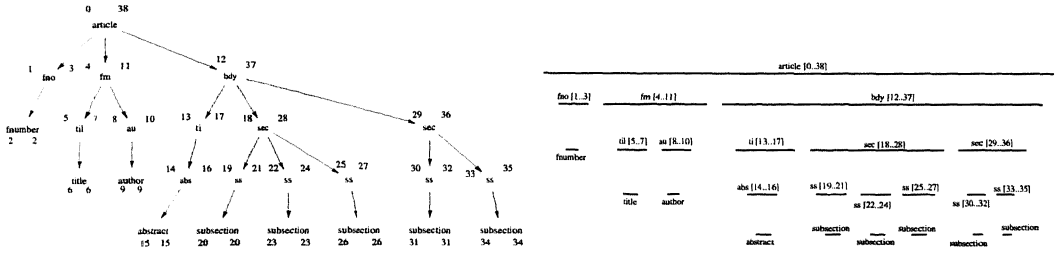
- $\mathbf{o_r} \in$ **oid** denotes a unique node identifier for region r;
- $\mathbf{s_r}$ and $\mathbf{e_r}$ represent the start and end positions of the text region $\mathbf{r}$ respectively;
- $\mathbf{t_r} \in$ **string** is the node name of region $\mathbf{r}$;
- $\mathbf{p_r} \in$ **oid** is the identifier of the parent region of region $\mathbf{r}$.

We also define the node index $\mathcal{N}$ as the projection of o_r over the set of all indexed regions.

3.2 Document Manipulation

The linearized string view enabled us to use theory and practice from the area of text region algebras [7, 8, 9, 13, 15, 14] for selection and manipulation of (sets of) text regions. Table 1 summarizes the operators in our system. The *containment* operation $a \supset b$ determines if the region a contains some other region b, *length* gives the length of a region including markup and *textlength* gives the length of a region excluding markup. Analogous join operators are defined on region sets (A and B).

The use of text regions shows us efficient implementation possibilities. Generating the transitive closure of a region a requires a contains-operation, a selection on the word index $\mathcal{W}$ with lower and upper bounds s_a and e_a. Generating the original XML structure of a (sub-) document d encompasses:



(a) Start point and endpoint assignment

(b) Region representation

Figure 4: Region indexing of XML documents

Table 1: Region and region set operators (the set operators are given in comprehension syntax [6]). Note that s_r and e_r denote the starting and ending positions of region r.

Operator	Definition
$a \supseteq b$	$true \iff s_b \geq s_a \wedge e_b \leq e_a$
$a \supset b$	$true \iff s_b > s_a \wedge e_b < e_a$
$length(a)$	$e_a - s_a + 1$
$textlength(a)$	$\lvert\{a\} \bowtie_{\supset} \mathcal{W}\rvert$
$A \bowtie_{\supseteq} B$	$\{(o_a, o_b) \mid a \leftarrow A,\ b \leftarrow B,\ a \supseteq b\}$
$A \bowtie_{\supset} B$	$\{(o_a, o_b) \mid a \leftarrow A,\ b \leftarrow B,\ a \supset b\}$
$length(A)$	$\{(o_a, length(a)) \mid a \leftarrow A\}$
$textlength(A)$	$\{(o_a, textlength(a)) \mid a \leftarrow A\}$

- a containment operation on the node index $\mathcal{N}$ to retrieve all descendant nodes of d: $desc := \{d\} \bowtie_{\supseteq} \mathcal{N}$. The containment is non-proper since we want the root element d in the set as well;
- a (proper) containment operation on the word index $\mathcal{W}$ to retrieve all context text: $text := \{d\} \bowtie_{\supset} \mathcal{W}$;
- a union of *desc* and *text*, followed by sorting and some string manipulation for finalization.

Note that the approach outlined in this subsection is similar to the preordering and post-ordering approach for acceleration of XPath queries, proposed by Grust [10] (we consider Grust's approach a specific instance of general text region algebras, as is ours).

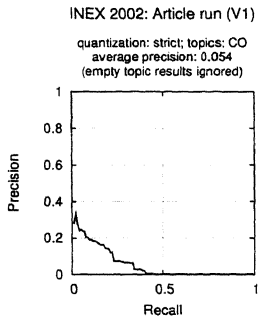
Table 2: Experimentation scenarios

Scenario	Retr. Unit	Dimension(s)
V_1	$\{tr('article')\}$	*topicality*
V_2	$\{tr('*')\}$	*topicality*
V_3	$\{tr('*')\}$	*top., quant.(500)*
V_4	$\{tr('*')\}$	*top., quant.(2516)*
V_5	$\{tr('*')\}$	*top., quant.(5106)*

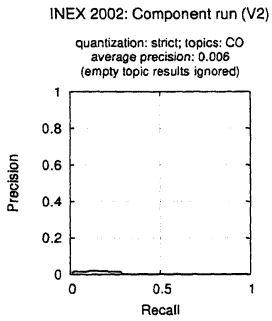
4 Experiments

We designed three experimentation scenarios. The first scenario represents the baseline scenario of 'flat-document' retrieval, i.e. retrieval of documents which possess no structure. After examination of the document collection, we decided to perform retrieval of article-components. The second scenario regarded all subtrees or transitive closures in the collection as separate documents. For the third scenario we re-used the result sets of the second run and used a log-normal distribution to model the quantity dimension. To penalize the retrieval of extremely long document components (this in contrast with the language model that assigns a higher probability to longer documents), as well as extremely short document components, we set the mean at 500 (representing a user with a preference for components of 500 words). We summarized our experimentation scenarios in Table 2. Also note that we focused on content-only queries only (i.e. we used the same approach for content-and-structure queries).

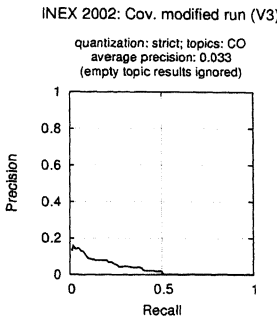
The official recall-precision graphs of our three submitted runs are presented in Figures 5a through 5f. The recall-precision graphs are constructed after mapping relevance/coverage combinations to a binary scale. The



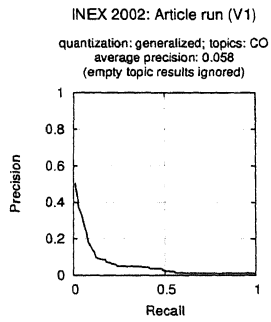
(a) Scenario 1 (Article retrieval)



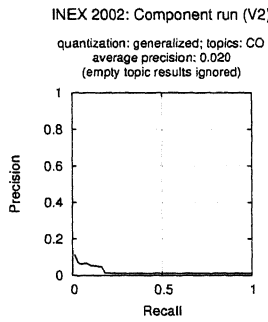
(b) Scenario 2 (Component retrieval)



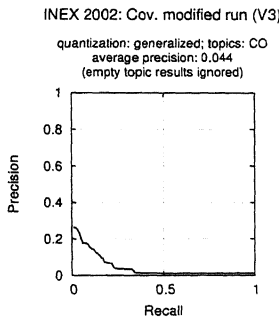
(c) Scenario 3 (Cov. modified component retrieval)



(d) Scenario 1 (Article retrieval)

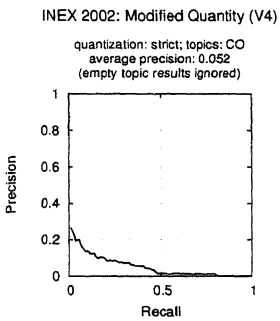


(e) Scenario 2 (Component retrieval)

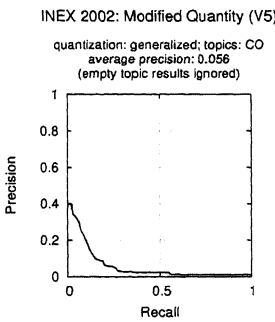


(f) Scenario 3 (Cov. modified component retrieval)

Figure 5: Recall - precision graphs for our experimentation scenarios, CO-topics only (first row: strict evaluation, second row: generalized evaluation).



(a) Additional run V_4 (strict), with preferred component size set at 2516 words



(b) Additional run V_5 (generalized), with preferred component size set at 5106 words

Figure 6: Recall-precision figures for additional runs 4 and 5.

mapping function for strict evaluation is:

$$f_{strict}(r,c) = \begin{cases} 1 & \text{if 3E} \\ 0 & \text{otherwise} \end{cases}$$

The mapping function for generalized evaluation is:

$$f_{generalized}(r,c) = \begin{cases} 1.0 & \text{if 3E} \\ 0.75 & \text{if 3\{L,S\}, 2E} \\ 0.50 & \text{if 1E, 2L, 2S} \\ 0.25 & \text{if 1S, 1L} \\ 0.00 & \text{if 0N} \end{cases}$$

4.1 An Informal Analysis

A more detailed analysis of the evaluation results for all three runs showed us two observations that triggered our curiosity. The first observation was that for many topics, far more relevant components exist than the result set size could fit. Traditional retrieval collections constructed in the Cranfield tradition contain a small amount of relevant documents in the collection (at least, the amount of relevant documents per query is much smaller than the result set size). This small amount of relevant documents enables a 'perfect' retrieval system to retrieve all relevant documents in the result set, which in turn enables the calculation of system (and run) comparable recall-precision graphs.

However, with a large discrepancy between number of relevant documents and the result set size, higher percentages of recall could never be reached, causing meaningless recall-precision curves. To illustrate this effect further, consider the following example. Let us assume we have a query that has 1000 relevant documents in the collection. The result set size is set at 100 documents. When we determine a precision-recall graph for this query, we will see that after 0.1 recall we get precision values which say nothing meaningful about the performance of a system. Even if all results in the result set are relevant (we will reach maximum precision at 0.1 recall), the precision values at higher levels of recall will always decrease, simply because no more documents have been retrieved (resulting in an average precision of 33% instead of 100%).

For fair evaluation, we can follow two possible paths. Firstly, we can use a measure that is invariant with regard to the difference between 1) the number of relevant documents in the collection (for a given topic) and 2) the result set size. A possibility would be to use precision at various document cutoff levels, instead of precision at various levels of recall [12].

The second observation we made was the observation that, even with the strict evaluation that is most demanding coverage-wise, the article run (Figure 5a) still outperformed all other runs. We had expected that many article components would have been judged as too large. Examination of the judgements for the assessed CO-topics only[2] showed us the results in Tables 3 and 4. Note that the probability in the fourth column is not the probability of a node type being relevant *for all topics*, but the probability of a node type being relevant *for one of the assessed 25 CO-topics*. Both tables show that article-components have a much higher probability of being relevant for one of the CO-topics, when we would draw document components randomly from the collection. Knowing this, it is not surprising the article run performs very well.

We make one last remark regarding our second run, where each component was regarded as a document. The result sets of our second run were saturated with short document components. Looking at the language model used for estimating topical relevance, the cause of this saturation is clear: (query) terms occurring in short components will receive a higher weight than (query) terms occurring in longer components, result-

Table 3: Top 5 of node types present in the judgements for the assessed 25 CO-topics only (strict evaluation function). The '*' denotes the any-element type.

Node type	# relevant	# in collection	P(D)
p	371	762.223	0.0004
article	**308**	**12.107**	**0.025**
sec	273	69.735	0.0039
ss1	111	61.492	0.0018
bdy	90	12.107	0.0074
*	**1360**	**8239997**	**0.0001**

Table 4: Top 5 of node types present in the judgements for the assessed 25 CO-topics only (generalized evaluation function). The '*' denotes the any-element type.

Node type	# relevant	# in collection	P(D)
p	4198	762223	0.005
sec	2781	69735	0.039
article	**2606**	**12107**	**0.21**
bdy	1555	12107	0.12
ss1	1096	61492	0.017
*	**18686**	**8239997**	**0.002**

[2] At the time of writing this paper, 25 CO-topics had been assessed.

ing in higher overall rankings for short components. To remove this bias for short components, additional normalization will be necessary.

4.2 Preferred Component Length

In order to see whether our subjective guess of 500 words for acceptable document components was valid, we calculated the average length of relevant components (relevant according to the strict and generalized evaluation functions): 2516 terms (strict) and 5106 terms (generalized). We used these two means for updating the log-normal in two additional runs V_4 and V_5. The recall-precision graphs of these two additional runs are shown in Figures 6a and 6b, which also show that using the new averages does improve retrieval performance, but not radically. In short, using just document component length seems too naive for estimation of component coverage.

5 Conclusions and Future Work

Our participation in INEX can be summed up as an exercise in applying current and state of the art information retrieval technology to a structured document collection. In hindsight, we have not looked deeply into the possibilities for integrating structure, apart from describing a simple model with which structural properties of documents can be injected into the retrieval process. The experimental results and analysis of the assessments and additional fourth and fifth run showed us that using document component only is too naive an approach for estimation of component coverage.

Future work includes more extensive experimentation with the model described in this paper, especially in the area of relevance feedback and research into a fair normalization mechanism for removing the bias of the language model for short components.

Acknowledgements

Gabriella Kazai has greatly improved our understanding of the definitions of relevance and coverage in INEX.

References

[1] Inex relevance assessment guide. In N. Fuhr, N. Goevert, G. Kazai, and M. Lalmas, editors, *Proceedings of the First Workshop of the Initiative for the Evaluation of XML Retrieval (INEX), Dagstuhl 9-11 Dec. 2002*, ERCIM Workshop Proceedings, March 2003.

[2] C.L. Barry. User-defined Relevance Criteria: An Exploratory Study. *Journal of the American Society for Information Science*, 45(3):149–159, 1994.

[3] N.J. Belkin, R.N. Oddy, and H.M. Brooks. ASK for Information Retrieval: Part 1. Background and Theory. *Journal of Documentation*, 38(2):61–71, 1982.

[4] N.J. Belkin, R.N. Oddy, and H.M. Brooks. ASK for Information Retrieval: Part 2. Results of a Design Study. *Journal of Documentation*, 38(3):145–164, 1982.

[5] H.W. Bruce. A Cognitive View of the Situational Dynamism of User-centered Relevance Estimation. *Journal of the American Society for Information Science*, 45(3):142–148, 1994.

[6] P. Buneman, L. Libkin, D. Suciu, V. Tannen, and L. Wong. Comprehension Syntax. In *SIGMOD Record*, 1994.

[7] F.J. Burkowski. Retrieval Activities in a Database Consisting of Heterogeneous Collections of Structured Texts. In *Proceedings of the 15th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 112–125, 1992.

[8] C.L.A. Clarke, G.V. Cormack, and F.J. Burkowski. An Algebra for Structured Text Search and a Framework for its Implementation. *The Computer Journal*, 38(1):43–56, 1995.

[9] M. Consens and T. Milo. Algebras for Querying Text Regions. In *Proceedings of the ACM Conference on Principles of Distributed Systems*, pages 11–22, 1995.

[10] T. Grust. Accelerating XPath Location Steps. In *Proceedings of the 21st ACM SIGMOD International Conference on Management of Data*, pages 109–120, 2002.

[11] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, Twente, The Netherlands, 2000.

[12] D. Hull. Using Statistical Testing in Evaluation of Retrieval Experiments. In *Proceedings of the 16th ACM SIGIR Conference on Research and Development in Information Retrieval*, 1993.

[13] J. Jaakkola and P. Kilpelainen. Nested Text-Region Algebra. Technical Report C-1999-2, Department of Computer Science, University of Helsinki, 1999.

[14] P. Kilpelainen and H. Mannila. Retrieval from Hierarchical Texts by Partial Patterns. In *Proceedings of the 16th ACM SIGIR International Conference on Research and Development in Information Retrieval*, 1993.

[15] R.C. Miller. *Light-Weight Structured Text Processing*. PhD thesis, Computer Science Department, Carnegie-Mellon University, 2002.

[16] S. Mizarro. How Many Relevances in Information Retrieval? *Interacting With Computers*, 10(3):305–322, 1998.

[17] A.R. Schmidt, M.L. Kersten, M.A. Windhouwer, and F. Waas. Efficient Relational Storage and Retrieval of XML Documents. In *International Workshop on the Web and Databases (in conjunction with ACM SIGMOD)*, pages 47–52, 2000.