Information Retrieval, 8, 601–629, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

A Fusion Approach to XML Structured Document Retrieval

RAY R. LARSON ray@sims.berkeley.edu
School of Information Management and Systems, University of California, Berkeley, Berkeley, CA 94720-4600, USA

Abstract. In this paper we evaluate the application of data fusion or meta-search methods, combining different algorithms and XML elements, to content-oriented retrieval of XML structured data. The primary approach is the combination of a probabilistic methods using Logistic regression and the Okapi BM-25 algorithm for estimation of document relevance or XML element relevance, in conjunction with Boolean approaches for some query elements. In the evaluation we use the INEX XML test collection to examine the relative performance of individual algorithms and elements and compare these to the performance of the data fusion approaches.

Keywords: XML retrieval, data fusion, probabilistic retrieval, logistic regression

1. Introduction

XML has emerged as a *lingua franca* of the WWW and is rapidly replacing other formats as the preferred form for information ranging from protocol exchange messages to full documents and databases. With this rapid growth, and the conversion of information resources to XML, comes an increasing need for effective search and retrieval of XML documents and their constituent elements. The XML retrieval problem (as formulated for the Initiative for the Evaluation of XML Retrieval or INEX) (Fuhr et al. 2002) is to retrieve not only complete documents, but also the *component* parts of those documents that may contain relevant information. Thus, an effective retrieval system for XML retrieval must deal with retrieval and ranking of both full documents and *components* derived from the document structure. In this research, and in the Cheshire II system used for the research, we define a *document component*, or simply *component*, as a continuous segment of an XML document representing some part of an XML document tree structure, and comprised of one or more XML document *elements* (i.e., spans of data consisting of a begin tag, and ending with the corresponding end tag).

In the research reported here, we examine the application of data fusion methods to the XML retrieval problem. The basic notion of "data fusion" or "meta-search" approaches to IR is quite simple and intuitively appealing. Early observations by researchers examining different algorithms and query combination methods (Croft 2000, Shaw and Fox 1994, Belkin et al. 1995) indicated that no single retrieval algorithm could be shown to be consistently better than any other algorithm for all types of searches, and therefore some combination of different search strategies should be more effective than any single strategy.

In principle, we would expect that the more evidence the system has about the relationship between a query and a document (including the sort of structural information about the documents found in XML documents), the more accurate it should be in predicting the probability that the document will satisfy the user's need. Other researchers have also shown that additional information about the location and proximity of Boolean search terms can also be used to provide a ranking score for a set of documents (Hearst 1996). In addition, the inference net model for IR has shown that Boolean search elements can be used as additional evidence of the probability of relevance in the context of a larger network of probabilistic evidence (Turtle and Croft 1990).

The concept of data fusion tested in early TREC evaluations, where a number of participating groups found that fusion of multiple retrieval algorithms provided an improvement over a single search algorithm (Shaw and Fox 1994, Belkin et al. 1995). With ongoing improvements of the algorithms used in the TREC main (i.e., ad hoc retrieval) task, later analyses (Lee 1997, Beitzel et al. 2003) found that the greatest effectiveness improvements appeared to occur between relatively ineffective individual methods. These researchers also observed that the fusion of ineffective techniques, while often approaching the effectiveness of the best single IR algorithms, seldom exceeded them for individual queries and never exceeded their average performance (Beitzel et al. 2003). However, these observations were based on the retrieval of full documents where the query results from multiple algorithms were combined into a single result set.

In addition to these studies of algorithms, some early analyses of search results that were based on retrieval from different representations, or component parts, of documents (Katzer et al. 1982, Das-Gupta and Katzer 1983) showed that those different representations provided similar overall retrieval performance, but retrieved different sets of relevant and non-relevant documents.

In the following experiments we combine not only separate algorithms, but also combinations of separate indexes derived from different *components* of XML documents, where the index statistics are derived from the individual components within the document and collection rather than from the entire document, with result lists that merge components ranging from full documents to individual bibliographic entries from a document's references. In this analysis we are examining a pair of hypotheses:

H_0: For XML retrieval, there is no difference between effective individual search algorithms and fusion of multiple algorithms (the null hypothesis).

H_1: Fusion of the results of searches of different components of XML documents is more effective than searches of single components.

The research reported here extends our work on fusion approaches conducted for the 2002 and 2003 INEX Evaluations (Larson 2003, 2004) using the Cheshire II XML retrieval system. The basic approach used in INEX and in the current evaluation was to combine the results of searching different document components and different probabilistic retrieval algorithms within single search operation. The use of different algorithms, and combinations of algorithms for structured document retrieval has been examined (in a non-XML context) by others (Fuller et al. 1994, Wilkinson 1994, Navarro and Baeza-Yates 1995, Kaszkiel and

Zobel 1994). Most earlier approaches used vector space algorithms and documents with simpler structure (such as the simple SGML structures used for TREC documents). The INEX database used in this research consists of all Computer Science publications of the IEEE for the years 1995–2002 in the complex XML format used in the publication process. Thus the collection is a medium-scale digital library of Computer Science information about 500 MB in size.

In the remainder of the paper we present the experiments and the evaluation results for combining different retrieval algorithms and XML components (including full articles) using data fusion methods with the INEX XML test collection. The following section discusses the algorithms used in the evaluation and reviews related work. We then describe the experimental methodology, the test collection used, and the characteristics of the queries. Finally, results of the evaluation, suggestions for future work and conclusions are presented.

2. The retrieval algorithms and fusion operators

In his analysis of fusion approaches to improving retrieval performance (Lee 1997) found that the best results were obtained by combining algorithms where similar sets of relevant documents were returned but that retrieved different sets of non-relevant documents. With this in mind, we chose for this research two probabilistic algorithms that at fulfill this criteria. The first algorithm is based on logistic regression and the second is the well-known Okapi BM-25 algorithm.

We conducted an analysis of the overlap between the result lists retrieved by these algorithms. We found that on average, over half of the result lists retrieved by each algorithm in these overlap tests were both non-relevant *and* unique to that algorithm, fulfilling Lee's criteria for effective algorithm combination: similar sets of relevant documents and different sets of non-relevant. We will return to parts of this overlap analysis in the later evaluation and discussion section.

In the remainder of this section we describe the Logistic Regression and Okapi BM-25 algorithms that were used for the evaluation and we also discuss the methods used to combine the results of the different algorithms. The algorithms and combination methods are implemented as part of the Cheshire II XML/SGML search engine (Larson 2003, 2002) which also supports a number of other algorithms for distributed search and operators for merging result lists from ranked or Boolean sub-queries.

2.1. Logistic regression algorithm

The *logistic regression* (LR) algorithm used in this study was originally developed by Cooper et al. (1992) and shown to provide good full-text retrieval performance in the TREC ad hoc task and in TREC interactive tasks (Larson 2001) and for distributed IR (Larson 2002). As originally formulated, the LR model of probabilistic IR attempts to estimate the probability of relevance for each document based on a set of statistics about a document collection and a set of queries in combination with a set of weighting coefficients for those statistics. The statistics to be used and the values of the coefficients are obtained from regression analysis of a sample of a collection (or similar test collection) for some set of queries where relevance and non-relevance has been determined. More formally, given a particular query and a particular document in a collection $P(R \mid Q, D)$ is calculated and the documents or components are presented to the user ranked in order of decreasing values of that probability. To avoid invalid probability values, the usual calculation of $P(R \mid Q, D)$ uses the "log odds" of relevance given a set of S statistics, s_i, derived from the query and database, such that:

$$\log O(R \mid Q, D) = b_0 + \sum_{i=1}^{S} b_i s_i \tag{1}$$

where b_0 is the intercept term and the b_i are the coefficients obtained from the regression analysis of the sample collection and relevance judgements. The final ranking is determined by the conversion of the log odds form to probabilities:

$$P(R \mid Q, D) = \frac{e^{\log O(R|Q,D)}}{1 + e^{\log O(R|Q,D)}} \tag{2}$$

Based on the structure of XML documents as a tree of XML elements, we define a "document component" as an XML subtree that may include zero or more subordinate XML elements or subtrees with text as the leaf nodes of the tree. For example, in the XML Document Type Definition (DTD) for the INEX test collection used in this study (described in detail in the introductory paper of this special issue), an article (marked by XML tag <*article*>) contains front matter (<*fm*>), a body (<*bdy*>) and optional back matter (<*bm*>). The front matter (<*fm*>), in turn, can contain a header <*hdr*> and may include editor information (<*edinfo*>), author information (<*au*>), a title group (<*tig*>), abstract (<*abs*>) and other elements. A title group can contain elements including article title (<*atl*>) the page range for the article (<*pn*>), and these in turn may contain other elements, down to the level of individual formatted words or characters. Thus, a component might be defined using any of these tagged elements. However, not all possible components are likely to be useful in content-oriented retrieval (e.g., tags indicating that a word in the title should be in italic type, or the page number range) therefore we defined the retrievable components selectively, including document sections and paragraphs from the article body, and bibliography entries from the back matter (see Table 2).

Naturally, a full XML document may also be considered a "document component". As discussed below, the indexing and retrieval methods used in this research take into account a selected set of document components for generating the statistics used in the search process and for extraction of the parts of a document to be returned in response to a query. Because we are dealing with not only full documents, but also document components (such as sections and paragraphs or similar structures) derived from the documents, we will use C to represent document components in place of D. Therefore, the full equation describing the LR algorithm used in these experiments is:

$$\begin{aligned}\log O(R \mid Q, C) = {} & -3.70 + \left(1.269 \cdot \left(\frac{1}{|Q_c|}\sum_{j=1}^{|Q_c|}\log qtf_j\right)\right) + (-0.310 \cdot \sqrt{|Q|}) \\ & + \left(0.679 \cdot \left(\frac{1}{|Q_c|}\sum_{j=1}^{|Q_c|}\log tf_j\right)\right) + (-0.0674 \cdot \sqrt{cl}) \\ & + \left(0.223 \cdot \left(\frac{1}{|Q_c|}\sum_{j=1}^{|Q_c|}\log \frac{N - n_{t_j}}{n_{t_j}}\right)\right) + (2.01 \cdot \log |Q_d|) \end{aligned} \tag{3}$$

where

Q is a query containing terms T,
$|Q|$ is the total number of terms in Q,
$|Q_c|$ is the number of terms in Q that also occur in the document component,
tf_j is the frequency of the jth term in a specific document component,
qtf_j is the frequency of the jth term in Q,
n_{t_j} is the number of components (of a given type) containing the jth term,
cl is the document component length measured in bytes.
N is the number of components of a given type in the collection.

This equation, used in estimating the probability of relevance in this research, is essentially the same as that used in Cooper et al. (1994). The coefficients were estimated using relevance judgements and statistics from the TREC/TIPSTER test collection. In this evaluation we used the same coeffients for each of the main document components used. This means, that we, in effect, are treating all components smaller than a full document as if they were exactly that, small documents.

2.2. *Okapi BM-25 algorithm*

The version of the Okapi BM-25 algorithm used in these experiments is based on the description of the algorithm in Robertson and Walker (1997), and in TREC notebook proceedings (Robertson et al. 1998). As with the LR algorithm, we have adapted the Okapi BM-25 algorithm to deal with document components:

$$\sum_{j=1}^{|Q_c|} w^{(1)} \frac{(k_1+1)tf_j}{K+tf_j} \frac{(k_3+1)qtf_j}{k_3+qtf_j} \tag{4}$$

Where (in addition to the variables already defined):

K is $k_1((1-b)+b \cdot cl/avcl)$,
k_1, b and k_3 are parameters (1.5, 0.45 and 500, respectively, were used),

avcl is the average component length measured in bytes,
$w^{(1)}$ is the Robertson-Sparck Jones weight,

$$w^{(1)} = \log \frac{\left(\frac{r+0.5}{R-r+0.5}\right)}{\left(\frac{n_{t_j}-r+0.5}{N-n_{t_j}-R-r+0.5}\right)}$$

r is the number of relevant components of a given type that contain a given term,
R is the total number of relevant components of a given type for the query.

Our current implementation uses only the *a priori* version (i.e., without relevance information) of the Robertson-Sparck Jones weights, and therefore the $w^{(1)}$ value is effectively just an IDF weighting. The results of searches using our implementation of Okapi BM-25 and the LR algorithm seemed sufficiently different to offer the kind of conditions where data fusion has been shown to be be most effective (Lee 1997), and our overlap analysis of results for each algorithm (described in the evaluation and discussion section) has confirmed this difference and the fit to the conditions for effective fusion of results.

2.3. Boolean operators

The system used supports searches combining probabilistic and (strict) Boolean elements, as well as operators to support various merging operations for both types of intermediate result sets. Although strict Boolean operators and probabilistic searches are implemented within a single process, using the same inverted file structures, they really function as two parallel *logical* search engines. Each logical search engine produces a set of retrieved documents. When a only one type of search strategy is used then the result is either a probabilistically ranked set or an unranked Boolean result set. When both are used within in a single query, combined probabilistic and Boolean search results are evaluated using the assumption that the Boolean retrieved set has an estimated $P(R \mid Q_{\text{bool}}, C) = 1.0$ for each document component in the set, and 0 for the rest of the collection. The final estimate for the probability of relevance used for ranking the results of a search combining strict Boolean and probabilistic strategies is simply:

$$P(R \mid Q, C) = P(R \mid Q_{\text{bool}}, C)P(R \mid Q_{\text{prob}}, C) \tag{5}$$

where $P(R \mid Q_{\text{prob}}, C)$ is the probability of relevance estimate from the probabilistic part of the search, and $P(R \mid Q_{\text{bool}}, C)$ is the Boolean. In practice the combination of strict Boolean "AND" and the probablistic approaches has the effect of restricting the results to those items that match the Boolean part, with ranking based on the probabilistic part. Boolean "NOT" provides a similar restriction of the probabilistic set by removing those document components that match the Boolean specification. When Boolean "OR" is used the probabilistic and Boolean results are merged (however, items that only occur in the Boolean result, and not both, are reweighted as in the "fuzzy" and merger operations described below.

A special case of Boolean operators in Cheshire II is that of proximity and phrase matching operations. In proximity and phrase matching the matching terms must also satisfy proximity constraints (both term order and adjacency in the case of phrases). Thus, proximity operations also result in Boolean intermediate result sets.

2.4. Result combination operators

The Cheshire II system used in this evaluation provides a number of operators to combine the intermediate results of a search from different components or indexes. With these operators we have available an entire spectrum of combination methods ranging from strict Boolean operations to fuzzy Boolean and normalized score combinations for probabilistic and Boolean results. These operators are the means available for performing fusion operations between the results for different retrieval algorithms and the search results from different components of a document. We will only describe four of these operators here, because they were the only types used in the evaluation reported in this paper.

The MERGE_MEAN operator combines the two result lists (like a Boolean OR) but takes the mean of the weights from items in both lists or half the weight of items in only a single list. Similarly, the MERGE_NORM operator combines the two results but it performs the min-max normalization of the weights suggested by Lee (1997) before it takes the mean of the weights from items in both lists and half of the weight of items in only a single list. The MERGE_NSUM operator performs min-max normalization of the weights, but sums the normalized weights. The MERGE_CMBZ operator is based on the "CombMNZ" fusion algorithm developed by Shaw and Fox (1994) and used by Lee (1997). In our version we take the normalized scores, but then further enhance scores for components appearing in both lists (doubling them) and penalize normalized scores appearing low in a single result list, while using the unmodified normalized score for higher ranking items in a single list.

3. Experimental methods

In this section we discuss the data and methods used in conducting the evaluation. We begin by describing the test collection and the indexing methods applied to it. We then describe the query processing and combinations of operators used in the experiments, and the evaluation methods and metrics.

3.1. Indexing the INEX collection

The INEX test collection (version 1.4) is composed of an XML document collection, sets of search topics and document component relevance assessments. The INEX XML document collection contains the full content of the IEEE Computer Society's journal publications starting in 1995, which represents 12107 article-level documents (about 525 MB in size). The specific contents and coverage are described in the introductory paper of this special issue.

During the INEX evaluation process, each participating organisation submitted a number of candidate topics in XML format (figures 1 and 2 show typical INEX topics), and some

```
 <?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE inex_topic SYSTEM "topic.dtd">
<inex_topic topic_id="98" query_type="CO" ct_no="26">
<title>"Information Exchange", +"XML", "Information Integration"</title>
<description>How to use XML to solve the information exchange
(information integration) problem, especially in heterogeneous data
sources?  </description>
<narrative>Relevant documents/components must talk about
techniques of using XML to solve information exchange
(information integration) among heterogeneous data sources
where the structures of participating data sources are
different although they might use the same ontologies
about the same content.  </narrative>
<keywords>information exchange, XML, information
integration, heterogeneous data sources</keywords>
</inex_topic>
```

Figure 1. INEX CO Topic #98.

of these were selected (along with occasional modifications) as the queries for each year. After all groups had returned their retrieval results for the queries, the results were pooled and relevance assessments of selected document components (and other components within the same document) were performed (in most cases) by the participants who submitted the topics. For this research we used the 2003 INEX topics and assessments for the evaluation. INEX topics include two query types, CO or “Content Only”, like that shown in figure 1 and CAS or “Content and Structure”, as shown in figure 2, in which an extended form of XPath is used to describe the document elements and retrieval criteria. The CAS queries were used for two retrieval tasks, “VCAS” and “SCAS”, where the specified structural elements were treated as suggestions or requirements, respectively. The INEX 2003 topics include 30 CAS topics and 30 CO topics. In this evaluation we used both the INEX CO and CAS topics, although our primary focus is on the CO topics. For both types of query some of the conventions of internet search engines have been adapted to indicate special processing for search terms included in the title element. These include “+” preceding a word or phrase that *should* be present in the result component, “−” to indicate results *should not* contain a term, and double-quotes around terms to indicate that the *exact phrase* is desired in the results. However, these criteria need not (necessarily) be treated as Boolean constraints on the results, and relevance assessments do not strictly enforce them in all cases.

```
 <?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE inex_topic SYSTEM "topic.dtd">
<inex_topic topic_id="74" query_type="CAS" ct_no="75">
<title> //article[about(., 'video streaming
applications')]//sec[about(., 'media stream synchronization')
OR about(., 'stream delivery protocol')] </title>
<description>I am looking for algorithms for stream delivery
in audio and video streaming applications, including audio and
video stream synchronization algorithms used for stream delivery.
</description>
<narrative>Documents/document components describing delivery control
for a specific setting are considered relevant.  Document/document
components describing audio and video stream synchronization
(if this is necessary due to the architecture) are considered
relevant.  Documents/document components describing architectures
or servers for stream delivery (media servers) are relevant, if a
part of the document component describes a communication or
delivery protocol (even if lowlevel).</narrative>
<keywords>algorithms,audio,video,streaming,applications,
stream,delivery,media,stream,synchronization,
stream,delivery,protocol</keywords>
</inex_topic>
```

Figure 2. INEX SCAS Topic #74.

As noted above, the system used in this research permits document components to be defined, indexed and retrieved as if they were individual documents, with separate indexes and ranking statistics used during retrieval. In addition to flexible indexing, it includes facilities for document component retrieval and display, including the ability to request any individual XPATH specification from any document selected during searching. The system also permits a variety of term extraction methods to be specified for indexed elements, including proximity information and different data types (dates, integers, etc.). Only keyword extraction with proximity was used in the tests reported here. In addition several types of normalization can applied to the indexing data extracted from the text nodes of the XML document components. In this study we used only an slightly enhanced version of the Porter stemmer for the extracted indexes (the enhancements correct some of the incorrect stems for specific words).

Each index generated by the system can have its own specialized stopword list, so that, for example, XML elements containing corporate names can have a different set of stopwords from document titles or personal names.

Most of the indexes used for the evaluation used keyword with proximity extraction and stemming of the keyword tokens. Exceptions to this general rule were date elements (which were extracted using date extraction of the year only) and the names of authors which were extracted without stemming or stoplists to retain the full name.

Table 1 lists the document-level (/article) indexes created for INEX and the XPaths of the document elements from which the contents of those indexes were extracted. As noted above, the system permits document component subtrees to be treated as separate documents

Table 1. Article-level indexes for INEX.

Name	Description	Contents
docno	Digital Object ID	//doi
pauthor	Author Names	//fm/au/snm
		//fm/au/fnm
title	Article title	//fm/tig/atl
topic	Content words	//fm/tig/atl
		//abs
		//bdy
		//bibl/bb/atl
		//app
topicshort	Content words 2	//fm/tig/atl
		//abs
		//kwd
		//st
date	Date of publication	//hdr2/yr
journal	Journal title	//hdr1/ti
kwd	Article keywords	//kwd
abstract	Article abstract	//abs
author_seq	Author seq.	//fm/au@sequence
bib_author_fnm	Bib author forename	//bb/au/fnm
bib_author_snm	Bib author surname	//bb/au/snm
fig	Figure contents	//fig
ack	Acknowledgements	//ack
alltitles	All title elements	//atl, //st
affil	Author affiliations	//fm/aff
fno	IEEE article ID	//fno

Table 2. Document components for INEX.

Name	Description	Contents
COMP_SECTION	Sections	//sec|//ss1|//ss2|//ss3
COMP_BIB	Bib Entries	//bib/bibl/bb
COMP_PARAS	Paragraphs	//ilrj|//ip1|//ip2| //ip3|//ip4|//ip5| //item-none|//p| //p1|//p2|//p3| //tmath|//tf
COMP_FIG	Figures	//fig
COMP_VITAE	Vitae	//vt

with their own separate indexes. Tables 2 and 3 describe the XML components defined for the evaluation and the component-level indexes that were created for them.

Table 2 shows the components and the path used to define them. The COMP_SECTION component consists of each identified full section (<sec> $\cdots$ </sec>) in all of the documents, permitting each individual section of a article to be retrieved separately. Similarly, each of the COMP_BIB, COMP_PARAS, and COMP_FIG components, respectively, treat each bibliographic reference (<bb> $\cdots$ </bb>), paragraph (with all of the alternative paragraph elements shown in Table 2), and figure (<fig> $\cdots$ </fig>) as individual documents that can be retrieved separately from the entire document.

Table 3. Component indexes for INEX.

Component or Name	Description	Contents
COMP_SECTION		
sec_title	Section title	//sec/st
sec_words	Section words	//sec
COMP_BIB		
bib_author	Bib. author	//au
bib_title	Bib. title	//atl
bib_date	Bib. date	//pdt/yr
COMP_PARAS		
para_words	Paragraph words	*[†]
COMP_FIG		
fig_caption	Figure caption	//fgc
COMP_VITAE		
Vitae_words	Words from vitae	//vt

Table 3 describes the XML component indexes created for the components described in Table 2 (Note also that the para_words index includes all subelements of paragraph elements). These indexes make the contents of a number of individual document components available for searching. For example, sections (COMP_SECTION) of the INEX documents are retrievable by their titles, or by any terms occurring in the section. These indexes also support proximity searches, so phrase search is available for most of the indexes. Bibliographic references in the articles (COMP_BIB) are made accessible by the author names, titles, and publication date of the individual bibliographic entry, with proximity searching supported for bibliography titles. Individual paragraphs (COMP_PARAS) are searchable by any of the terms in the paragraph, also with proximity searching. Individual figures (COMP_FIG) are indexed by their captions, and vitae (COMP_VITAE) are indexed by keywords within the text, with proximity support.

3.2. Query characteristics and tests

Analysis of the INEX 2002 relevance assessments showed that relatively large document components were more likely be judged relevant. This also makes sense, given that the CO queries, in most cases specified a discussion of a topic as the desired result, and elements smaller than a paragraph are generally insufficient for such a discussion. Therefore in our INEX 2003 submission, and the subsequent tests reported here, we restricted results to document components including: complete articles, the article body only, individual sections (including nested subsections), and paragraphs (including the different tag types shown for the COMP_PARAS component in Table 2). The query construction method described below was applied to each of these component types, and the results combined using the min-max normalization of the weights in the result lists, as suggested by Lee (1997).

The INEX topics, as discussed above and shown in figures 1 and 2 may include *suggested* constraints on the terms used. That is, they specify phrases, desired terms and deprecated terms. To construct the queries for this evaluation, the parts of the topic to be used in the query were extracted. In all of the tests reported here only the terms from "title" and "keywords" elements of the topic were used in constructing the queries.

The simplest form of query is one where no phrases, suggested terms, or deprecated terms were specified in the topic. In this case the query (for a given index and document component) consists of just a list of the terms extracted from the topic, along with a specification of the probabilistic algorithm to use in searching and the index to be searched. This type of query alone was used in the "PROB_BASE" and "OKAPI_BASE" tests shown in Tables 5 and 6 which use, respectively, the LR and Okapi algorithms.

If the terms extracted from a topic include a phrase specification, the phrase was extracted and a Boolean phrase search of the given index was added to the simple query above as a sub-query, combined with the probabilistic part of the query using the MERGE_MEAN, MERGE_NORM or MERGE_CMBZ, operators described above.

Deprecated terms were handled by constructing a Boolean subquery using a Boolean NOT (restricting the results to components that did not contain the deprecated terms). Desired terms were extracted and a separate ranked search performed with only the desired terms, but with increased query term frequency for them. The results of these subqueries were then combined with the simple (base) search using the MERGE_NORM or MERGE_CMBZ operator. Because topics often contained all of the above types of term specification, the resulting generated queries were often quite complex and this was compounded by the use of multiple indexes and components for most queries.

The tests shown in Tables 5 and 6 that include "_FULL" in the name include all of the above expansions of the topic terms in the queries. Thus, "PROB_FULL" and "OKAPI_FULL" use the LR and Okapi algorithms, respectively, and include the full expansion.

For fusion operations between different indexes for a particular document component, the MERGE_NORM operator was used to combine the sub-query results. In Tables 5 and 6 "FUSION_FULL" combines full queries of only the topic, sec_words, and para_words indexes for both LR and Okapi, "FUSION_T_FULL" combines both the topic, alltitles, sec_words, sec_title, and para_words, "FUSION_TA_FULL" adds the abstract index to this. As in the preceding, "FUSION_T_P_ABS_FULL" and "FUSION_T_P_ABS_FULL" use the same indexes, but perform an additional LR search of the abstract and extract and merge the abstract in the final results used in evaluation. The "FUSION_T_CMBZ" run used the same indexes as "FUSION_T_FULL", but instead of using the MERGE_NORM operator, it used the MERGE_CMBZ operator which enhances scores for components appearing in both intermediate result lists, and penalizes the lower ranked scores in a single list.

3.3. Evaluation metrics

The INEX evaluation metrics described here are discussed in greater detail in Gövert et al. (2003) and Kazai et al. (2004). The INEX evaluations involved relevance assessments of the submitted results of each participating group on two (separate though related) dimensions. The dimensions were *Exhaustivity*, describing the extent to which the document component discusses the topic of request, and *Specificity*, describing the extent to which the document component focuses on the topic of request. For exhaustivity assessments a 4-point scale was used in the INEX evaluation:

0: Not exhaustive, the document component does not discuss the topic of request at all.
1: Marginally exhaustive, the document component discusses only few aspects of the topic of request.
2: Fairly exhaustive, the document component discusses many aspects of the topic of request.
3: Highly exhaustive, the document component discusses most or all aspects of the topic of request.

To assess specificity, another 4-point scale was used:

0: Not specific, the topic of request is not a theme of the document component.
1: Marginally specific, the topic of request is a minor theme of the document component
2: Fairly specific, the topic of request is a major theme of the document component.
3: Highly specific, the topic of request is the only theme of the document component.

For the calculation of the Recall and Precision analogs used for INEX, two different quantizations were used that map the assessed values of these two dimensions for document components into a single value representing relevance. These quantization functions on exhaustiveness (e) and specificity (s), $\mathbf{f}_{\text{quant}}(e, s) : ES \rightarrow [0, 1]$ are:

- A "strict" quantization which indicates whether a given retrieval approach is capable of retrieving highly exhaustive and highly specific document components. This is defined as:

$$\mathbf{f}_{\text{strict}} := \begin{cases} 1 & \text{if } e = 3 \quad \text{and} \quad s = 3 \\ 0 & \text{otherwise} \end{cases}$$

- A "generalized" quantization that scores document elements according to their *degree* of relevance, defined as:

$$\mathbf{f}_{gen} := \begin{cases} 1 & \text{if } (e, s) = (3, 3) \\ 0.75 & \text{if } (e, s) \in \{(2, 3), (3, 2), (3, 1)\} \\ 0.5 & \text{if } (e, s) \in \{(1, 3), (2, 2), (2, 1)\} \\ 0.25 & \text{if } (e, s) \in \{(1, 2), (1, 1)\} \\ 0 & \text{if } (e, s) = (0, 0) \end{cases}$$

Based on the quantized relevance values, procedures that calculate recall/precision curves for standard document retrieval can be applied directly to the results of the quantization functions. The primary measure used in comparing the test results is the Mean Average Precision (MAP). It is worth noting that for those readers unfamiliar with INEX and the evaluation tools that the calculation uses more data points in calculating MAP, and different interpolation than in TREC. The MAP values for INEX CO retrieval are considerably lower than those seen in TREC (the highest MAP of an official content-only run, over all participants in INEX 2003, for strict quantization, was 0.1214. See Gövert et al. (2003) for a more complete discussion of these metrics and their derivation.

4. Evaluation and discussion

As the above discussion of metrics indicates, the relevance judgements used in the INEX test collection are not predicated on binary relevance judgements at the article level, but instead on XML component retrieval with scales of both specificity and exhaustivity. Consequently the fusion approaches that we have been been exploring must consider both the optimal combinations of search elements and algorithms that should be used in the retrieval process. For these experiments we have not re-estimated the logistic regression parameters or examined the possibility of differential weightings that could be applied to the search elements to best estimate the probability of relevance for a given query and document element or combination of elements. However, in the conclusions we will show some recent preliminary results from re-estimating the logistic regression parameters for different XML components.

In this section we will first examine the overlap analysis that explores the distribution of relevant components in the results obtained with different retrieval algorithms. We then examine how retrieval of individual components compares to combination of components in the results. We will then present and discuss how some of the fusion methods tested perform relative to the base methods for both the content-only (CO) and the strict content and structure (SCAS) INEX tasks. Finally we will discuss the results and consider the implications of the analyses and results for XML retrieval.

4.1. Overlap analysis

In our introduction to the retrieval algorithms used in this study we pointed out how (Lee 1997) found that the most effective fusion algorithms were those that combined base algorithms that retrieved similar sets of relevant documents, but different sets of non-relevant documents. In this section we examine the overlap of relevant and non-relevant results for the LR and Okapi BM-25 algorithms.

We conducted an analysis of the results and relevance for base versions of each algorithm (PROB_BASE and OKAPI_BASE, as described above) using the INEX 2003 "Content Only" queries and relevance judgements and the top-ranked 1500 items for each algorithm. In Table 4 the raw counts of components retrieved retrieved only by PROB_BASE or OKAPI_BASE along with the counts for components retrieved by both algorithms. Note that the "both" numbers are not the results of fusion, but of the analysis and comparison of the components returned by each separate algorithm. As the table shows, on average, only 47.75% of the combined results were retrieved by both algorithms, while the remaining 52.25% of each result set was unique to one algorithm or the other. Table 4 also shows the percentages of the retrieved components judged to be relevant (for all non-zero combinations of specificity and exhaustivity). On average, the majority of the relevant items (67.43%) were retrieved by both algorithms, and the PROB_BASE-only and OKAPI_BASE-only items were 14.13% and 18.44% respectively.

In addition to the data shown in Table 4 we also examined the retrieved items that matched the relevant items under the "strict" metric for relevance. The results were not radically different from the generalized relevance criteria discussed above. The common results had the majority (68.68%) of the highly relevant items (using the "strict" metric) while the PROB_BASE-only and OKAPI_BASE-only items had 15.19% and 16.14% respectively. Interestingly, these similar average percentages mask the fact that each algorithm performed quite differently on different queries, with each algorithm out-performing the other in some cases, and for a few queries each algorithm uniquely retrieved more relevant documents than appeared in the common set. This is shown in Table 4 where, for example, PROB_BASE's logistic regression algorithm is clearly is superior for topic 96, while the OKAPI_BASE BM-25 algorithm is superior for topic 111. Needless to say, we are investigating what causes this differential for particular queries, but we have no firm conclusions to report yet.

Many additional analyses of the overlap between the results from the base algorithms and the fusion results were conducted as well. These showed, as expected, that fusion is not

Table 4. Overlap analysis: Numbers of components and percentage of relevant components retrieved by OKAPI_BASE (OK) and PROB_BASE (PR).

TOPIC	Num both	Num OK only	Num PR only	OK % relev.	PR % relev.	Both % relev.
91	946	554	554	11.07	12.65	76.28
92	827	673	673	11.54	23.08	65.38
93	933	567	567	13.91	15.75	70.34
94	717	783	783	27.85	5.48	66.67
95	839	661	661	3.93	25.15	70.92
96	188	1312	1312	0.99	67.33	31.68
97	881	619	619	28.57	9.52	61.90
98	830	670	670	20.94	15.15	63.91
99	877	623	623	28.85	1.54	69.62
100	862	638	638	0.00	0.00	100.00
101	229	1271	1271	33.33	17.78	48.89
102	318	1182	1182	22.05	43.31	34.65
103	682	818	818	26.53	10.20	63.27
104	1004	496	496	4.26	2.13	93.62
107	557	943	943	48.61	5.56	45.83
108	884	616	616	12.66	0.00	87.34
109	517	983	983	14.52	6.45	79.03
110	966	534	534	17.30	11.70	70.99
111	371	1129	1129	54.40	9.33	36.27
112	787	713	713	2.16	6.03	91.81
113	263	1237	1237	51.38	15.60	33.03
115	837	663	663	11.89	6.15	81.97
116	517	983	983	25.65	14.78	59.57
117	894	606	606	8.47	3.39	88.14
119	1096	404	404	4.21	17.84	77.96
121	480	1020	1020	22.89	28.92	48.19
122	853	647	647	9.09	4.74	86.17
123	937	563	563	5.39	15.49	79.12
124	575	925	925	29.48	16.76	53.76
125	1030	470	470	11.58	17.11	71.31
126	1048	452	452	8.27	9.06	82.68
Means	716.39	783.61	783.61	18.44	14.13	67.43

a guarantee that *all* of the relevant items of individual algorithms will become part of the fused results. Of course, it not possible to know at retrieval time which of the components that are retrieved by different algorithms are relevant and which are not. Therefore both relevant and non-relevant common items in the fusion process are given enhanced weights while relevant items that appear in a single result list are penalized. It appears, however that fusion, on average, improves the rankings for relevant components.

On average, over half of the result lists retrieved by each algorithm in these overlap tests were both non-relevant and unique to that algorithm, fulfilling Lee's criteria for effective algorithm combination: similar sets of relevant documents and different sets of non-relevant. This analysis tends to confirm Lee's observations. Because the majority of items retrieved by a single algorithm are non-relevant, they will tend to be demoted in rank while the common relevant items are promoted, however common non-relevant items will also tend to be promoted.

4.2. Component retrieval analysis

We conducted a number of tests where the retrieval results using different individual indexes and individual components were compared to each other, and to the base and fusion methods. The results are shown graphically in figure 3, where each document component is labeled according to it's XML tag, with the exceptions of "paras" and "sections" which represent all of types of paragraphs, and sections respectively. All of these tests made use of the LR algorithm.

As figure 3 shows, none of the individual components come at all close to the the performance of the baseline LR method PROB_BASE, which merges the "paras", "sections", "article", and "bdy" into a single search result list. We used t-tests on paired samples at the individual query level to test for significant differences between each of the individual component results and the merged baseline results. In spite of the small sample used in this analysis (only the 30 INEX 2003 Content-Only queries) there were statistically significant differences between each of the component results and the combined baseline results, with t values ranging from a low of -3.299 to -5.320 with significance at the 0.003 level or better.

This provides fairly strong support for our H_1 hypothesis that fusion of the results of searches of different components of XML documents is more effective than searches of single components. However, this observation must be accompanied by a caveat. Because the INEX evaluation method requires that judges rate not only the specific component retrieved, but also the parents and children of that component in the XML document structure, there is necessarily overlap of components judged relevant. Attempts are underway to provide a solution to this issue for the INEX 2004. The overlap problem in XML retrieval and proposed solutions are discussed in Kazai et al. (2004).

4.3. Content-only evaluation results

The summary average precision results for the baseline and fusion tests (described in Section 3.2) are shown in Tables 5 and 6 for the strict and generalized quantization of the INEX evaluation metrics, respectively. In these tables ΔP shows the percentage

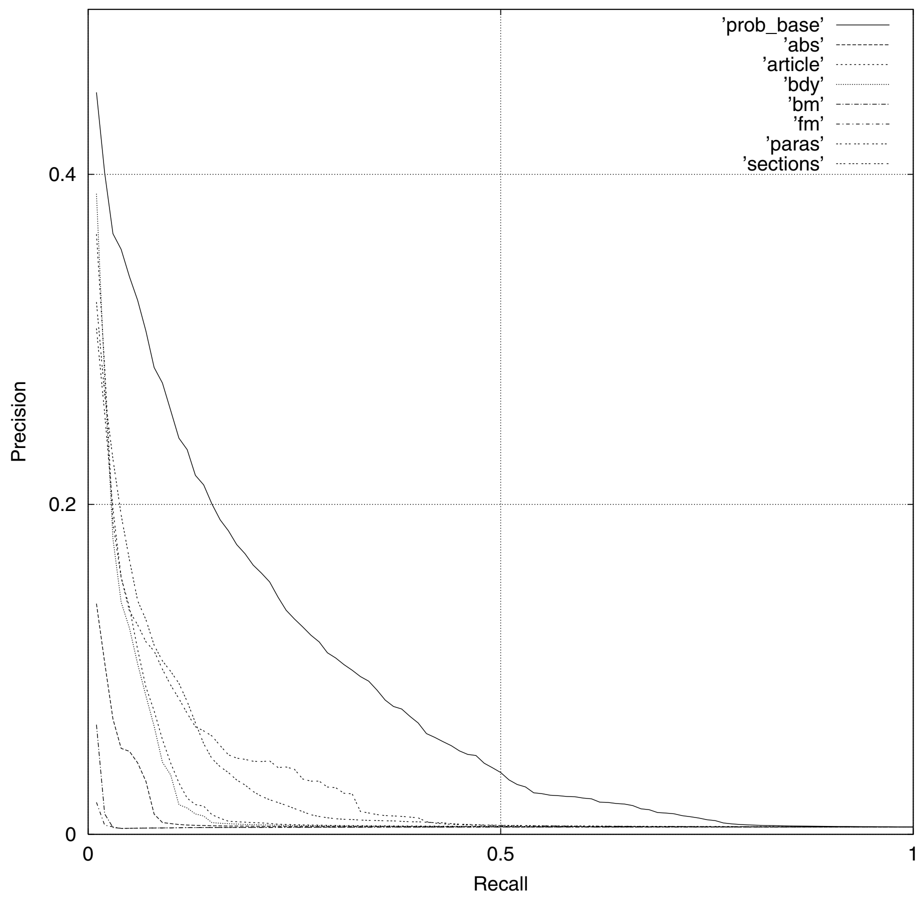

Figure 3. Recall-Precision for different components using LR retrieval algorithms (generalized quantization).

difference for the test from the "PROB_BASE" baseline and 'ΔO shows the difference from "OKAPI_BASE". We also used t-tests on paired samples at the individual query level to test for significant differences between each of the base methods and the other methods. Because of the small sample used in this analysis (only the 30 INEX 2003 Content-Only queries) the best methods were only able to show significance at 0.07 or better under the strict metric (these are shown with asterisks next to the values in Table 5). With the generalized metric the only worst performing fusion method showed a statistically significant difference from the base methods (and that only at the 0.011 significance level when compared to the LR base, and the 0.038 level when compared to the Okapi base). Thus, for this limited sample size we are unable to strongly reject our H_0 null hypothesis, that there is no difference between effective individual search algorithms and fusion of multiple algorithms. However the results obtained here are very encouraging for the effectiveness of fusion methods for XML retrieval.

Table 5. Mean average precision of different algorithms and search element combinations using the strict quantization metric.

Run name	MAP	Δ P	Δ O
FUSION_T_CMBZ	0.1151	27.48*	39.23*
FUSION_T_FULL	0.1002	16.66	30.17*
FUSION_T_P_ABS	0.0930	10.28	24.82
FUSION_TA_FULL	0.0907	7.97	22.88
FUSION_T_CMBZ4	0.0891	6.34	21.51
PROB_BASE	0.0834	0.00	16.21
PROB_FULL	0.0834	0.00	16.21
FUSION_FULL	0.0818	−2.06	14.48
OKAPI_FULL	0.0714	−16.92	2.03
OKAPI_BASE	0.0699	−19.34	0.00

Table 6. Mean average precision of different algorithms and search element combinations using the generalized quantization metric.

Run name	MAP	Δ P	Δ O
FUSION_T_FULL	0.0948	9.26	9.18
FUSION_TA_FULL	0.0925	7.07	6.99
FUSION_FULL	0.0921	6.66	6.57
FUSION_T_P_ABS	0.0913	5.82	5.73
FUSION_T_CMBZ	0.0904	4.86	4.77
OKAPI_FULL	0.0888	3.16	3.08
OKAPI_BASE	0.0861	0.09	0.00
PROB_BASE	0.0860	0.00	−0.09
PROB_FULL	0.0860	0.00	−0.09
FUSION_T_CMBZ4	0.0711	−20.90*	−21.01*

Figures 4 and 5 show the Recall/Precision curves for strict quantization of the base algorithms (PROB_BASE and OKAPI_BASE) in combination with the full expanded queries (figure 4) or the best performing fusion query (FUSION_T_FULL). Figures 6 and 7 show the same tests with the generalized quantization metric.

As Tables 5 and 6 indicate, the use of query expansion, as discussed in Section 3.2, appears to offer some benefit over unexpanded queries for both quantizations, PROB_FULL shows improvement over PROB_BASE and OKAPI_FULL shows improvement over OKAPI_BASE. What is somewhat more interesting is that under strict quantization the LR approach in PROB_FULL performs better than either okapi test, but for generalized quantization both Okapi tests perform better than either LR test (and indeed better than some of the fusion

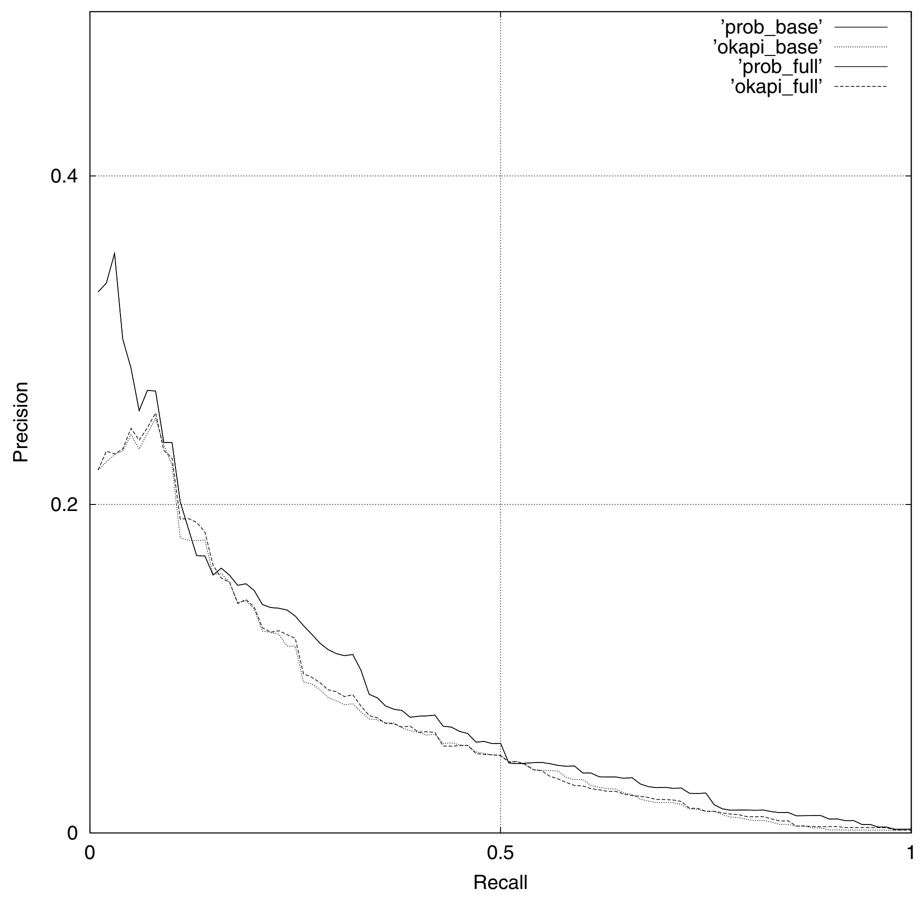

Figure 4. Recall-Precision of LR and Okapi retrieval algorithms (strict quantization).

approaches. This implies that the Okapi algorithm is better at identifying a wider range of degrees of perceived relevance, while the LR algorithm is better at identifying the highly relevant items.

When the two algorithms are combined (with only topic and word searches in FUSION_FULL) the results for both the strict and generalized measures are better than any of the single algorithms. This is different from the kind of results reported in Beitzel et al. (2003), and seems to confirm the improvements from data fusion reported by Lee (1997). When the searches include a separate ranking of title searches merged with the topic searches the performance is further improved and performs the best for both quantizations of all of the query forms examined here. It is worth noting that if the FUSION_T_CMBZ run under strict quantization had been submitted during the official INEX 2003 evaluation, it would have been the second highest ranking run, and the FUSION_T_FULL run under generalized quantization would have been the fifth highest ranked run.

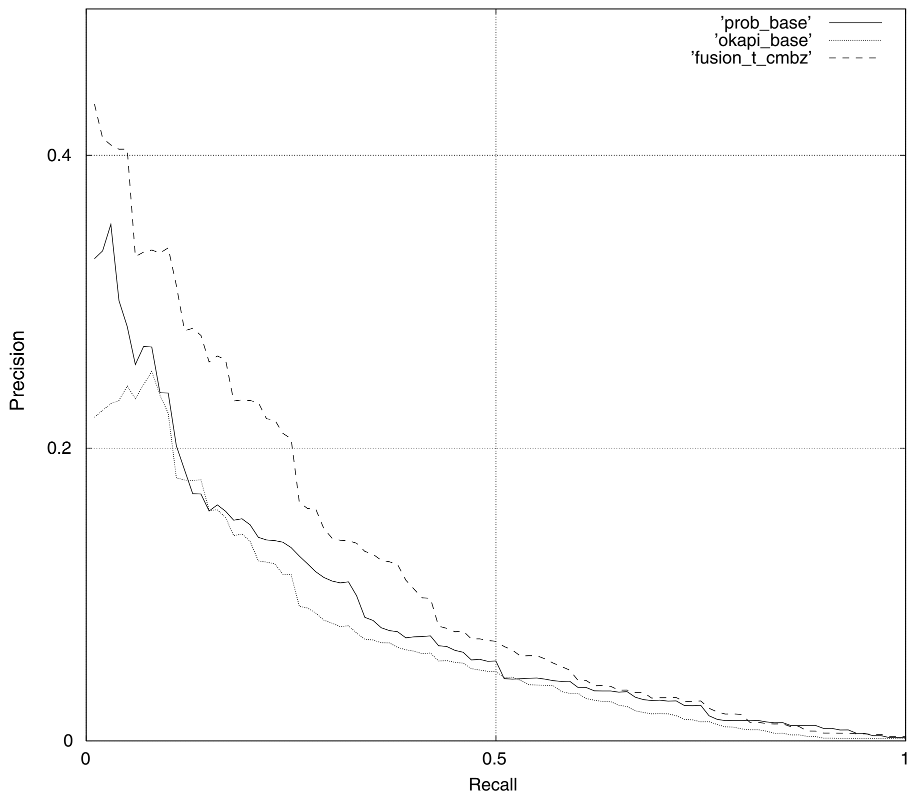

Figure 5. Recall-Precision of the best fusion method compared to baselines (strict quantization).

However, it appears that element indexes cannot be arbitrarily combined in attempting to improve performance, as the reported results show, adding the abstract index results in reduced performance relative to topic and titles alone. Dozens of other combinations of merger operators and indexes were tested, and only the best performing ones are reported here. It is likely to be the case that different XML collections will require different combinations of indexes and operators to achieve similar results.

4.4. SCAS evaluation results

We also applied the fusion approaches tested above to the "content and structure" (SCAS) task of INEX. The Mean Average Precision results for some of these SCAS tests are shown in Table 7. The table shows that the LR-based queries (indicated by "SCAS.P" in the names) seem to be generally less effective than the Okapi-based queries (including "SCAS.O" in the run names). Of course, the SCAS queries are in general more complex than the CO queries, and make use of many additional merging operations driven by the individual Xpath queries. The main operator needed in SCAS queries was the "RESTRICT"

Table 7. Evaluation of SCAS Queries: Mean average precision of different algorithms and search element combinations.

Run name	Avg prec (gen.)	Avg prec (strict)
SCAS.FUS.258	0.2107	0.2403
SCAS.FUS.78	0.2075	0.2395
SCAS.FUS.p28o8	0.1985	0.2304
SCAS.FUS.p8o87	0.2020	**0.2444**
SCAS.O.2	0.2010	0.2205
SCAS.O.7	0.1996	0.2247
SCAS.O.8	**0.2120**	0.2308
SCAS.P.2	0.1877	0.2092
SCAS.P.8	0.1948	0.2174

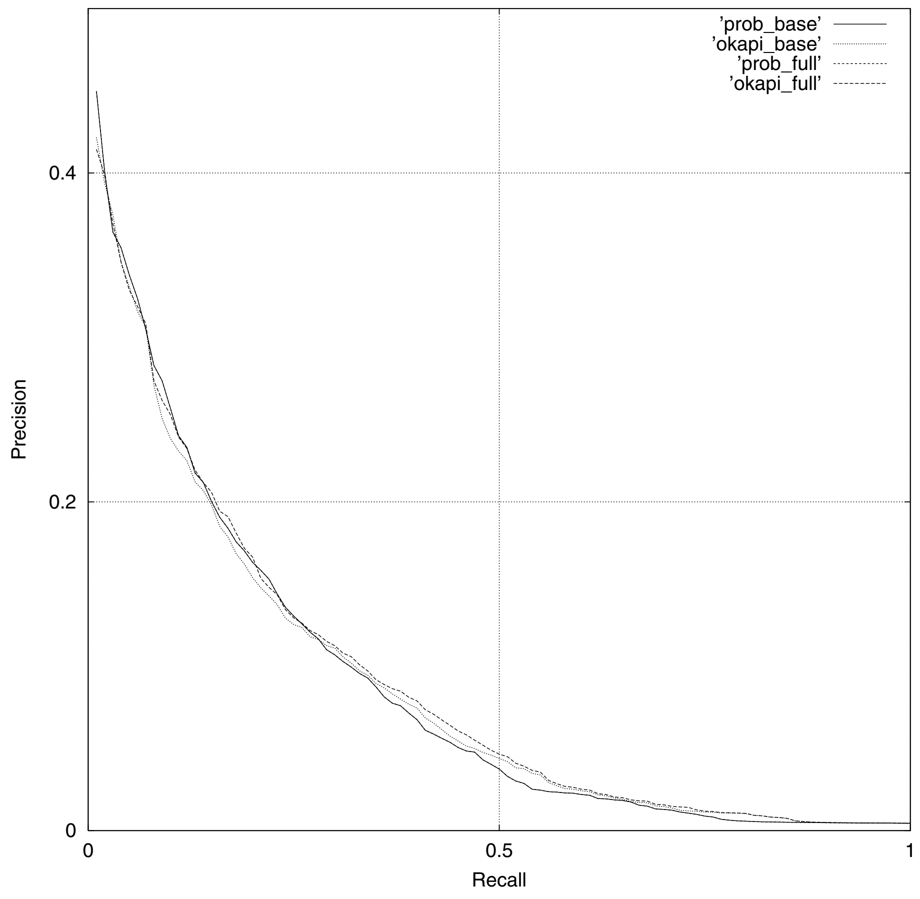

Figure 6. Recall-Precision of LR and Okapi retrieval algorithms (generalized quantization).

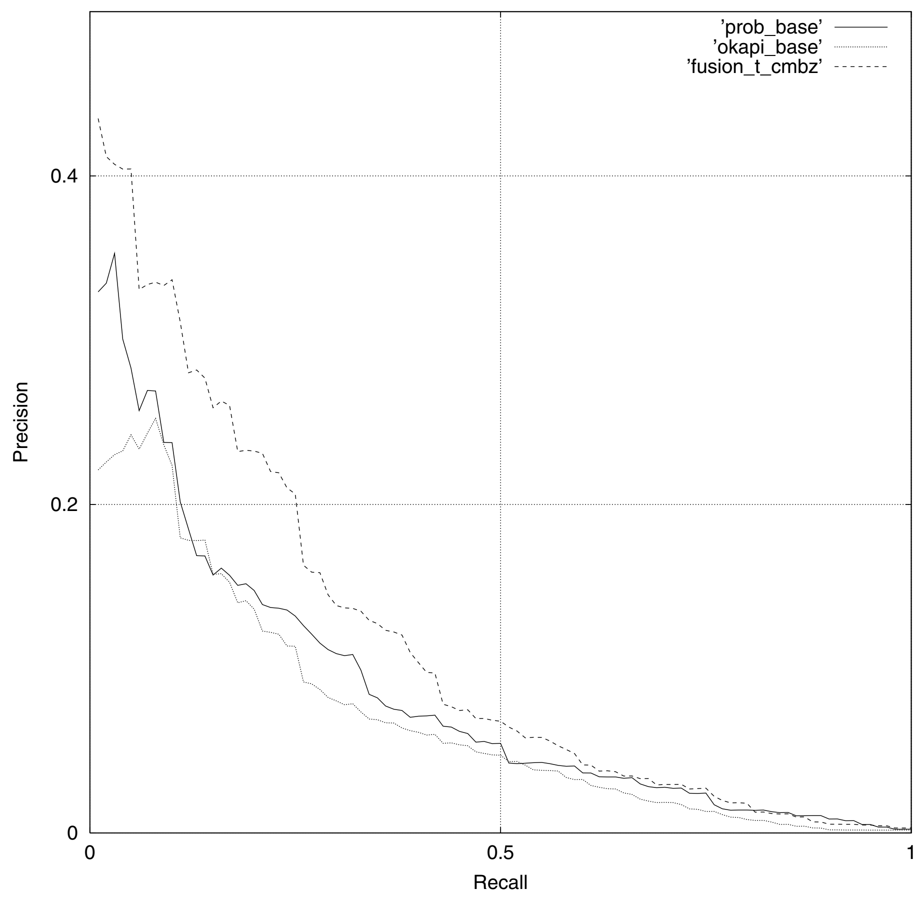

Figure 7. Recall-Precision of the best fusion method compared to algorithm baselines (generalized quantization).

operator that requires components matching queries in one part of an XPath to be contained within other components matching a different query. An example of the type of query constructed for the INEX topic shown in figure 2 is shown in figure 8. In that query the fusion and restriction operators described above are operators used to combine the results of the component searches (using the indexes shown in Tables 1 and 3) for both Okapi BM-25 (specified by "@+") and LR (specified by "@"). This query is automatically generated from the "<title>" element of the INEX topic. Because SCAS queries impose structural constraints on the results, they tend to be more limited in scope than the CO queries, that is, there is less, or no, flexibility in the choice of the elements of the document to return.

The results shown in Table 7 use single digits in the name to indicate particular combinations of fusion and restriction operators and single letters to indicate the ranking algorithm used. Thus, a test containing "o2" in the name used the same fusion and restriction operation as a test containing "p2", but the former used Okapi BM-25 and the latter used the

```
 ((topic @+ {video streaming applications})
!RESTRICT_FROM ((sec_words @+ {media stream synchronization})
!MERGE_SUM (sec_words @+ {stream delivery protocol}))
) !MERGE_MEAN (
(topic @ {video streaming applications})
!RESTRICT_FROM ((sec_words @ {media stream synchronization})
!MERGE_SUM (sec_words @ {stream delivery protocol})))
```

Figure 8. Query for INEX SCAS Topic #74.

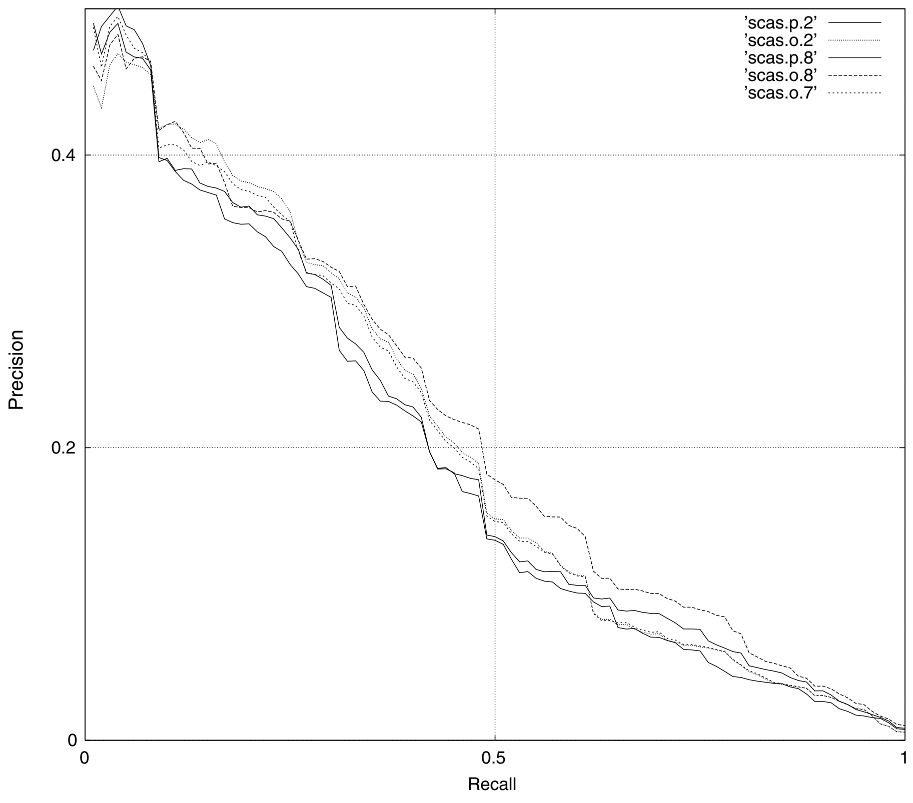

Figure 9. Recall-Precision of LR and Okapi retrieval algorithms for SCAS (generalized quantization).

LR algorithm. The Fusion tests (indicated by names beginning with "SCAS.FUS") each combine results from different combinations of fusion and restriction operators, those with numbers only in the last part of the name are Okapi only tests, and the others combine LR and Okapi results. The best performing SCAS test under the generalized evaluation metric was an Okapi run (SCAS.O.8) that used the "MERGE_NORM" operator when a "AND" was used in an "about" clause in a query, and "MERGE_SUM" was used for "OR". For Xpath expression with separate "about" clauses in nodes on different levels in the document tree, the "RESTRICT_FROM" operators were used. Terms with "+", "−", and quotes were handled the same way as in the CO runs, with added search elements for exact phrase matching, additional query term weighting for "+" and use of Boolean "NOT" for "−".

Figures 9 and 10 show the generalized recall-precision metrics for the SCAS runs above. Figure 9 shows the LR and Okapi results and figure 10 shows the different fusion results.

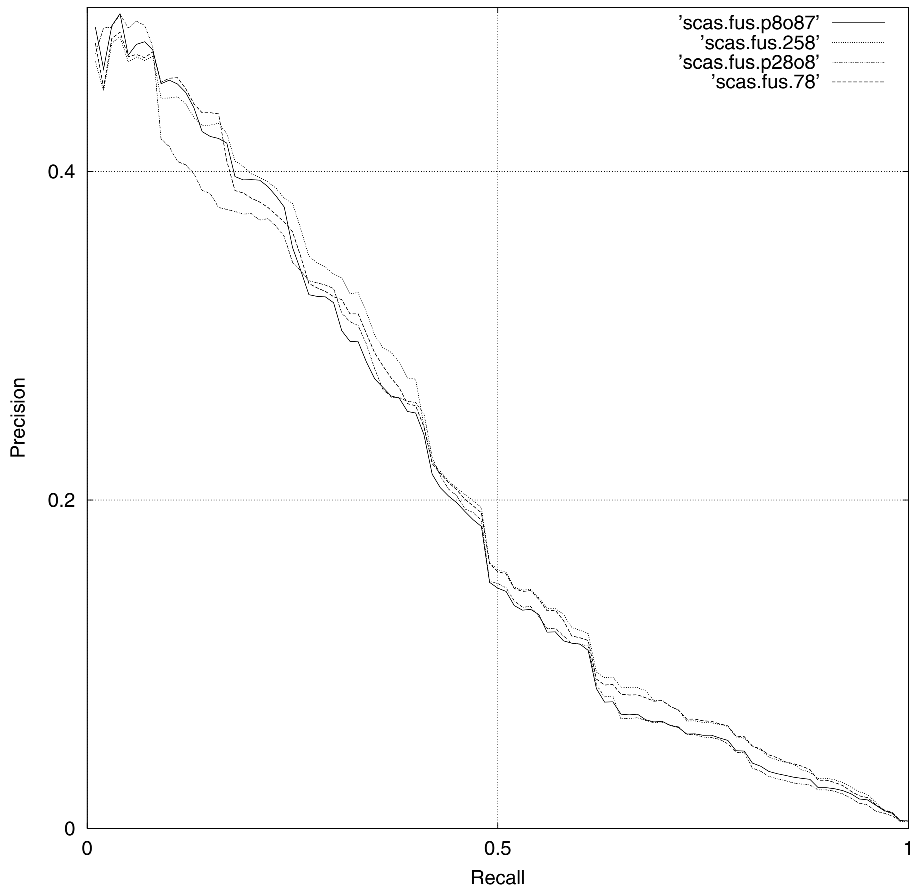

Figure 10. Recall-Precision of fusion approaches for SCAS (generalized quantization).

For the tests reported in Table 7 we also conducted t-tests on paired samples at the individual query level to test for significant differences between the least effective single algorithm and the others, for both the strict and generalized metrics. Under the "strict" metric, this compared the LR test "SCAS.P.2" as the baseline to the other tests. Two of the Okapi-based fusion results ("SCAS.FUS.258" and "SCAS.FUS.78") showed a significant difference from the baseline (with $t = 2.318$ and 2.340 respectively) at the 0.029 or better level. In comparing "SCAS.P.2" under the generalized metric, the only statistically significant difference (at the 0.044 level) was in the "SCAS.O.8" test. Interestingly, under the strict metric, the "SCAS.o.8" test was not as different from the baseline as the fusion results in spite of a overall higher score.

Essentially these results tend confirm the observations of our overlap analysis, that the different algorithms do perform differently on a query by query basis, and when the results of different algorithms are fused some of that distinction is lost. That is, the fusion results including the same algorithm as the baseline tend to be more similar to the baseline than those do not. However, on average fusion appears to improve the overall average results by deriving at least some of the relevant items from the results of different algorithms.

5. Conclusions and further research

At this early stage of the INEX test collection, with a small number of queries and relevance judgements, statistical significance is difficult to obtain, so these observations and conclusions will need to be re-tested and confirmed as the query collection expands. Revisiting our hypotheses for this research:

H_0: For XML retrieval, there is no difference between effective individual search algorithms and fusion of multiple algorithms (the null hypothesis).
H_1: Fusion of the results of searches of different components of XML documents is more effective than searches of single components.

Although we are unable to *strongly* reject H_0, the results seem to suggest that it may be cautiously rejected, subject to further testing. For H_1 the evidence is much stronger, and in fact required by the INEX task.

However, there is much room for further study, in particular this study did not include language models of XML, which have proved to be highly effective in the INEX evaluations. Future work will extend this study to include language model based XML retrieval algorithms and test it in combination with the logistic regression and Okapi algorithms tested here.

In addition we have recently re-estimated the coefficients of the Logistic regression algorithm based on the INEX 2003 relevance assessments. In fact, separate formulae were derived for each of the major components of the INEX XML document structure, providing a different formula for each index/component of the collection. These formulae were used in the official *ad hoc* runs submitted for the INEX 2004 evaluation. For testing purposes we resubmitted the INEX 2003 CO queries used in the "PROB_BASE" tests in this paper, and were able to obtain a mean average precision of 0.1158 under the strict metric and 0.1116 for the generalized metric, thus exceeding the best fusion results reported here. However, these tentative results cannot be properly compared to the tests in this paper, because the data used for training the LR model was obtained using the relevance data associated with the same topics. The true test will be the results in INEX 2004.

In closing we offer a few general observations, questions and directions for futher research derived from these analyses:

1. All fusion operators combine the scores of individual result lists, and all tend to promote those items in common between lists, while demoting those that appear in a single list. Because, as we have observed, a given retrieval algorithm will retrieve documents that are relevant, but which are not retrieved by other algorithms (or which may be ranked below the threshold for returned results by other algorithms), there appears to be a need for further research on fusion algorithms that will preserve more of the *uniquely relevant* items in different result lists.
2. The differences in performance on a query-by-query basis between the LR and Okapi algorithms suggest that each is taking into account some additional clues to relevance for particular queries. Can detailed analysis reveal what these additional clues to relevance might be that are not yet taking into account?
3. In TREC the use of "blind feedback" has proven to be quite effective in boosting performance for a given algorithm. How can blind feedback be used in the XML retrieval task, and what structural constraints should be imposed?

In this paper we have examined the fusion of different algorithms and document components in content-oriented and content and structure XML retrieval. The results indicate that several of the fusion approaches that we tested do perform better than the individual algorithms, and also that some Boolean structural constraints are beneficial (or necessary) for XML retrieval.

XML retrieval is becoming an increasingly important area in IR research, and it provides a new context in which the algorithms and techniques developed over the past half century can be re-examined and evaluated.

Acknowledgments

This work was supported in part by the National Science Foundation and Joint Information Systems Committee(U.K) under *NSF International Digital Libraries Program* award #IIS-9975164.

References

Beitzel SM, Jensen EC, Chowdhury A, Frieder O, Grossman D and Goharian N (2003) Disproving the fusion hypothesis: An analysis of data fusion via effective information retrieval strategies. In: Proceedings of the 2003 SAC Conference, pp. 1–5.

Belkin N, Kantor PB, Fox EA and Shaw JA (1995) Combining the evidence of multiple query representations for information retrieval. Information Processing and Management, 31(3):431–448.

Cooper WS, Gey FC and Chen A (1994) Full text retrieval based on a probabilistic equation with coefficients fitted by logistic regression. In: Harman DK, ed., The Second Text Retrieval Conference (TREC-2) (NIST Special Publication 500-215), National Institute of Standards and Technology, Gaithersburg, MD, pp. 57–66.

Cooper WS, Gey FC and Dabney DP (1992) Probabilistic retrieval based on staged logistic regression. In: 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Copenhagen, Denmark, June 21–24, ACM, New York, pp. 198–210.

Croft WB (2000) Combining approaches to information retrieval. In: Croft WB, ed., Advances in Information Retrieval: Recent research from the Center for Intelligent Information Retrieval, Kluwer, Boston, chapter 1, pp. 1–36.

Das-Gupta P and Katzer J (1983) A study of the overlap among document representations. In: Kuehn JJ, Ed., Research and Development in Information Retrieval, Sixth Annual International ACM SIGIR Conference, National Library of Medicine, Bethesda, Maryland, USA, June 6–8, ACM, pp. 106–114.

Fuhr N, Gövert N, Kazai G and Lalmas M (Eds.) (2002) INEX: Proceedings of the First Workshop of the Initiative for the Evaluation of XML Retrieval, DELOS Network of Excellence on Digital Libraries.

Fuller M, Mackie E, Sacks-Davis R and Wilkinson R (1994) Structured answers for a large structured documents collection. SIGIR '93: Proceedings of the Sixteenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, Pittsburgh, USA, June 27–July 1, pp. 204–213.

Gövert N, Kazai G, Fuhr N and Lalmas M (2003) Evaluating the effectiveness of content-oriented XML retrieval, Technical report, University of Dortmund, Computer Science 6.

Hearst MA (1996) Improving full-text precision on short queries using simple constraints. In: Proceedings of SDAIR '96, Las Vegas, NV, University of Nevada, Las Vegas, pp. 59–68.

Kaszkiel M and Zobel J (1994) Passage retrieval revisited. In: SIGIR '97: Proceedings of the Twentieth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Philadelphia, USA, July 27–31, 1997, pp. 178–185.

Katzer J, McGill MJ, Tessier JA, Frakes W and Das-Gupta P (1982) A study of the overlap among document representations. Information Technology: Research and Development, 1(2):261–274.

Kazai G, Lalmas M and de Vries AP (2004) The overlap problem in content-oriented XML retrieval. In: Järvelin K, Allan J, Bruza P and Sanderson M, Eds., SIGIR 2004: The 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, U.K., July 25–29, ACM, pp. 72–79.

Larson RR (2001) TREC interactive with cheshire II. Information Processing and Management, 37:485–505.

Larson RR (2002) A logistic regression approach to distributed ir. In: SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 11–15, Tampere, Finland, ACM, pp. 399–400.

Larson RR (2003) Cheshire II at INEX: Using a hybrid logistic regression and boolean model for XML retrieval. In: Proceedings of the First Annual Workshop of the Initiative for the Evaluation of XML Retrieval (INEX). DELOS workshop series, pp. 18–25.

Larson RR (2004) Cheshire II at INEX 03: Component and algorithm fusion for XML retrieval. In: INEX 2003 Workshop Proceedings, University of Duisburg, pp. 38–45. http://inex.is.informatik.uni-duisburg.de:2003/

Lee JH (1997) Analyses of multiple evidence combination. In: SIGIR '97: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 27–31, Philadelphia, ACM, pp. 267–276.

Navarro G and Baeza-Yates R (1995) A language for queries on structure and contents of textual databases. In: SIGIR '95: Proceedings of the Eighteeneth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, USA, July 9–13, pp. 93–101.

Robertson SE and Walker S (1997) On relevance weights with little relevance information. In: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM Press, pp. 16–24.

Robertson SE, Walker S and Hancock-Beauliee MM (1998) OKAPI at TREC-7: ad hoc, filtering, vlc and interactive track. Text Retrieval Conference (TREC-7), Nov. 9-1 (Notebook), pp. 152–164.

Shaw JA and Fox EA (1994) Combination of multiple searches. In: Proceedings of the 2nd Text REtrieval Conference (TREC-2), National Institute of Standards and Technology Special Publication 500-215, pp. 243–252. citeseer.nj.nec.com/fox94combination.html

Turtle H and Croft WB (1990) Inference networks for document retrieval. In: Vidick J-L, Ed., Proceedings of the 13th International Conference on Research and Development in Information Retrieval, Association for Computing Machinery, ACM, New York, pp. 1–24.

Wilkinson R (1994) Effective retrieval of structured documents. In: SIGIR '94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, Dublin, Ireland, July 3–6, pp. 311–317.