PAAVO ARVOLA

The Role of Context in Matching and Evaluation of XML Information Retrieval

■

ACADEMIC DISSERTATION
To be presented, with the permission of
the board of the School of Information Sciences
of the University of Tampere,
for public discussion in the Auditorium Pinni B 1100,
Kanslerinrinne 1, Tampere, on June 18th, 2011, at 12 noon.

UNIVERSITY OF TAMPERE

UNIVERSITY
OF TAMPERE

ACADEMIC DISSERTATION
University of Tampere
School of Information Sciences
Finland

Distribution
Bookshop TAJU
P.O. Box 617
33014 University of Tampere
Finland

Tel. +358 40 190 9800
Fax +358 3 3551 7685
taju@uta.fi
www.uta.fi/taju
http://granum.uta.fi

Cover design by
Mikko Reinikka

Acta Universitatis Tamperensis 1624
ISBN 978-951-44-8474-2 (print)
ISSN-L 1455-1616
ISSN 1455-1616

Acta Electronica Universitatis Tamperensis 1085
ISBN 978-951-44-8475-9 (pdf)
ISSN 1456-954X
http://acta.uta.fi

Tampereen Yliopistopaino Oy – Juvenes Print
Tampere 2011

Acknowledgements

There are two people I want to thank above all. They are my beloved supervisors: Professor Jaana Kekäläinen and Associate professor Marko Junkkari, in random order. They have had a crucial impact on the content, quality and even the existence of the present dissertation. The mixture of the expertises and personalities of them, myself and the whole TRIX (Tampere Retrieval and Indexing for XML) group has enabled an innovative environment. From the TRIX –group I would give additional thanks to Timo Aalto for his co-operation and Johanna Vainio for making the measures of the project available for a wider audience.

Finnish Information Retrieval Experts (FIRE) group, the leader Kalervo Järvelin among many others, have provided me with their knowledge and visions. In this group, special thanks go to Feza Baskaya who has given me technical advice and with whom I have had endless technical and other discussions. In addition, I wish to thank Heikki Keskustalo, Sanna Kumpulainen and Saila Huuskonen for their peer support. Turkka Näppilä and Janne Jämsen deserve to be granted as my inspiring research colleagues.

The institute, staff and administration of the School of Information Sciences (SIS), and its predecessor Department of Information Studies have provided me with all the support and facilities needed. Apart that, during the course I worked seven months at the Department of Computer Science, which also taught me the technical skills required for my research.

I am very grateful for my funding which came from the Academy of Finland for the most part, apart from SIS and its predecessor. In addition, NORSLIS (the Nordic School of Library and Information Studies) funded some of my conference trips.

Tampere 13.5.2011

Paavo Arvola

Abstract

This dissertation addresses focused retrieval, especially its sub-concept XML (eXtensible Mark-up Language) information retrieval (XML IR). In XML IR, the retrievable units are either individual elements, or sets of elements grouped together typically by a document. These units are ranked according to their estimated relevance by an XML IR system. In traditional information retrieval, the retrievable unit is an atomic document. Due to this atomicity, many core characteristics of such document retrieval paradigm are not appropriate for XML IR. Of these characteristics, this dissertation explores element indexing, scoring and evaluation methods which form two main themes:

1. Element indexing, scoring, and contextualization
2. Focused retrieval evaluation

To investigate the first theme, an XML IR system based on structural indices is constructed. The structural indices offer analyzing power for studying element hierarchies. The main finding in the system development is the utilization of surrounding elements as supplementary evidence in element scoring. This method is called contextualization, for which we distinguish three models: vertical, horizontal and ad hoc contextualizations. The models are tested with the tools provided by (or derived from) the Initiative for the Evaluation of XML retrieval (INEX). The results indicate that the evidence from element surroundings improves the scoring effectiveness of XML retrieval.

The second theme entails a task where the retrievable elements are grouped by a document. The aim of this theme is to create methods measuring XML IR effectiveness in a credible fashion in a laboratory environment. The credibility is pursued by assuming the chronological reading order of a user together with a point where the user becomes frustrated after reading a certain amount of non-relevant material. Novel metrics are created based on these assumptions. The relative rankings of systems measured with the metrics differ from those delivered by contemporary metrics. In addition, the focused retrieval strategies benefit from the novel metrics over traditional full document retrieval.

Table of Contents

List of Original Publications

This dissertation consists of a summary and the following original research publications:

I. Jaana Kekäläinen, Marko Junkkari, Paavo Arvola, Timo Aalto (2005) TRIX 2004 - struggling with the overlap. In *Proceedings of the 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Lecture Notes in Computer Science 3493*, Springer-Verlag Berlin Heidelberg, 127-139.

II. Paavo Arvola, Marko Junkkari, Jaana Kekäläinen (2005) Generalized contextualization method for XML information retrieval. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management, CIKM 2005,* 20-27.

III. Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2008) The effect of contextualization at different granularity levels in content-oriented xml retrieval. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008,* 1491-1492.

IV. Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2011) Contextualization Models for XML Retrieval, *Information Processing & Management, Article in Press* doi:10.1016/j.ipm.2011.02.006, Elsevier, 15 pages.

V. Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2006) Query evaluation with structural indices. In *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977*, Springer-Verlag Berlin Heidelberg, 134-145.

VI. Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2010) Expected Reading Effort in Focused IR Evaluation, *Information Retrieval*, Volume 13, Number 5, Springer Science+Business Media, 460-484.

VII. Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2010) Focused access to sparsely and densely relevant documents. In *Proceedings of the 33rd International ACM Conference on Research and Development in Information Retrieval, SIGIR 2010*, 781-782.

These publications are referred to as Studies I-VII in the summary.
Reprinted by permission of the publishers.

1. Introduction

The discipline of information retrieval (IR) seeks means to find relevant information to fulfil user's information need (Baeza-Yates and Ribeiro-Neto 1999). IR is sometimes defined as an area devoted to finding relevant documents (e.g. Grossman and Frieder 2004). This definition is not accurate, since a relevant document is not always completely relevant; instead, the relevant information may be embedded somewhere in the document. Thus, the definition of finding relevant documents applies rather to *document retrieval* than to information retrieval, which refers to finding relevant information *from* the documents. Document retrieval leaves the latter task to the end user, whereas *focused retrieval* endeavors to remove this task by providing direct access to the relevant passages. Within the present dissertation, a system for focussed retrieval is constructed and evaluation methods of such systems are developed.

In some broad definitions, IR covers searching data in structured databases (structured data) as well. A narrower definition, found in Manning and others (2008), takes information retrieval to be finding relevant material of an *unstructured nature* (typically text). The notion of unstructuredness in the retrieved material refers to the distinction between structured data in the databases and unstructured text in the documents, considering the latter to be the focus of information retrieval. However, defining text documents to be of unstructured nature is ambiguous, since many text documents, such as newspaper articles or books, have a structure consisting of parts like titles, paragraphs and sections. Retrieving such parts below (and including) the document level forms the focus of the present dissertation.

Document parts, referred to as *elements*, have both a hierarchical and a sequential relationship with each other. The hierarchical relationship is a partial order of the elements (Skiena 1990), which can be represented with a directed acyclic graph, or more precisely, a tree. In the hierarchy of a document, the upper elements form the *context* of the lower ones. In addition to the hierarchical order, the sequential relationship corresponds to the order of the running text. From this

perspective, the context covers the surroundings of an element. It is worth mentioning that an implicit chronological order of a document's text is formed, when the document is read by a user.

For digital storage and manipulation, the structure of a document is often presented using a mark-up language, which is a formal language describing the structure of the text and the way it is presented with meta information (Witt and Metzing 2010). The purpose of a mark-up language is to syntactically distinguish the mark-up notation from the body text. In other words, the logical structure of the text is separated from the content.

Many contemporary mark-up languages are based on XML[1] (eXtensible Mark-up Language) (Bray et al. 1998), which has become the de facto standard for the document mark-up. Actually, XML is a metalanguage describing other mark-up languages (Harold and Means 2004). These include, among numerous others, DocBook[2] used for technical documentation and XHTML[3] (eXtensible Hypertext Mark-up Language) for the web.

In the perspective of IR, XML mark-up specifies the retrievable units, i.e. elements, and forms a hierarchy among them. Therefore the combination of IR and XML, XML IR, is beneficial in providing more focused answers to a user's information needs by returning elements as answers instead of full documents. Without any mark-up, the retrievable units are arbitrary passages in the focused retrieval context. XML serves various users and use cases ranging from data management and exact querying to information retrieval with vague queries (Lehtonen et al. 2007). The syntax of XML is introduced in detail in Section 2.

It is worth noting that beside textual documents, XML is also used for marking up structured data, the approach of which is similar to the database like data storage. Therefore, the notion of XML IR can be approached on the basis on two main use cases of XML, which according to Goldfarb and Prescod (1988) are:

1) Data-oriented XML
2) Document-oriented XML

[1] http://www.w3.org/XML/
[2] http://www.docbook.org/
[3] http://www.w3.org/TR/xhtml1/

The data-oriented XML use case covers XML as an interchange format or a format for data storage. In storing XML in an especially designed database for XML structures (*native XML database*), the data can be easily queried, exported and serialized into a desired format in an efficient fashion. As a data exchange format, XML satisfies the need for a common language for computers, where one computer sends messages to another.

A data-oriented XML element or a fragment[4] corresponds to, for example, a nested table from an accounting report, whereas documented-oriented XML is more likely characterized by free text elements such as paragraphs, headers, lists, and style formatting. The document-oriented XML use case is intended for publishing the content in various media for humans to read. The elements are of coarser granularity and the document has a more irregular structure in the document-oriented approach (Elmasri and Navathe 2004). In addition, the sequential order of elements plays a role. Document-oriented XML typically has an even more complex hierarchical structure than data-oriented XML and is hardly suitable for traditional structured data storage such as in relational databases. Instead, native XML databases are more appropriate.

The distinction between data- and document-oriented XML use cases is highly contractual and has not been thoroughly studied, but in industrial discussions there have been some quantitative approaches to distinguish these types[5]. In practice many XML collections share some qualities of both use cases, and instead of the dichotomic classification, one should discuss the degree of document-orientation. For instance, many documents contain metadata, such as year of publication and the names and affiliations of authors, which are considered data-oriented elements.

[4] "A general term to refer to part of an XML document, plus possibly some extra information, that may be useful to use and interchange in the absence of the rest of the XML document." source: http://www.w3.org/TR/xml-fragment.html

[5] http://lists.xml.org/archives/xml-dev/200406/msg00022.html 25.10.2010

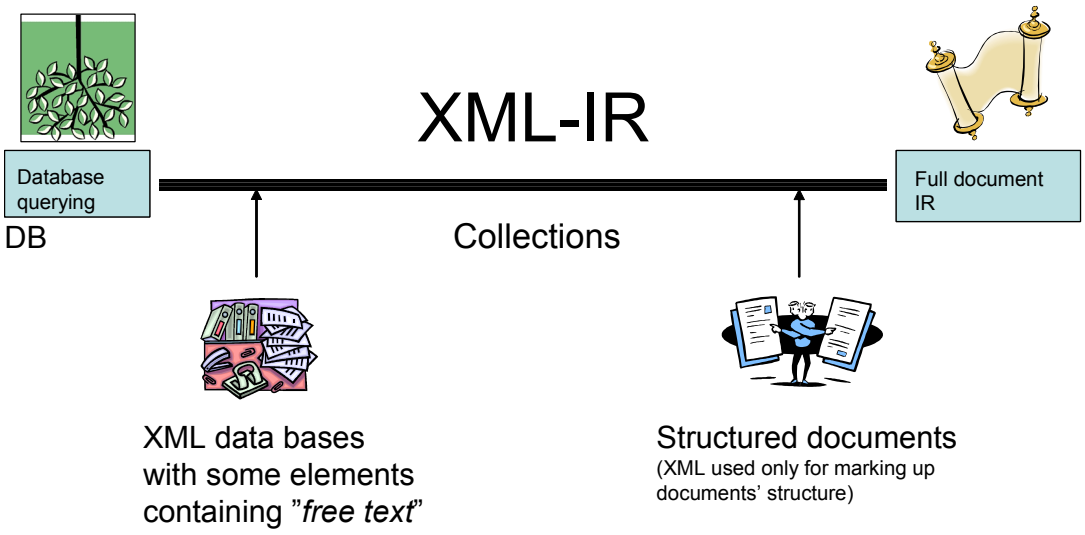

Figure 1. The scope of XML IR

Figure 1 presents the scope of XML IR. It falls between (and includes) full document IR and XML database manipulation and it is said that XML functions as a bridge between IR and databases (e.g. Luk et al. 2002). XML IR is applicable to XML collections[6] to some extent regardless of the degree of document orientation. In the XML database queries it enriches highly structured queries with IR primitives. At the full document retrieval end, the retrievable unit is an atomic document having the best matching elements highlighted. Grouping result elements by their document (ancestor) is called *fetch and browse* result organizing strategy (Chiaramella 2001).

XML IR has a relatively long history in IR research. An essential part of the research carried out in XML IR has been done at the yearly INEX (Initiative of the Evaluation of XML) workshop since 2002 (Fuhr et al. 2002). Analogically, as TREC (Text Retrieval Conference) (Voorhees and Harman 2005) is (mainly) for full document retrieval, INEX is for the evaluation of XML retrieval. In short, the initiative offers a collection of documents, with requests and corresponding relevance assessments, as well as various evaluation metrics for XML IR

The issues of the present dissertation are closely related to the work accomplished within INEX and the dissertation aims to contribute to the field of XML retrieval as well. The dissertation consists of an introductory part with six

[6] An XML collection is a predefined set of XML documents.

sections followed by seven separate articles, which all are summed up with the following two main themes:

I. Constructing an XML retrieval system and developing XML retrieval methods by using the system.
II. Developing focused retrieval evaluation methodology

The primary contribution within the first theme is the concept of *contextualization* (Arvola et al. 2005, Kekäläinen et al. 2009, II, III, IV), which is an XML retrieval method where the context of an element is taken into account as extra evidence in scoring individual elements. In order to study contextualization, and other XML retrieval methods, an XML IR system called TRIX (Tampere Retrieval and Indexing for XML) is constructed (I). The matching method of TRIX is based on structural indices and BM25 (Robertson et al. 2004, Sparck Jones et al. 2000) retrieval model, which are both introduced in Section 3. With the methods described, the TRIX system has yielded top rankings in comparison to the runs provided by the other INEX participants. The comparison and measurement of the methods was made with the metrics and test collections provided by INEX. These are introduced in Section 4. The evaluation metrics for calculating the fetch and browse strategy were developed further within the second theme.

The evaluation metrics developed for the second theme calculate user effort within a single result document. The calculation of the effort is based on a simulated user scenario – a *what if* model, where the chronological order of read passages in the document is assumed, likewise the amount of text read (Arvola 2008). In the model the chronological reading order depends on the natural sequential order of the document and the retrieved set of elements. The novel metrics based on the model include character precision-recall and cumulated effort. One of the measures developed (T2I@300) is used in the official system evaluations of INEX 2010. The background of the evaluation for XML retrieval using the fetch and browse result organizing strategy is given in Section 4.4.3. The themes are specified in the separate articles of the present dissertation and in the summary in Section 5. Section 6 concludes the dissertation.

2. Structured Documents

2.1 Document Structures

Data in computer science is roughly divided into three categories according to their degree of explicit structure; namely structured, unstructured and semi-structured data (Elmasri and Navathe 2004, Pal 2006). Unstructured data refers to items containing plain unannotated text or other unstructured items such as figures and videos, whereas structured data refers to data using a predefined strict format (database schema), for instance records in a relational database represent this kind of data. Semi-structured data (Abiteboul 1997, Buneman 1997) falls in between these two. In comparison to structured data, semi-structured data is irregular by its structure and it allows a lot of missing values. In structured data, such as a relational database, a NULL value is used instead. Semi-structured data is by definition schema-less, but the mark-up used is defined to be self-describing and possesses some structure.

Surprisingly, documents following the semi-structured data model are called structured documents. They have an explicit structure which separates them from unstructured documents, but their information organization is not as rigid as the records in a database, for instance. Accordingly, as a structured document forms a hierarchy, the data model of a structured document is a rooted ordered tree. In other words, a text document has a physical structure which is both hierarchical and sequential.

Characters form the most specific units in Latin (or equivalent) based scripts, and from the hierarchical perspective words consist of characters being the lowest meaningful atomic units. Phrases and sentences, in turn, consist of words. A document's text is typically organized in logical parts, such as chapters containing sections containing sub-sections, paragraphs and so forth.

Apart from the explicit (syntactic) document structure, any meaningful text also has implicit (semantic) content structures. According to Meyer (1985),

description, sequence, compare/contrast, cause/effect, and problem/solution are the most commonly found in such text structures. For example, a book about the Second World War may have passages that compare how the sea battles in Europe were different from the battles in the Pacific or an explanation of what led to the war in the first place. Obviously these text structures may exceed physical document parts, so that, for instance, the problem and solution of a subject may be present in the first and last paragraphs of a document. However, a document mark-up is typically used for explicit document structures only, and the implicit contentual structures remain unmarked.

2.2 XML Mark-Up

The document structures and different document parts need to be indicated somehow. Typically, this is achieved by a mark-up language, which in the present dissertation is XML. The predecessor of XML, SGML (Standard Generalized Markup Language) (SGML 1986), was used for storing reference works that were to be published in multiple media. Indeed, SGML was used mostly in document-oriented fashion. Because of its roots, one very common usage of XML is explicitly marking-up the document structure.

Actually, the XML language is a meta-language for defining mark-up (i.e. encoding) languages. As the name XML (eXtensible Mark-up Language) suggests, the language is extensible; it does not contain a fixed set of tags. In this section, we introduce the mark-up very cursory, i.e. in a magnitude that is needed for the comprehension of the present dissertation[7]. Actually, the mark-up is a mere technical detail. Namely, many other hierarchical presentations of data, such as hierarchical databases, could be used instead. Nevertheless, for the purposes of the present dissertation the essential features of XML are the hierarchical and sequential structure and to some extent the element names play a role. Sequential structure means that the items are in some linear order, which in structured documents and in XML terminology is called the document order. In other words, the elements retain

[7] The present dissertation focuses on a schema-less manipulation of XML data, thus DTD's and XMLSchemas are left out.

the order in which they appear in the document (Holzner 2004). This is, of course, not necessarily the chronological order the user follows when reading a document.

Below is an example of XML mark-up. It represents a simple XML document, where the elements are named according to their occurrence in document order (i.e. depth first order). XML elements are in and between start and end tags, denoted by an element (tag) name surrounded by angle brackets. The end tag starts with a slash. In addition to the name, the start tag may also include attribute value pairs. In the example the element *one* has a *description* attribute with the string value "this is the root element". Any XML document should be well formed, so that it contains a single root element, which contains (possible) other elements. In addition, an element may contain text or other elements or attributes. If an element contains other elements they form a hierarchy, otherwise the element is a leaf element.

```
<one description="this is the root element">
    <two>This is a leaf element.</two>
    <three>
        <four>This is also a leaf element.</four>
        <five my_attribute="This is the context element in Figure 2">
            <six>
                <seven>This is a leaf element.</seven>
            </six>
              Text content can be between any tags.
        </five>
        <eight>This is a leaf element.</eight>
     </three>
     <nine>This is a leaf element.</nine>
 </one>
```

As XML is a widely used standard for representing data, there are a number of methods for manipulating and querying it. A standard and probably the most utilized query language for XML is XPath (Clark and Derose 1999), which is introduced briefly next.

2.3 XPath and Structural Relationships of Elements

Within an XML document, one element has a positional relationship to another. Figure 2 represents an XML tree, where the relationships to element with id 5 are shown (it is the context element[8]), and some other relationships are defined with

[8] disambiguation note: here the context element refers to the point from which the relationships are defined, i.e. the starting point

XPath. The document order of the XML elements corresponds to the depth first order of a rooted tree (i.e. 1, 2, 3, 4, 5, 6, 7, 8, 9).

XPath is the standard basic query language for XML and is defined only through examples in the present dissertation. XPath is based on path expressions, where the elements are represented with their name or a star (*) denoting any element name. A path is basically a linear expression transferring from one element to another step by step through other elements and the query itself forms a tree. Informally, an (abbreviated) path expression is a sequence of steps separated by the slash "/" operator, for example /one/three/four yields element *four*, with *three* as the parent and *one* as the parent element of *three*. The slash operator denotes an immediate ancestor and a double slash "//" denotes any ancestor. For treelike querying XPath offers a notation between square brackets, which are branches in the paths, for example /one/three[.//foo]/four is interpreted as that there should be a *foo* element under the element *three* (which is not true in the example of Figure 2). The path may start from the root as an absolute path starting with the slash, or from a predefined, context element starting with a dot (i.e. self).

relation	element ids	XPath expr (abbrev)
self	5	.
child	6	./*
descendant	6, 7	.//*
sibling	4, 8	./sibling::*
following sibling	8	./following-sibling::*
preceding sibling	4	./preceding-sibling::*
following	8, 9	./following::*
preceding	2, 4	./preceding::*
parent	3	./..
ancestor	1, 3	.//..
context	1, 2, 3, 4, 8, 9	

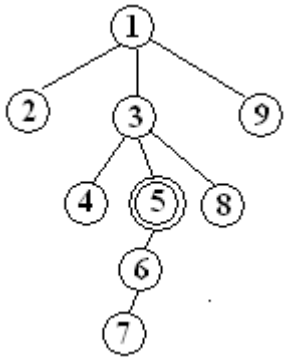

Figure 2. An XML structure, its tree representation and the relationships between element #5 and other elements

In XPath the relationships between the elements may be chained so that, for instance, the query /one//three/four/following-sibling::* yields any element, which

have *four* as the preceding sibling and *three* as the parent and *one* as the root element. Thus, the query yields elements *five* and *eight.*

In addition to presenting the standard relationships between elements, we define context, which covers everything in the document excluding descendants and self. Two general types of context can be distinguished based on the standard relationships. Vertical context, for one, refers to the ancestors, whereas horizontal refers to the preceding and following elements. It is worth noting that the ancestor elements contain the self element, i.e. the element *five* in the example is a part of elements *one* and *three*. The feature of elements being part of and containing other elements is called overlapping. In the example, the members of the sets (branches) {1, 2}, {1, 3, 4}, {1, 3, 5, 6, 7}, {1, 3, 8} and {1, 9} overlap with the other members of the same set.

3. Indexing and Retrieval in XML IR

An XML collection needs to be *indexed* in order to perform efficient searches. In XML retrieval indexing refers to both giving a label to the element and constructing an inverted file based on a mapping of keys and these labels. We call the former indexing the XML structure, and the latter indexing the content.

A basic way of indexing the structure is to use the path as the label (e.g. Geva 2006). In it, the element is given a label according to its absolute path, for instance, the path /article[1]/sec[2]/p[4] refers to the fourth paragraph of the second section in an article. Obviously, this kind of indexing enables a straightforward processing of XPath queries, but it is not (space) efficient, especially in processing queries for content only. A more efficient solution is to label the elements using numbers. For example, the structure of the XML document in Figure 2 is indexed according to the depth first order. This kind of indexing is called global indexing (Tatarinov et al. 2001), as it considers the whole structure of an XML document. In the XML hierarchy the elements are not independent of each other and the hierarchical and sequential relationship between each other needs to be maintained in some fashion. The present dissertation uses Dewey labels, also known as *structural indices*, because they serve the purposes of manipulating and analyzing complex XML structures. In Section 3.1, we focus on labeling the retrievable units (i.e. indexing the structure) and in Section 3.2 we focus on indexing the content based on the labels.

3.1 Dewey Labeling Scheme

A Dewey label is a chain of integers each denoting a position among the child list of an element. The child position is called also the local index (Tatarinov et al. 2002). The Dewey labeling considers the whole structure and combines global and local indexing. Figure 3 represents a tree model of an XML document with Dewey labels.

For instance, one can deduce the parent of an element labeled with ***1.2.2*** by removing the integer after the last dot, thus obtaining ***1.2***. Thus, it is trivial to recognize whether elements are overlapping according to the label. In addition to analyzing the hierarchical structure, Dewey labeling also works well in deducing the (exact) preceding and following element relationship between known indices. For example, it is trivial to say that an element labeled ***1.1.2*** is an immediate follower of the element ***1.1.1***. That means the exact sequential (document, reading) order is preserved.

Since the Dewey numbering system, when applied to a general hierarchical structure, does not follow the decimal system, the most convenient way of formal manipulation of the labels is to represent them as a tuple according to Niemi (1983). Thus, hereafter in this study we use set theoretical primitives for Dewey labels and interpret the label as a tuple so that label ***1.2.3*** is represented as ⟨1, 2, 3⟩.

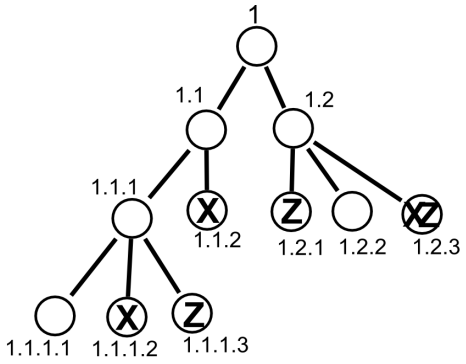

Figure 3. An XML tree with Dewey labels and marked elements having features X and Z

It is worth mentioning that in addition to XML, the Dewey labeling scheme or equivalent has been used for hierarchical data model (Niemi 1983), nf2 –relational data model (Niemi and Järvelin 1996), managing part-of relationships (Junkkari 2005) and data cube construction (Näppilä et al. 2008) as well.

Despite the popularity of the Dewey scheme, it has some disadvantages. First, the length of the storage space for paths from the root to each element varies and is long in deep structures (Tatarinov et al. 2002). Second, updating the XML structure may become costly in modifying the Dewey labels (Härder et al. 2007, O'Neil et al. 2004, Tatarinov et al. 2002). However, due to the good update qualities in the developed models of the Dewey labels, they have gained popularity in both academia and industry. Probably the most famous approach based on Dewey labeling is ORDPATHs by Microsoft (O'Neil et al. 2004), which presents a variant

of the Dewey labeling scheme. However, ORDPATHs can be criticized, since although they maintain the document order, the labeling is coarser grained than plain Dewey. This is because the adjacent (following or preceding) siblings cannot be deduced and thus the distance between two specific elements remains incalculable. Dewey maintains the exact document order.

Other similar prefix based labeling schemes not based directly on Dewey include e.g. LSDX (Duong and Chang 2005), FLEX (Deschler and Rundensteiner 2003) and a scheme introduced by Cohen and others (2002). These schemes share the same problem of at least the same magnitude as ORDPATHs (Böhme and Rahm 2004) when it comes to retaining the exact sequential order and thus deducing the distance between certain two specific elements.

Apart from these prefix based schemes, other fair labeling schemes also exist. For example, Li and Moon (2001, see also Agrawal et al. 1998) uses the pair of preorder and postorder numbers for each element. In this, given any two elements, it can be determined whether one of the two elements is an ancestor of the other in constant time. This kind of indexing is in use e.g. in the TopX XML IR system (Theobald et al. 2005).

Consequently, Dewey labels have more analyzing power in determining the complex ancestor relationship than most of the previously mentioned competitors (Christophides et al. 2003). With Dewey labels it is easy to discover which elements belong to the context of an element. With this quality in mind, constructing an inverted index containing ancestors is also straightforward as the following sections show.

3.2 Inverted Index

Actually, it is meant for finding records in a database for which the values of fields have been specified (e.g. Knuth 1997, 560–563). A typical application for an inverted index is the full-text search where for each key a list of key occurrences (i.e. labels) is given in the inverted index. In other words, an inverted index is defined as a set of key - occurrences pairs, where the occurrences indicate the locations of the keys. It is worth mentioning that an inverted index is widely utilized method in query processing because its efficiency (Zobel et al. 1998).

Record label	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
occur X and Z				X	Z	X		Z		ZX		X		Z			ZX			Z

Figure 4. Database with fields containing X and Z are marked. (The opening part of the database resembles the pre-order numbering of the tree in Figure 3)

In Figure 4 there is a sample flat database where the occurrences of the items X and Z are marked. The key X occurs in the following set of locations: set_X={4, 6, 10, 12, 17} and similarly key Z in: set_Z={5, 8, 10, 14, 17, 20}. The inverted index containing X and Z can be represented as: *IF*={⟨X, {4, 6, 10, 12, 17}⟩, ⟨Z, {5, 8, 10, 14, 17, 20}⟩}. Now, the co-occurrences of X and Z can be calculated trivially as an intersection X AND Z: $set_X \cap set_Z$={10, 17} as well as the occurrences where either of the keys occur as an union X OR Z: $set_X \cup set_Z$={4, 5, 6, 8, 10, 12, 14, 17, 20}.

An inverted index based on Dewey labels can be constructed similarly to the approach above with the difference that the occurrences are now presented as Dewey labels. For example in Figure 3 the inverted index containing X and Z would be IF_{dewey}={⟨X, {⟨1, 1, 1, 2⟩, ⟨1, 1, 2⟩, ⟨1, 2, 3⟩}⟩, ⟨Z, {⟨1, 1, 1, 3⟩, ⟨1, 2, 1⟩, ⟨1, 2, 3⟩}⟩}. It is worth noting that only the lowest hierarchy level of occurrences needs to be stored, because the Dewey labeling enables upper hierarchy levels to be deduced. When also considering the ancestors the same inverted index would then be $IF_{implicit\_dewey}$={⟨X, {⟨1⟩, ⟨1, 1⟩, ⟨1, 1, 1⟩, ⟨1, 1, 1, 2⟩, ⟨1, 1, 2⟩, ⟨1, 2⟩, ⟨1, 2, 3⟩}⟩, ⟨Z, {⟨1⟩, ⟨1, 1⟩, ⟨1, 1, 1⟩, ⟨1, 1, 1, 3⟩, ⟨1, 2⟩, ⟨1, 2, 1⟩, ⟨1, 2, 3⟩}⟩. The standard operations AND and OR can be implemented trivially as an intersection and union respectively.

3.3 Retrieval

Due to the popularity of the Dewey labeling scheme, there have been numerous proposals where Dewey labels have been utilized for various types of querying (e.g. Lu et al. 2006, Theobald and Weikum 2002, Arvola 2007). As in full document retrieval, in XML retrieval, the retrieval models are divided roughly into exact and partial match models.

3.3.1 Matching in XML

Exact match in traditional IR is executed using standard set operations (union and intersection). As examples of the exact match in XML one could mention finding the lowest common ancestor (LCA) or smallest lowest common ancestor (SLCA) (Christophides et al. 2003, Guo et al. 2003). They correspond to the AND semantics of traditional Boolean queries (Sun et al. 2007).

Roughly speaking, in full document IR, the partial match models constitutes two families: vector space (Salton et al. 1975), and probabilistic models (see Baeza-Yates and Ribeiro-Neto 1999). In full document IR, the partial match model is based on the statistical occurrences of the query keys (i.e. terms) in a document. In other words, the query keys are matched against the documents and a combined score over all query keys is calculated for each document. According to the combined score, the documents are organized in descending order. Both retrieval models, however, do not include document structure and only flat queries are supported.

One of the most effective XML IR systems in the early years of INEX, JuruXML by IBM Haifa labs, is based on a modification of the vector space model (e.g. Carmel et al. 2003). A special and theoretically intriguing case in the probabilistic family is the language model where a document is a good match to a query if the document model is likely to generate the query, which will occur if the document contains the query words often. The modifications for XML are provided, for instance, by the systems of Carnegie Mellon University (e.g. Ogilvie and Callan 2006) and the University of Amsterdam (e.g. Sigurbjörnsson and Kamps 2006).

Various other retrieval models have also been applied to XML IR, but here we focus on one of the most popular (also in INEX), BM25 (Robertson et al. 2004), the modification of which is also applied in the present dissertation (I). In BM25, the basic components for key weighting are:

1. key frequency (kf_k), i.e. how many times the query key (k) occurs in the document
2. inverted document frequency (idf_k), i.e. what is the prevalence of the query key in the whole collection, and
3. document length normalization. That is, the density of the key occurrences is taken into account as well.

The final score for a document is typically a sum of the key weights. In BM25, the weight for a key in a document is calculated as follows:

$$w_k = \frac{q_k \cdot kf_k}{kf_k + a \cdot \left(b + (1-b) \cdot \frac{dl}{avgdl} \right)} \cdot \log\left(\frac{N - n + 0.5}{n + 0.5} \right) \qquad (1)$$

In the formula the latter part of the multiplication is the *idf* part, where N is the total frequency of all documents in the collection and n the frequency of the documents containing key k. The document length dl is normalized with the average document length $avgdl$. Finally, a and b are parameters for tuning the roles of individual components in the formula. The q_k indicates the weight the key has in the query, for instance if it has a plus sign.

In XML retrieval, instead of the full documents, the elements are given a score. This makes a difference in matching and formulas such as the BM25, do not suit well for overlapping elements, because they would favor large elements over their descendants. This is because the key frequency is always the same or greater in the ancestors of an element. On the other hand, interpreting the document length component as the element length compensates this aspect. A recent study by Itakura and Clarke (2010) introduced probably the most straightforward way to apply the BM25 formula for XML retrieval. They calculated a score for individual XML elements as if they were documents and filtered the resulting ranked list to remove overlap. Other systems using BM25 for XML retrieval include (Lu et al. 2006), Gardens Point XML (GPX) (Geva 2006) and the TopX system (e.g. Theobald et al. 2005).

In a partial match system, the weight is allocated to an element according to the key. In the following, an extended version of the inverted index from Section 3.2 is given with related weights. The following example illustrates only the matching. The calculation in the example is very basic: key frequency divided by the number of descendant elements plus self (rounded to two digits).

$IF_{partial\_dewey}$ ={⟨X, {⟨⟨1⟩, 0.27 ⟩, ⟨⟨1, 1⟩, 0.33⟩, ⟨⟨1, 1, 1⟩, 0.25⟩, ⟨⟨1, 1, 1, 2⟩, 1⟩,
⟨⟨1, 1, 2⟩, 1⟩, ⟨⟨1,2⟩, 0.25⟩, ⟨⟨1, 2, 3⟩, 1⟩}⟩, ⟨Z, {⟨⟨1⟩, 0.27⟩, ⟨⟨1, 1⟩, 0.17⟩,
⟨⟨1, 1, 1⟩, 0.25⟩, ⟨⟨1, 1, 1, 3⟩, 1⟩, ⟨⟨1, 2⟩, 0.5⟩, ⟨⟨1, 2, 1⟩, 1⟩, ⟨⟨1, 2, 3⟩, 1⟩}⟩}

Now the query “X Z” and the final score calculated as the sum of weights, delivers the following sorted result list:

⟨“X Z” ,⟨⟨1, 2, 3⟩, 2⟩, ⟨⟨1, 1, 1, 2⟩, 1⟩, ⟨⟨1, 1, 2⟩, 1⟩, ⟨1, 1, 1, 3⟩, 1⟩ ,

⟨⟨1, 2, 1⟩, 1⟩, ⟨⟨1,2⟩, 0.75⟩, ⟨⟨1⟩, 0.54 ⟩, ⟨⟨1, 1⟩, 0.5⟩, ⟨⟨1, 1, 1⟩, 0.5⟩⟩⟩.

When reading the result list, the user may feel uncomfortable, because the same content appears several times in the results. In other words, the result list contains overlapping elements. Therefore the result ought to be organized.

3.3.2 Result Organizing Strategies

The XML IR can be divided into three strategies in relation to result organization (Kamps 2009):

- Thorough
- Focused[9]
- Fetch and Browse

In the thorough strategy elements in the result list may be overlapping, as in the previous example, whereas in the focused strategy the elements in the result list should not overlap. In the non-overlapping result list, only one of the overlapping elements should be selected. A typical strategy is to select the elements on the basis of their scores so that among overlapping elements the element with the best score is selected. Following the example in the previous section, the result list following the focused strategy would be:

⟨⟨⟨1, 2, 3⟩, 2⟩, ⟨⟨1, 1, 1, 2⟩, 1⟩, ⟨⟨1, 1, 2⟩, 1⟩, ⟨1, 1, 1, 3⟩, 1⟩ , ⟨⟨1, 2, 1⟩, 1⟩⟩⟩

The common denominator in focused and thorough strategies is that the elements are considered as individual instances, i.e. they are not grouped in any way. In the fetch and browse strategy (Chiaramella 2001) the returned elements are grouped by a grouping element, which in the present dissertation is always the whole document (i.e. the root). Finding the best entry point for each document is considered a special case of this strategy. In it, only one element or spot per document is retrieved. Otherwise the quality of selection in fetch and browse retrieval means in practice selecting an appropriate amount of material from a relevant document.

Within INEX, several tasks have been proposed, tested and discussed for being representative use cases of XML IR (Lehtonen et al. 2007, Trotman et al.

[9] Disambiguation note: The focused result organizing strategy is different than focused retrieval as a retrieval paradigm.

2007). In 2008, two different use cases were modeled: "In focused retrieval the user prefers a single element that is relevant to the query even though it may contain some non-specific content (elements must not overlap). With in-Context retrieval, the user is interested in elements within highly relevant articles - they want to see what parts of the document will best satisfy their information need."[10] In the first mentioned task a flat, non-overlapping result list is required. A fetch and browse based result list presentation is needed to accomplish the latter task.

Regardless of the abovementioned strategies, the query may contain structural conditions and these conditions restrict the number of elements in the results. Next we introduce structured queries in XML.

3.4 Structured Queries in XML

The user conveys his or her information need to an IR system with a query. The query should be given in a language which is interpretable by the system. Many modern IR systems support a so-called bag-of-words query, where only the words describing the information need are written in a search box and these words are used by the system as keys in matching documents. These keywords occurring in a document are considered as evidence of the relevance of the document's content.

In content-only (CO) queries, the element type is obtained automatically. In addition the structural similarity between the query and retrievable elements can be incorporated. This is enabled by a more expressive querying with a structured query language. This is motivated by getting even more precise answers by explicitly defining the element type to be retrieved.

There are a number of query languages capable of manipulating XML IR. These query languages may contain keywords and structural constraints, or keywords only. In the latter case it is up to the system to decide what level of granularity best answers the user's query. Queries that contain conditions both for content and structure are called content and structure (i.e. CAS) queries (Malik et al. 2005). These queries are expressed with an XML IR query language.

[10] http://inex.is.informatik.uni-duisburg.de/2007/adhoc.html

XML IR query languages with a non-complex syntax for naïve users such as XSearch (Cohen et al. 2003) and other simple content only query languages are not sufficient to fulfil users' information needs once search tasks become more complex, (van Zwol et al. 2006). Therefore, more expressive power is required.

The syntax of manipulating structural relationships in XML IR query languages with expressive power is based on query languages meant for database and data-oriented querying, such as XPath (Clark and DeRose 1999) and XQuery (Boag et al. 2008). The fundamental advance in XML IR query languages is that while database languages look for exact matches, the XML IR query languages support relevance ranking of elements as well. In many cases, the relevance ranking is applied with language primitives embedded within a query language intended for data-oriented querying.

The relevance calculation is often implementation-dependent, as are in the most famous and novel extensions of XQuery and XPath, the XQuery-FT and XPath-FT recommended by the W3C (Amer-Yahia et al. 2008). Another example is NEXI, which is used within the INEX initiative (see Gövert and Kazai 2003, Fuhr et al. 2008).

NEXI is a query language which is a facilitated version of XPath with IR features. It was designed and found to be a fairly simple query language for both novices and experts (Trotman and Sigurbjörnsson 2005). The IR features include inexact keyword search, Boolean operations, and partial structural vagueness. NEXI leaves the interpretation of structure, operations and ranking open, and thus gives no strict semantics. Namely, NEXI contains constraints concerning both the content and the structure. In NEXI, the content constraints are expressed within *about* clauses, which are surrounded by structural constraints derived from the XPath language. A sample NEXI query can be represented as follows:

```
//article[.//abstract, about(″description logics″)]//sec//p[.,about("semantic networks")]
```

The intuitive interpretation of the above query is that *p* elements about semantic networks are sought. In addition, following XPath, the *p* element should be in a *sec* element, which in turn should be in an *article* element, which has an abstract about description logics.

One of the motivations of IR systems is to provide approximate answers to a query and deliver the result in descending order of estimated relevance. Clearly, in a partial match system the content constraints ought to be interpreted vaguely, but

there has been a trend to take the structural conditions merely as hints of the possible location of the content (Kamps et al. 2005). This is because the user may not be thoroughly familiar with the XML structure (O'Keefe and Trotman 2004). In INEX, making the content constraints vague is referred to as the VCAS strategy, V standing for vague, in contrast to SCAS, S standing for the strict interpretation of the structural constraints (Trotman and Lalmas 2006).

One special case of the vague interpretation of the constraints is to divide them into constraints concerning the target element and constraints concerning the source element e.g. the path. In the example above, the *p* element is the target element constraint and article//sec form the source element constraints. In INEX, the strategies interpreting the target and source element differently are referred to as VSCAS strategy having source element constraint interpreted vaguely and target element constraint strictly (Trotman and Lalmas 2006, Trotman 2009). SVCAS means that these constraints are interpreted vice versa. Accordingly, VVCAS and SSCAS, refer respectively to interpreting both of these constraints vaguely and strictly.

4. Measuring Effectiveness in XML IR

In this section, we discuss measuring the effectiveness of IR systems, in other words, how well an IR system responds to the user's information need (Manning et al. 2008). The evaluation of IR systems is divided roughly into two categories: interactive IR research and laboratory evaluations. Observing real users in real situations or a more controlled research exploring users performing assigned tasks pertains to the first category. This approach gives a good insight into system performance in interaction with an IR system. However, when studying, for instance, which values of the *a* and *b* parameters in the BM25 formula deliver the best results, this kind of research is very expensive and hard to replicate with different parameter combinations.

The laboratory evaluation aims to measure the objective quality of the results delivered by an IR system. This means that the real users are marked out from the actual study and use "objectively defined" right answers as the goal for an IR system to retrieve instead. This kind of approach is based on a collaborative effort within a laboratory model, which is next described in the XML context.

4.1 Laboratory Model in XML IR

Comparing and quantifying an IR system from the perspective of retrieving good quality results is done using a specific laboratory framework. This evaluation framework provides a method to compare the effectiveness of retrieval strategies. The main components of this framework consist of a collection of XML documents, requests and evaluation metrics. An XML collection typically consists of a fixed set of XML documents that is searched by the retrieval systems under evaluation. In addition, there is a set of user requests that the systems aim to satisfy. For every request there are relevance assessments, indicating which documents, or in focused

retrieval, document parts, satisfy a given request. Finally, evaluation metrics quantify the system performance.

Figure 5 represents the laboratory model of XML IR evaluation used in the present dissertation. It is a modification of the theoretical framework for system-oriented IR research presented by Ingwersen and Järvelin (2005). The basic components are the same; the dependent variable is the retrieval effectiveness which depends on the XML IR system and the evaluation components. The evaluation components consist of a set of elements as a collection and requests. The relevance of the elements in relation to each request is judged by human assessors and a relevance score is given to each element. These judged elements are compared with the results delivered by a retrieval system and an evaluation score is calculated based thereon. To simplify the diagram, indexing and searching are embedded in the XML IR system diamond. The retrievable elements are indexed and the query[11] is executed against the inverted index.

Depending on the search task, some of the elements are not to be retrieved, even if they are relevant by reason of their content and they need to be filtered out. This makes it possible to vary the retrievable element type, for example when measuring the strict CAS interpretation, e.g. retrieving abstracts only. In this case, only the pre-selected elements are allowed and the evaluation focuses on those elements as well. However, some elements not included in retrievable elements belong to the context of these elements and are utilized in contextualization (see Figure 5).

The outcome of the matching is a set of elements which overlap each other. In focused result organizing strategy, the retrieval system selects an element from the right granularity from each branch and discards all overlapping elements. In fetch and browse strategy the results are grouped by a grouping element, typically a document.

[11] A query is a representative for the request, which is given to the system.

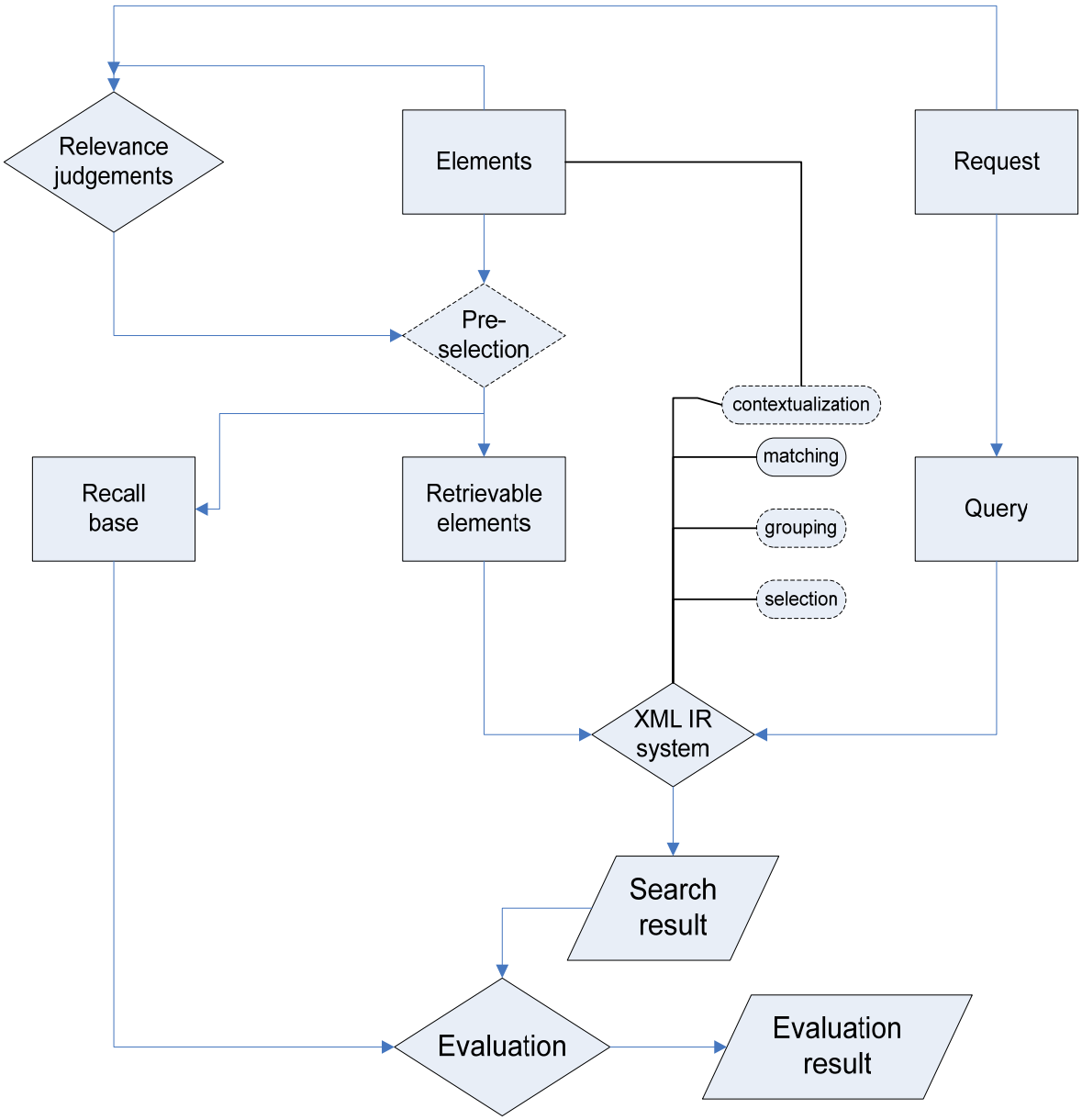

Figure 5. The laboratory model for XML IR

4.2 Topical Relevance and Specificity

Information retrieval is typically human-computer interaction, where the user explicates his or her request as a query and retrieves the documents as the result. There are a number of factors affecting the usefulness of a document within the results. The primary criterion is the topical relevance (Schultz 1970). This means a correspondence between the topic of interest and the "aboutness" of the document. In other words, is the document about the request? This aims to be an objective definition. However, in reality the usefulness is also dependent on the user's

personal attributes, such as whether the document is new to the user, and the situation the user is in (Saracevic 1996). In traditional information retrieval systems and matching models, the users' attributes and the retrieval situations are beyond the retrieval system's reach. Thus, these user and situational relevance dimensions are left out of the evaluations and measuring the retrieval effectiveness is based on the topical relevance only. This enables a simplified research setting, which can objectively be used in comparing various information retrieval systems and methods.

Measuring the effectiveness of XML retrieval with the standard IR measures is not meaningful if heterogeneous result lists are allowed. In a hierarchical structure, the relevance is upward inheritable, i.e. since the parent contains all the information of its descendants, the parent is at least as relevant as any of its descendants in relation to the request. Therefore, returning the elements of maximum coverage, (the largest elements) would be the best strategy for getting as much relevant material as possible. However, the aim of XML retrieval is to retrieve relevant elements which are at the right level of granularity. In other words, the elements should be answers to the request which are as focused as possible, still covering the request exhaustively. Roughly speaking, the evaluation of XML retrieval addresses the combination of *scoring quality*, that is, how well a system retrieves the relevant elements, and *selection quality*, which means the selection of an element of appropriate size.

Accordingly, Chiaramella (2001) introduces two concepts: exhaustivity and specificity to characterize relevance dimensions. For a document D and a request Q it holds that implications: $D \rightarrow Q$ characterizes the exhaustivity and $Q \rightarrow D$ characterizes the specificity of the document. In other words, for a perfect document, exhaustivity refers to the complete fulfilment of a request, while specificity means that *only* these constraints are fulfilled. In XML retrieval evaluations exhaustivity is defined as the extent to which the document component discusses and specificity the extent to which the document component focuses on the request.

4.3 The Initiative for the Evaluation of XML retrieval (INEX)

As with full document retrieval, evaluating the XML Retrieval effectiveness requires a document collection, topics, relevance assessments and metrics. Accordingly, Initiative for the Evaluation of XML retrieval (INEX) (see Gövert and Kazai 2004, Fuhr et al. 2002) with its yearly workshop, has been the forum for XML IR evaluation. In 2002, INEX started to address the evaluation of XML retrieval. An infrastructure was established, and a large XML test collection and appropriate evaluation metrics were provided for the evaluation of content-oriented XML retrieval systems.

In INEX, several tasks have been proposed, tested and discussed as representative use cases of XML IR. In the main tasks the querying is based on textual content i.e. content-only queries and on both textual content and XML structure (CAS queries) (Malik et al. 2005, Malik et al. 2006). Following result organizing strategies, these main tasks can be interpreted as retrieval and ranking of XML elements without context or within context. In the former, elements are returned as independent information units; in the latter, elements are returned within the containing document (i.e. fetch and browse) (Malik et al. 2005).

Accordingly, the Relevant-in-Context (RiC) and Best-in-Context (BiC) XML IR tasks evaluate the fetch and browse result organizing strategy, i.e. they aim at retrieving, ranking and returning relevant elements in the context of a document. The difference between these tasks is that in the Relevant-in-Context task multiple items within a document can be retrieved, whereas in the Best-in-Context task only the best entry point of a document is indicated. The Relevant-in-Context task is mentioned as the most credible task in the light of use cases (Lehtonen et al. 2007, Trotman et al. 2007).

4.3.1 Test Collections and Topics

The INEX test collection used until 2004 consists of 12,107 XML articles, from 12 magazines and 6 transactions of the IEEE Computer Society publications, covering the period of 1995-2002, and totalling 494 MB in size, and 8 millions elements. The collection contains scientific articles of varying length. On average, an article

contains 1,532 XML elements, where the average depth of the element is 6.9. In 2005, the collection was enlarged with 4,712 new articles from the period 2002 - 2005 of the IEEE Computer Society, in size to a total size of 764Mb, and circa 11 million elements. In 2006, the IEEE collection was replaced with the Wikipedia collection with XML mark-up (Denoyer and Gallinari 2006). This collection contains over 660,000 English documents and 4.6 gigabytes of data. The mark-up of the collection was made afterwards and many text passages remained unmarked. The collection provides a massive link structure and a category hierarchy to utilize in retrieval (see e.g. Jämsen et al. 2007). In 2009, the Wikipedia collection was again enlarged and a semantic mark-up was provided (Schenkel et al. 2007).

4.3.2 Exhaustivity and Specificity

During the years 2002-2005 of INEX, exhaustivity (exhaustiveness) and specificity dimensions were assessed for each element. In the assessments, exhaustivity (E) describes the extent to which the document component discusses the request. Specificity (S) describes the extent to which the document component (element) focuses on the request. Each of the elements was assessed using a four-point scale for both of the dimensions: 1, denoting marginally exhaustive/specific and 3 highly exhaustive/specific. Naturally, the value 0 means not relevant. In addition in 2005, an element considered being "too small" was given an exhaustivity value denoted by a question mark '?'.

From 2005 on, the relevance assessments were executed so that the assessors marked up (i.e. painted) the relevant passages regardless of the element borders (Piwowarski and Lalmas 2004). From 2006 on, only specificity was measured. However, the specificity was calculated automatically based on the proportion of the relevant text. In other words, if half of the text of an element is relevant, then the relevance score, i.e. specificity for the element by definition is 0.5. Similar relevance assessments are also available in TREC Hard Track's passage retrieval (Allan 2004). Obviously, the concept of specificity has been changed during the years, and the proportion of relevant text is clearly not the same as it was defined explicitly.

Relevance density has been used as a substitute term for precision for example, by Lee and others (2008). Arvola and others (VII) defined the relevance density of a document as its proportion of relevant text. Zhao and Callan (2009) in their conference presentation also used density in the same fashion as the proportion of relevant texts (in bytes) in the document. As a sideswipe, we think the term relevance density suits better than specificity as the proportion of relevant text. The reason for using relevance density as specificity is that Ogilvie and Lalmas (2006) showed that the fraction of relevant text approximates well to the earlier definition of specificity.

4.4 Metrics

The development of the metrics within INEX is related to the overlap problem as well as the trade off between exhaustivity and specificity. Addressing these issues has led to the development of numerous metrics.

During the early years of the initiative, the elements were retrieved regardless of the overlap and measured with the *inex_eval* metrics (Gövert et al. 2006). Since then, the metrics have been developed to favour retrieval approaches limiting the overlap among elements. Eventually, that led to separate tasks, one allowing overlap (CO.thorough), and the other not (CO.focussed).

The evaluation of the fetch and browse approach is a prominent issue in content-oriented XML retrieval. This is mainly because the Relevant-in-Context (RiC) task is considered as the most credible from the users' perspective in INEX (Trotman et al. 2007). The task corresponds fully to the fetch and browse approach. The evaluation of the task is introduced in Section 4.4.3.

Next we focus on metrics used for measuring the performance of focused and thorough result organizing strategies. Among the metrics in INEX, two are used in the individual studies of the present dissertation to measure system performance. These include inex_eval_ng and extended cumulated gain (XCG). There are a number of other metrics for evaluating XML retrieval, such as EPRUM (Piwowarski and Dupret 2006) and a method developed by Ali and others (2008), which are not included in the summary of the present dissertation (see Study VI).

4.4.1 inex_eval and inex_eval_ng

When the initiative started in 2002, a metric for XML element retrieval called *inex_eval* was presented. It is based on a *precall* measure (Raghavan et al. 1989). This has been applied for XML elements using the probability that the element viewed by the user is relevant, as follows:

$$P(rel \mid retr, x) = \frac{x \cdot n}{x \cdot n + esl_{x \cdot n}} \tag{2}$$

In the equation x is an arbitrary recall point, n is the number of relevant elements for the request. Expected search length is denoted by $esl_{x \cdot n}$ Expected search length (ESL, Cooper 1968) takes the expected user effort into account as the average number of documents (here elements) the user has to browse in order to retrieve a given number of relevant documents. It is defined as follows:

$$esl_{x \times n} = j + \frac{s \cdot i}{r + 1} \tag{3}$$

In the equation, j denotes the total number of non-relevant elements on all levels preceding the final level, s is the number of relevant elements required from the final level to satisfy the recall point, i is the number of non-relevant elements in the final level, and r is the number of relevant elements in the final level. The term "level" is used here to denote the set of elements that have the same rank in the retrieval process.

The criticism against the inex_eval metrics was that it does not take the overlap of elements into account, and the recall was calculated against all assessed and overlapping elements. This phenomenon is called an overpopulated recall base (Kazai et al. 2004).

In 2003 a new metric *inex_eval_ng* was introduced to address the overpopulated recall base problem (Gövert et al. 2006). In inex_eval_ng, the recall and precision values for a ranked result list $\langle c_1, c_2, \ldots, c_n \rangle$ are defined separately:

$$recall = \frac{\sum_{i=1}^{k} e(c_i) \cdot \frac{|c'_i|}{|c_i|}}{\sum_{i=1}^{N} e(c_i)} \tag{4}$$

$$precision = \frac{\sum_{i=1}^{k} s(c_i) \cdot |c'_i|}{\sum_{i=1}^{k} |c'_i|} \quad (5)$$

In the equations N is the total number of elements in the collection; $e(c)$ and $s(c)$ denote the quantised assessment values of element c according to the exhaustivity and specificity dimensions. $|c|$ denotes the size of the element, and $|c'|$ is the size of the element that has not previously been seen by the user. $|c'|$ is computed as:

$$|c'_i| = \left| c_i - \bigcup_{c \in C[1,n-1]} (c) \right| \quad (6)$$

where n is the rank position of $|c_i|$ and $C[1, n-1]$ is the set of elements retrieved between the ranks $[1, n-1]$.

In the inex_eval_ng metrics the exhaustivity and specificity values vary between [0,1]. In 2003 the generalized quantization exhaustiveness (e) was defined as $e/3$ and similarly specificity (s) was $s/3$. In INEX 2004 several quantizations were introduced (Malik et al. 2004). For example a specificity-oriented (so) quantization defines an element to be relevant if and only if it is highly exhaustive, formally:

$$f_{s3\_e321} = \begin{cases} 1, if\ e \in \{3,2,1\}\ and\ s = 3 \\ 0, \quad otherwise \end{cases} \quad (7)$$

Similarly, meaningful combinations are those which have exhaustivity and specificity values within the sets {3, 2, 1}, {3, 2} and {3}, totaling 9 different combinations. If both of the dimensions are 3 (i.e. f_{s3_e3}), then the quantization is called strict.

4.4.2 eXtended Cumulated Gain

The extended cumulated gain (xCG) metric (Kazai and Lalmas 2006) is an extension of the cumulated gain (CG) metric introduced by Järvelin and Kekäläinen (2002). The metric considers the overlap and near misses of XML elements. The xCG value at rank i (i.e. without normalization) is calculated similarly to CG as follows:

$$xCG[i] = \sum_{j=1}^{i} xG[j] \quad (8)$$

For example the ranking xG = ⟨2, 1, 0, 0, 2, 1⟩ gives the cumulated gain vector xCG = ⟨2, 3, 3, 3, 5, 6, 6, 6,…⟩. Note that the list is considered to be infinite, so that e.g. xCG[100] = 6. The lengths of the search result lists vary with each request. Thus, the value at each rank should be normalized across queries. This is done by sorting the documents of a result list by relevance, producing an ideal xCG at position *i*. The ideal vector comes from the ideal ranking of elements. Formally, for a request, the normalized extended discounted cumulated gain (nxCG), is computed as follows:

$$nxCG[i] = \frac{xCG[i]}{xCI[i]} \quad (9)$$

Continuing the previous example, let us assume that the ideal ranking is *xI* = ⟨2, 2, 2, 1, 1, 1⟩ then *nxCG* = ⟨1, 0.75, 0.5, 0.43, 0.63, 0.67, 0.67,…⟩. The xCG metric is intended for the focused result organizing strategy where the overlapping elements are excluded. Therefore, in comparison to the flat result list selecting the relevant elements for the ideal vector can be seen somewhat contractual, because one can select the "optimal" relevant element from any level of an XML branch. In xCG it is defined so that the element having the highest relevance value on the relevant XML path is chosen. If elements in the same branch have the same score, the lowermost element is chosen.

In addition to the various cut-off values at a rank, we can calculate the average up to rank *i* as follows:

$$AnxCG[i] = \frac{\sum_{j=1}^{i} nxCG[j]}{i} \quad (10)$$

where AnxCG is short for average normalized discounted cumulated gain. Analogously to mean average precision, the MAnxCG refers to mean values over all requests.

The previous cutoff based measures can be considered user-oriented (Kazai and Lalmas 2006). A more "system-oriented" measures of the metric is called effort-precision/gain-recall (EP/GR). The method is analogous to standard precision/recall in a sense that precision is dependent on the recall. This makes it

possible to use interpolation techniques, for instance. Effort-precision/gain-recall is defined to calculate the relative effort (i.e. the number of visited ranks) the user is required to expend when scanning a system's result ranking compared with the effort an ideal ranking would take in order to reach a given level of gain relative to the total gain that can be obtained.

The effort-precision is defined with the following equation:

$$ep[r] := \frac{i_{ideal}}{i_{run}} \tag{11}$$

in which i_{ideal} is the rank at which extended cumulated gain value r is reached and i_{run} is similarly the rank at which the r is reached with the system run. Effort precision can be calculated at arbitrary gain-recall points, which are defined with the following equation:

$$gr[i] := \frac{xCG[i]}{xCI[N]} \tag{12}$$

where N is the total number of relevant components.

4.4.3 Measuring the Effectiveness of the Relevant-in-Context Task

From the evaluation perspective, the fundamental difference between content-oriented XML retrieval and passage retrieval is that in XML retrieval the passages are atomic elements, i.e. text between the element's start and end tags, whereas in passage retrieval the passages are not dependent on element boundaries. Currently, both approaches are supported in INEX. That is because the relevance assessments are executed so that the assessors have been marking up the relevant passages regardless of any element boundaries (Piwowarski and Lalmas 2004). Thus, a recall base for a document consists of a character-position set.

The contribution of this dissertation in developing evaluation metrics focuses on measuring the fetch and browse result organizing strategy, and more specifically the Relevant-in-Context (RiC) task. The task was introduced in 2006 and may be seen as a special case of document retrieval, where the access to a document is enriched with focused (i.e. element or passage) retrieval. In the evaluations, the system performance is calculated similarly to full document retrieval, except that the

relevance score for each individual relevant document depends on the elements or passages the system provides. According to the RiC task, separate scores are calculated for each individual retrieved document as a document score, and for the document result lists as a list score. We refer to the former as the document level evaluation and to the latter as the list level evaluation. The list score is calculated in INEX with generalized precision/recall metric (Kekäläinen and Järvelin 2002).

The official INEX metric for the document level evaluation is an f-score, which is calculated for each retrieved document *d* as follows:

$$F_{\alpha}(d) = \frac{(1+\alpha^2) \cdot P(d) \cdot R(d)}{\alpha^2 \cdot P(d) + R(d)} \qquad (13)$$

in which $P(d)$ (document precision) is the number of retrieved relevant characters divided by the number of retrieved characters and $R(d)$ (document recall) is the number of retrieved and relevant characters divided by the total number of relevant characters in the document. The α value is used to adjust the role of the precision in the formula. It is worth mentioning that in order to favor focused retrieval strategies over those retrieving everything from the document, the α value has been 0.25 to emphasize precision in the official INEX evaluations since 2008 (Kamps et al. 2008). Finally, the list score is calculated over ranked lists of documents based on these f-scores (Kamps et al. 2007).

Document level evaluation can be associated with traditional document retrieval evaluation in the sense that the unit to be retrieved is a character and is treated as if it were a document. The f-score is based on the set of characters. Hence, the f-score, can be paralleled to correspond with the full match evaluation measures in document retrieval, and the implicit user model behind it, is that the user reads all information provided by the system and nothing else.

Accordingly, the flow diagram in Figure 6 by Arvola and Kekäläinen (2010) represents the user's browsing model within a document. The model is a consideration of the logical alternatives to the browsing of any document. First, the user accesses a document and starts to browse. The browsing may simply lead either to reading the document from the beginning on or from some other point. For example, the user may utilize an interface providing guiding gadgets to reach the expected best entry point, for example. Nevertheless, any browsing strategy will eventually lead to reading some text passage (or picture, if sought) and determining

its relevance. After assessing the passage's relevance, the user decides either to read another passage or move on to the next document (or to perform a new search). If the passage seems to completely fulfil the user's information needs, he or she leaves the document. On the other hand, the user may become frustrated and discover that there is no (further) relevant material to be read. If the user is still willing to continue, he or she again faces the alternatives as to how to proceed with browsing the document.

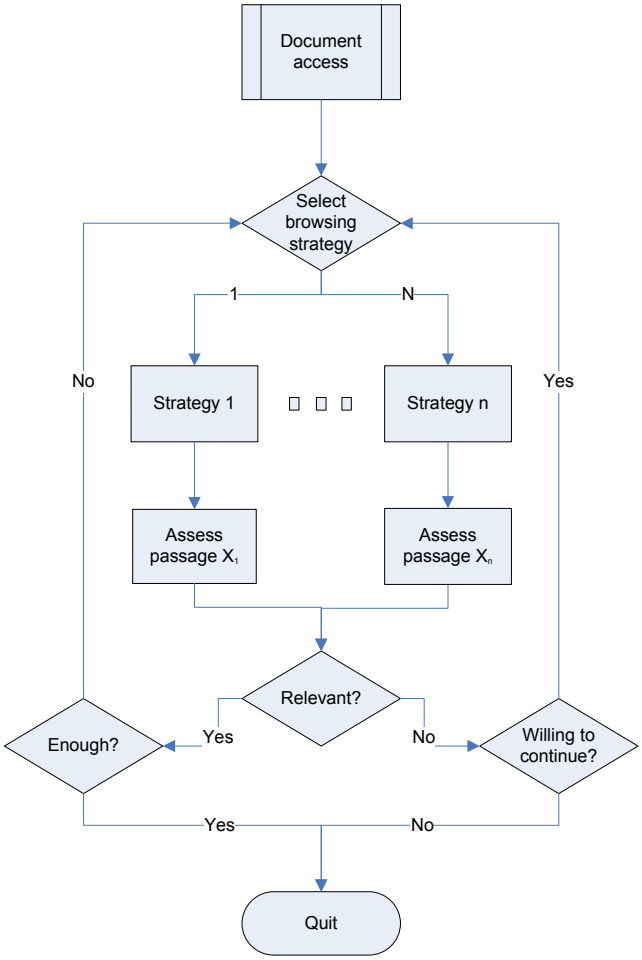

Figure 6. A flow diagram of a user's browsing within a document

This implies that there ought to be an order in which the passages are assessed and a point at which the user quits reading the document. For example, the concept of tolerance to irrelevance (T2I) (de Vries et al. 2004) is related to the willingness to continue with the current document if non-relevant material is read. T2I refers to the point at which the amount of non-relevant material is exceeded. The reading effectiveness of a user is related to the issue of which passages are read first.

5. Summary of the Studies

The present dissertation has two main themes related to XML retrieval:

I. Element retrieval, indexing and contextualization (I-V).
II. Retrieval evaluation by taking the context and the reading order of retrieved elements into account (VI and VII).

The first theme addresses the focused and thorough result organizing strategies and retrieval enriched with structural queries (i.e. CAS). The second theme addresses the fetch and browse result organizing strategy. In both themes, context plays a crucial role, since in the first theme context is used as extra evidence for element scoring, and in the second theme the document is considered as the context of the retrieved elements. In other words, in both of these themes the context of the retrieved material is taken into account.

The studies utilize the laboratory environment provided by INEX. In Studies I - V the IEEE test collection is used and Studies VI - VII use the Wikipedia (2008) test collection (Denoyer and Gallinari 2006). The metrics used in Studies I, II and V include inex_eval_ng and nxCG.

5.1 Theme I: TRIX Retrieval System and Contextualization

Studies I - V include XML IR system development and retrieval methods, especially contextualization. The TRIX system is constructed in Study I and used throughout the Studies I - V. The effectiveness of the methods is tested within the laboratory environment provided by the INEX initiative. Study I and II use the INEX 2004 laboratory environment and related CO topics. Contextualization, later labeled vertical contextualization, is introduced in Study II. A wider variety of different contextualization models was given in Study IV. For Studies III and IV a tailored

test setting is used to measure only the scoring quality of contextualization at three different granularity levels. In Study III the INEX 2004 laboratory environment and in Study IV INEX 2005 laboratory environment is used as well. The main contribution of Study V in light of contextualization, was applying vertical contextualization to CAS. Next, we sum up what was done in the individual studies related to the first theme.

5.1.1 Study I: TRIX 2004 – Struggling with the Overlap

The primary research goal of Study I is to construct an XML IR system based on structural indices, i.e. the Dewey labeling schema. Accordingly, the study introduces TRIX (Tampere Retrieval and Indexing for XML), a DTD and schema independent XML IR system which methods are thus applicable to heterogeneous XML collections. The matching method of TRIX is a BM25 modification for XML retrieval, where the modified BM25 formula is parameterized to favour different sizes of XML elements.

Instead of normalization based on the length of documents or elements, TRIX has a normalization function based on the number of *content elements* (i.e. elements which are the highest elements in XML hierarchy having their own textual content). In TRIX, these elements constitute the lowest level to index.

Aside with the constructive part, this study addresses measuring the performance in relation to the result list presentation, which still constitutes an open question in XML information retrieval. For instance, if no pre-selection of the result elements is conducted, the result list contains overlapping elements and thus redundant data. Accordingly, the study criticizes measuring the performance without taking the overlap into account.

In order to study what is the impact of overlap on the evaluation results using measures of that time, the TRIX system is tested using the INEX 2004 laboratory environment including the CO topics and the inex_eval_ng metric. The results measured with the inex_eval_ng metric are notably better when the overlap is increased from the results from full overlap via partial overlap to no overlap. In later experiments using the nxCG metric instead of inex_eval_ng, the ranking of TRIX runs with full overlapping reported in the study rose from rank 40 to rank 1

(UTampere_CO_average), and from rank 42 to rank 2 (UTampere_CO_fuzzy) among all INEX 2004 runs (Kazai et al. 2005).

5.1.2 Study II: Generalized Contextualization Method for XML Information Retrieval

The structural indices used in TRIX give a straightforward access to the ancestors of an element. Therefore, the indices allow mixing the evidence (scores) among elements in the vertical context in an elegant fashion. Study II generalizes the use of context in a hierarchical structure in element scoring and conceptualizes the idea of utilizing an element's context by labeling it contextualization. The study distinguishes finer grained scoring models than merely using the root by separating distinct context levels, parent, ancestor and root. The roles are assigned to those levels by a contextualization vector. In addition the study shows how vertical contextualization is operationalized using structural indices as the theoretical framework. Three vertical contextualization methods are distinguished: *parent* using the parent as context, *root* using the root as context and *tower* utilizing all ancestors as context.

Study II investigates, whether contextualization has an effect on the effectiveness on the results, in comparison to the situation where no context is utilized. In other words, should a text passage in a good context be ranked higher than a similar passage in a poor context (and vice versa)? The laboratory environment is the same as in Study I and the results obtained with inex_eval_ng metric indicate that using any of the contextualization methods presented significantly improve the effectiveness in comparison to the baseline (TRIX, no contextualization). However, the *root* contextualization with root multiplied by two delivers the best results. In addition a comparison with the gold standard, i.e. the best performing INEX run (IBM Haifa Research Lab: CO-0.5-LAREFIENMENT) (Malik et al. 2004), is successful. Our term "contextualization" has been adopted as part of the Encyclopedia of Database Systems (see Kekäläinen et al. 2009).

5.1.3 Study III: The Effect of Contextualization at Different Granularity Levels in Content-Oriented XML Retrieval

Study III examines how the models presented in Study II perform in distinct granularity levels. In other words, we test how the model works on small, intermediate and large elements within the given contextualization framework. The following research questions are addressed:

- What is the effectiveness of vertical contextualization in the retrieval of elements of different granularities?
- How to address the previous question by using traditional evaluation measures?

In the study the INEX 2004 collection and related relevance assessments are used. The queries are executed against each of the predefined three granularity levels (content element, minor section, major section) with TRIX. The content element level is defined in Study I. The empirical part of the study is carried out in a conventional laboratory setting, where the INEX recall base is granulated, i.e. the set of elements is pre-selected (see the diamond in Figure 5). In other words, the relevance assessments are simplified so that an element is relevant if it contains relevant text. This kind of relevance interpretation is widely used e.g. in TREC making it possible to measure performance separately at different levels, and above all to use traditional metrics over those intended for heterogeneous element lists. Note that contextualization goes beyond the pre-selected set of retrievable elements, i.e. extra evidence is gathered above. The evaluation method is described in detail in Study IV.

The results confirm the effectiveness of contextualization, and show how the effects of different contextualization methods vary along with the granularity level. The results show that vertical contextualization improved the effectiveness mostly on deep and small elements. The results suggest that with the given contextualization model, utilizing the near context delivers better results than the root for the small elements.

5.1.4 Study IV: Contextualization Models for XML Retrieval

Study IV extends the work reported in Study III and augments its first research question with a broader definition and a more detailed parameterization of contextualization. The study utilizes the levels described in Study III and provides a more detailed and systematic definition of the research setting. Special attention is paid to the lowest elements, i.e. content elements (I). A classification of three contextualization models is introduced: vertical (i.e. hierarchical), horizontal (i.e. ordinal) and ad hoc contextualizations. The novel method, horizontal contextualization, followed the document order in the document structure, is also formalized and tested. In other words, the following research questions are addressed:

- What is the effectiveness of vertical and horizontal contextualization models in the retrieval of elements of different granularities?
- How to manipulate the presented contextualization models?

For the vertical contextualization, a finer grained parameterization than in the earlier studies (II, III) is introduced learned and tested in two distinct settings. That is, we use INEX 2004 data for training and INEX 2005 for testing different contextualization parameters and then 2005 data for training and 2004 for testing. The horizontal contextualization is tested using only the content element level.

Generally, the results are somewhat in line with the previous studies (II, III) in relation to contextualization. The improvements measured with MAP and nDCG are notable and the results of most topics are improved by contextualization. The results between the 2004 and 2005 collections show some inconsistency, because the small elements benefit in the 2004 collection but in the 2005 collection the larger elements benefit. We conducted a lot of experiments in order to study this phenomenon. The XML document collections not fundamentally different in 2004 and 2005, and cannot be seen as the distinguishing factor of the somewhat contradictory results. In addition to the reported results, we tried to find topical qualities which could possibly prognose the success of contextualization. We analyzed, for instance, the recall bases and estimated the quality of context as the relevance densities of articles in relation to the chances in the result lists. Unfortunately, no hard evidence or dependencies are found.

5.1.5 Study V: Query Evaluation with Structural Indices

Study V investigates the contextualization in CAS queries. In other words, how do CAS queries benefit from contextualization? Actually, studying the benefit of contextualization with SVCAS and SSCAS, i.e. queries having strict target element constraints, is quite similar to the studies with granularity levels (III, IV). This is because the result list is flat (except when retrieving overlapping sections) and the retrievable elements are of same kind. The major difference is that the recall base has no full coverage of the collection.

The study presents our experiments and results with INEX 2005 using the INEX 2005 laboratory environment. The results delivered by the TRIX system are evaluated within six retrieval tasks of INEX 2005: CO.thorough, CO.focused, VVCAS, VSCAS, SVCAS and SSCAS. The CO.thorough and CO.focused refer to the corresponding result organizing strategies. The results for the CO task show that *root* contextualization is not generally better than *root* + *tower*, except for the early precision. The analyzing power of structural indices enables a straightforward processing of CAS queries, and the successful combined effectiveness of the TRIX matching method and contextualization especially with the strict interpretation of target element constraint (SVCAS, SSCAS) is illustrated.

5.2 Theme II: XML IR Evaluation Based on Expected Effort and Reading Order

XML retrieval provides focused access to the relevant content of documents. For the second theme we generalize element retrieval to passage retrieval, where the elements are considered as passages between start and end tags. In addition, passages, even arbitrary ones, are considered as the retrievable units. Therefore, the second theme aims to develop methods for measuring focused retrieval.

A good evaluation methodology is a prerequisite for any systematic IR system development. The TRIX results of Study I measured with inex_eval_ng and nxCG indicated that the qualities of the evaluation metrics have a crucial impact on the evaluation results in XML retrieval. Therefore, the evaluation metrics should be designed carefully. In Study VI, we develop a novel evaluation framework and test

it over all runs of INEX 2008 RiC task. In Study VII we test the same framework for sparsely and densely relevant documents. The framework considers the fetch and browse result organizing strategy, where the retrieved passages within a document form the *primary categor*y and are assumed to be read first, whereas their context (other passages) form the *secondary category* and is assumed to be read thereafter. The reading of a document is expected to end when tolerance-to-irrelevance (de Vries et al. 2004) is met, or the document is read right through. The passages in the secondary category are missed in passage retrieval (browse), but they belong to the retrieved document. The INEX 2008 laboratory environment was used in both of the studies. Next, we sum up what was done in the individual studies related to the second theme.

5.2.1 Study VI: Expected User Reading Effort in Focused IR Evaluation

The study introduces a novel framework for evaluating passage and XML retrieval in the context of fetch and browse result organizing strategy. This study contributes to the RiC task of INEX. Special attention is paid to document level evaluation, in contrast to list level evaluation.

The document level metric (f-score) of the RiC task (until INEX 2010) tends to favor systems returning everything within a document. This is partially because of more focused systems, retrieving only parts of documents, loses easily in recall. This is especially likely when the document is thoroughly relevant. The study proposes that the implicit assumption behind the f-score of the user reading all the retrieved text and nothing but the retrieved text is not sufficient. Instead, a reading order of the document and a breaking condition of reading are assumed. The study hypothesizes that taking these assumptions into account affects the set of read text and thus the mutual rankings of systems in comparison to f-score and using the simplistic assumptions behind it. More importantly, the study claims that focused retrieval benefits from these assumptions, especially when the breaking condition is assumed to occur early.

The framework focuses on a user's effort in locating the relevant content in a result document. We introduce a metric called cumulated effort (CE) and character precision/recall (ChPR) and justify our approach with a small screen scenario

adopted from (Arvola et al. 2006). A small screen is assumed to make the user readings breaking point occur earlier and to give a better justification for the use of navigational gadgets over skim reading.

For the metrics, we consider T2I as the breaking condition and the reading order to be the modeled user parameters. The T2I is bound to the screen size and two generic browsing strategies are introduced. The baseline (default) browsing strategy is reading the document consecutively from the beginning to the breaking condition, which is either T2I or the end of the document. This strategy is compared with the focused strategy, where the retrieved passages are read first in document order.

Measuring the effort is based on the system guided reading order of documents. The effort is calculated as the quantity of text the user is expected to browse through. More specifically, this study considers evaluation metric following a specific fetch and browse approach, where in the fetch phase documents are ranked in decreasing order according to their document score, as in document retrieval. In the browse phase, for each retrieved document, a set of non-overlapping passages representing the relevant text within the document is retrieved.

The proposed metrics calculate the system effectiveness in two phases. The list score is based on the document score. The document level relevance is "graded". With the ChPR metric, it is within the scale [0,1]. This allows the utilization of the generalized precision recall metric (Kekäläinen and Järvelin 2002), which is used also in INEX for the list level score having f-score at the document level. In the CE metric, the document score is calculated using document effort score (ES).

The document level metrics are compared with the official INEX metric (f-score), by observing the correlations between the mutual rankings of all 38 INEX 2008 runs. MAgP is used as the ranking criterion for f-score and ChPR, and mean average cumulated effort (MANCE) for the CE. In addition, we pay special attention to some of the top performing runs. From these runs, we construct additional runs by transforming them so that the browse phase is discarded. This means that instead of considering the retrieved passages within a document only, everything is returned. This aims to resemble to the full document retrieval.

The results show that unlike measuring with the f-score, using the reading order assumptions and the proposed metric, the focused retrieval performs better in comparison to the document retrieval. This trend is even stronger when a smaller

screen (i.e. earlier breaking point) is assumed, i.e. the T2I level is 300 instead of 2000 characters.

5.2.2 Study VII: Focused Access to Sparsely and Densely Relevant Documents

The study complements Study VI by taking a more focused view of single document level evaluation, instead of the whole list of documents. The study addresses the following research question within the fetch and browse result organizing strategy:

- How does a document's relevance density affect the performance of focused retrieval in comparison to full document retrieval when tolerance to irrelevance and expected reading order are assumed?

In the study document level evaluation is considered, totally ignoring document ranking in the result list. Relevance density as the proportion of relevant text in retrieval is introduced.

All the relevant documents of INEX 2008 recall base are sorted according to their relevance density. Then the sorted list of documents is split into deciles, each covering 10% of the documents. The best RiC run of INEX 2008 is analyzed as in Study VI. In other words, it is compared with the same run, but as if it were full document retrieval. The study assumes a reading model and T2I described in Study VI and shows that in sparsely relevant documents focused retrieval performs better than full document retrieval. For densely relevant documents the performance is equal. Surprisingly, when measured with document recall instead of precision, the gap between the full document and focused runs is greater. In addition, the expected amount of read text is substantially smaller in focused retrieval than in document retrieval. In addition the study illustrates the average distribution of the relevant text of the documents over each decile.

6. Discussion and Conclusions

The main purpose of the present dissertation is to contribute to the field of focused retrieval in the development of retrieval methods and evaluation metrics. Accordingly, it has two related main themes. Theme one covers element retrieval, indexing and contextualization and theme two covers the retrieval evaluation by taking the context and the reading order of retrieved elements into account. Both themes address XML retrieval and theme two more generally focused retrieval as well. The common denominator of the themes is taking the retrieved element's context into account, both in element scoring and in evaluation. This approach as a whole is novel and unique, since the traditional evaluation metrics, as well as scoring methods of XML IR to some extent consider XML elements and fragments, to be context independent units of retrieval.

In order to support findings for theme one, an XML retrieval system, TRIX, is introduced in Study I. It is based on structural indices which enable schema free manipulation of XML, and make TRIX suitable for heterogeneous XML collections. In addition, the indices enable smooth access to the XML hierarchies and element contexts. The BM25 inspired matching method of TRIX gives an adequate baseline for further experiments in theme one.

Using the TRIX system and the INEX test bed, in Studies II – V, a method called contextualization, is recognized, labeled, developed and tested. Contextualization improves the scoring quality of individual elements and as Studies III, IV and V show, the feature is stressed for small items on the test collection. The contextualization method is tested, and found successful with focused (II), thorough and known item (CAS and granulated collection) (III, IV and V) search strategies.

The experiments are conducted using the IEEE collection only. While contextualization is discovered to be beneficial within the INEX test bed and IEEE collection used, it is worth noting that the benefit is dependent on the text genre. Obviously, if a collection consists of semantically individual items (such as an

encyclopedia), using contextualization may not be helpful. The contextualization itself is a general method, but the effect may vary from collection to collection. Also, the performance improvements in the fetch and browse result organizing strategy remains to be researched. In that case contextualization with the elements other than root, and with horizontal contextualization, may be hypothesized to be beneficial.

Even if we tested the contextualization with a single system, contextualization as a re-scoring method is applicable to various other matching strategies, including language models, vector space models etc. The only requirement is that every element is given an initial score, and the final, contextualized score is based on the linear combination of the initial scores. Due to its general and effective applications the term "*Contextualization*" is adopted as an Encyclopedia of Database Systems entry (Kekäläinen et al. 2009).

In measuring the benefit of contextualization on different granularity levels, the method of granulating a collection and measuring the results using standard evaluation metrics derived from full document retrieval, is among the advances of the evaluation of XML retrieval, and thus somewhat overlaps with theme two. The contemporary INEX metrics for XML element retrieval for focussed and thorough result organizing strategies (see Section 4.4) measure the combination of scoring quality and the selection of the 'right' granularity level. By contrast, the method introduced in Study III and described in detail in Study IV, removes the task of selecting the granularity level and makes it possible to measure the scoring quality only. Thus, the evaluation setting can be simplified by omitting the specificity dimension.

An XML collection can be granulated in numerous ways. Using the IEEE collection, in Studies III and IV only the content element level was generic, other levels were rather ad hoc. Obviously, granulation could be defined differently and for any other collection as well, such as the Wikipedia (Schenkel et al. 2007). However, it is worth noting that a decent granulation of the previous Wikipedia (Denoyer and Gallinari 2006) would deliver highly complex granularity level definitions.

The first theme addresses element retrieval without context, whereas the second theme focuses solely on in context retrieval. The INEX initiative has studied focused retrieval since 2002. Within its ad hoc track, various tasks have been

introduced. Of these, the fetch and browse task came along in 2006 (later Relevant-in-Context or Best-in-Context) and gained popularity through its credibility from the user's point of view. Unfortunately, from the perspective of focused retrieval, with the past evaluation measures of INEX, full document retrieval has appeared competitive in comparison to focused retrieval with this task (RiC). However, when taking the reading order and tolerance to irrelevance into account, the evaluation framework of the present dissertation seems to be beneficial in comparison to full document retrieval. The effect is particularly strong when focused retrieval is applied to sparsely relevant documents as the Study VII showed. The method reported in Study VI is a step towards a user simulation (Arvola and Kekäläinen 2010) as a "*what if*" type of simulation. In other words, we measure what happens if the reported conditions are assumed as user behavior.

The second theme addressing the in context retrieval evaluation together with Studies III and IV addressing element retrieval without context, comprise a toolset for focussed retrieval evaluation. This toolset can be claimed to cover a reasonable portion of use cases within XML retrieval, and thus the toolset serves as an alternative to the antecedent INEX ad hoc track evaluation. Accordingly, the most visible outcome of the toolset is the adoption of the T2I@300 measure of the ChPR metric (VI) as an official measure in the INEX performance evaluations in the ad hoc track 2010[12].

The evaluation methods in the second theme are aiming to mimic user behaviour in a more credible way than the other existing methods. At this stage this better credibility is based only on intuitive judgments. In the present dissertation, we expect how the user behaves based on the user interface and search results. For example a small display restricts the number of browsing alternatives; with such a device it is not meaningful or even possible to use a flip through browsing strategy for a long document. Naturally, the real user behaviour may be something else. User studies may give more light for developing the methods further.

[12] http://www.inex.otago.ac.nz/tracks/adhoc/runsubmission.asp?action=specification 25.10.2010

References

Abiteboul, S. (1997). Querying semi-structured data. In *Proceedings of the 6th International Conference on Database Theory, ICDT 1997*, 1-18.

Agrawal, R. Borgida, A., and Jagadish, H.V. (1989). Efficient management of transitive relationships in large data and knowledge base. In *Proceedings of the International Conference on Management of Data, SIGMOD 1989*, 253 – 262.

Ali, M. S., Consens, M. P., Kazai, G., and Lalmas, M. (2008). Structural relevance: a common basis for the evaluation of structured document retrieval. In *Proceedings of ACM 17th Conference on Information and Knowledge Management, CIKM 2008,* 1153-1162.

Allan, J. (2004) Hard track overview in TREC 2004: High accuracy retrieval from documents. In *Proceedings of the 13th Text Retrieval Conference (TREC 2004)*. Nist Special Publication, SP 500-261, 11 pages.

Amer-Yahia, S., Botev, C., Buxton, S., Case, P., Doerre, J., Holstege, M., Melton, J., Rys, M., and Shanmugasundaram, J. (eds.) (2008): XQuery 1.0 and XPath 2.0 Full-Text. W3C Candidate Recomm. 16 May 2008. http://www.w3.org/TR/xquery-full-text/ (2008). Accessed 30 June 2009

Arvola, P. (2007). Document Order Based Scoring for XML Retrieval, In *INEX 2007 Workshop Pre-Proceedings*, 6 pages.

Arvola, P. (2008). Passage Retrieval Evaluation Based on Intended Reading Order, In *Workshop Information Retrieval, LWA 2008*, 91-94.

Arvola, P., Junkkari, M., Aalto, T., and Kekäläinen, J. (2005). Utilizing Context in Weighting of XML Elements for Information Retrieval, *Report A-2005-1*, Department of Computer Sciences, University of Tampere, 20 pages.

Arvola, P., Junkkari, M., and Kekäläinen, J. (2006). Applying XML Retrieval Methods for Result Document Navigation in Small Screen Devices. In *Proceedings of Mobile and Ubiquitous Information Access (MUIA) at MobileHCI 2006*, 6-10.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.

Boag, S., Chamberlin, D., Fernandez, M. F., Florescu, D., Robie, J., and Siméon, J. (2008). XQuery 1.0: An XML query language. W3C Recommendation. Retrieved September 15, 2010 from http://www.w3.org/TR/xquery/.

Böhme, T., and Rahm, E. (2004). Supporting Efficient Streaming and Insertion of XML Data in RDBMS, In *Data Integration over the Web* (*DIWeb 2004*), 70-81.

Bray, T., Paoli, J and Sperberg-McQueen, C.M. (1998). Extensible Markup Language (XML) 1.0. http://www.w3.org/TR/1998/REC-xml-19980210. W3C Recommendation. Technical report, W3C (World Wide Web Consortium).

Buneman, P. (1997). Semistructured Data. In *Proceedings of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, PODS 1997*, 117-121

Carmel, D., Maarek, Y., Mandelbrod, M., Mass,Y., and Soffer, A. (2003). Searching XML documents via XML fragments. In *Proceeding of the 26th International ACM Conference on Research and Development in Information Retrieval, SIGIR 2010,* 151-158

Chiaramella, Y. (2001). Information retrieval and structured documents. In *Lectures on information retrieval*, 286–309.

Christophides, V., Plexousakis, D., Scholl, M., and Tourtounis, S. (2003). On labeling schemes for the semantic web. In *Proceedings of the 12th International Conference on World Wide Web*, *WWW 2003*, 10 pages.

Clark, J., and Derose, S. (eds.) (1999). XML path language (XPath) version 1.0. W3C Recommendation. 16 November 1999. http://www.w3.org/TR/xpath20/ Accessed 23 October 2010.

Cohen, S., Mamou, J., Kanza, Y., and Sagiv, Y. (2003). XSEarch: A semantic search engine for XML, In *Proceedings of the 28th Conference on Very Large Databases, VLDB 2003*, 719-730.

Cooper, W. (1968). Expected search length; a single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation*, 19(1), 30–41.

Denoyer, L. and Gallinari, P. (2006). The Wikipedia XML Corpus. *SIGIR Forum*, 40(1). 64-69.

Deschler, K., and Rundensteiner, E. (2003). MASS: a multi-axis storage structure for large XML documents. In *Proceedings of ACM 13th Conference on Information and Knowledge Management, CIKM 2003,* 520-523.

Duong, M. and Zhang, Y. (2005). LSDX: a new labelling scheme for dynamically updating XML data, In *Proceedings of 16th Australasian Database Conference, ADC 2005*, 185-193.

Elmasri, R., and Navathe, S. (2004). *Fundamentals of Database Systems*. Addison Wesley, 5th edition.

Fuhr, N., Gövert, N., Kazai, G., and Lalmas, M. (eds). (2002). *Proceedings of the 1st Workshop of the INitiative for the Evaluation of XML Retrieval*.

Fuhr, N., Kamps, J., Lalmas, M., Malik, S., and Trotman, A. (2008). Overview of the INEX 2007 Ad Hoc Track, In *Proceedings of 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007, Lecture Notes in Computer Science 4862*, 1-23.

Geva, S. (2006). GPX - Gardens Point XML IR at INEX 2005. In *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977*, 240-253.

Goldfarb, C. F., and Prescod, P. (1988). The Xml Handbook, Prentice Hall.

Gövert, N., and Kazai, G. (2003). Overview of the initiative for the evaluation of XML retrieval (INEX) 2002, In *Proceedings of the 1st International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2002*, 1-17.

Gövert, N., Fuhr, N, Lalmas, M. and Kazai, G. (2006). Evaluating the effectiveness of content-oriented XML retrieval methods. *Information Retrieval*, 9(6), 699–722.

Grossman D., and Frieder, O. (2004). *Information Retrieval: Algorithms and Heuristics*, Springer.

Guo, L., Shao, F., Botev, C., and Shanmugasundaram, J. (2003). XRANK: ranked keyword search over XML documents, In *Proceedings of the International Conference on Management of Data, SIGMOD 2003*, 16-27.

Härder, T., Haustein, M., Mathis, C., and Wagner, M. (2007). Node labeling schemes for dynamic XML documents reconsidered, *Data and Knowledge Engineering*, 60(1), 126-149.

Harold, E, and Means, W. (2004). *XML in a Nutshell*, third edition, O'Reilly Media.

Holzner, S. (2004). *XPath. Navigating XML with XPath 1.0 and 2.0: kick start*, Sams Publishing, 366 pages

Ingwersen, P., and Järvelin, K. (2005). *The Turn: Integration of Information Seeking and Retrieval in Context*, The Information Retrieval Series, Springer-Verlag.

Itakura, K., and Clarke, C.L.A. (2010). A Framework for BM25F-based XML Retrieval. In *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010*, 843 – 844.

Jämsen, J., Näppilä, T., and Arvola, P. (2008). Entity Ranking Based on Category Expansion. In *Proceedings of 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007, Lecture Notes in Computer Science 4862,* 264-278.

Järvelin, K., and Kekäläinen, J. (2002). Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions on Information Systems*. 20 (4). 422-446.

Junkkari, M. (2005). PSE: An object-oriented representation for modeling and managing part-of relationships, *Journal of Intelligent Information Systems*. 25(2), 131-157.

Kamps, J. Marx, M., de Rijke, M., and Sigurbjörnsson, B. (2005). Structured queries in XML retrieval. In *Proceedings of ACM 14th Conference on Information and Knowledge Management, CIKM 2005,* 4-11.

Kamps, J., Lalmas, M., and Pehcevski, J. (2007). Evaluating relevant in context: document retrieval with a twist. In *Proceedings of the 30th annual International ACM SIGIR conference on Research and development in information retrieval*, *SIGIR 2007*, 749-750.

Kamps, J., Geva, S., Trotman, A., Woodley, A., and Koolen, M. (2008). Overview of the INEX 2008 ad hoc track. In *INEX 2008 workshop pre-proceedings*. 1–28.

Kamps, J. (2009). Presenting Structured Text Retrieval Results. *Encyclopedia of Database Systems 2009*, 2130-2134.

Kazai, G. (2004). Report of the INEX 2003 metrics working group. In *Proceedings of the 2nd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2003*.

Kazai, G., Lalmas, M., and De Vries, A.P. (2004). The overlap problem in content-oriented XML retrieval evaluation, In *Proceeding of the 27th International*

ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2010, 72-79.

Kazai, G., and Lalmas, M. (2006). eXtended cumulated gain measures for the evaluation of content-oriented XML retrieval. *ACM Transactions on Information Systems.* 24(4), 503-542.

Kazai, G., Lalmas, M., and de Vries, A.P. (2005). Reliability Tests for the XCG and inex-2002 Metrics In *Proceedings of 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Lecture Notes in Computer Science 3493,* 60-72.

Kekäläinen, J. and Järvelin, K. (2002). Using graded relevance assessments in IR evaluation. *Journal of the American Society for Information Science and Technology*. 53(13), 1120-1129.

Kekäläinen, J., Arvola, P., and Junkkari M. (2009). Contextualization. *Encyclopedia of Database Systems 2009*. 474-478.

Knuth, D. E. (1997). *The Art of Computer Programming* (Third ed.). Reading, Massachusetts: Addison-Wesley.

Manning, C.D., Raghavan, P. and Schütze, H. (2008*). Introduction to Information Retrieval*. Cambridge University Press.

Lee, K., Croft, W. B., and Allan, J. (2008). A cluster-based resampling method for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in information Retrieval*, *SIGIR 2008*, 235-242.

Lehtonen, M., Pharo, N., and Trotman, A. (2007). A taxonomy for XML retrieval use cases, In *Proceedings of 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006, Lecture Notes in Computer Science 4518*, 413-422.

Li, Q., and Moon B. (2001). Indexing and Querying XML data for regular path expressions, In *Proceedings of the 26th Conference on Very Large Databases, VLDB 2001*, 361-370.

Lu, W, Robertson, S and MacFarlane, A. (2006). Field-Weighted XML Retrieval Based on BM25. In *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977,* 161-171.

Luk, R.W., Leong, H.V., Dillon, T.S., Chan, A.T., Croft, W.B., and Allan, J. (2002). A survey in indexing and searching XML documents. *Journal of the American Society for Information Science and Technology*. 53(6), 415-437.

Malik, S., Lalmas, M., and Fuhr, N. (2005). Overview of INEX 2004, In *Proceedings of 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Lecture Notes in Computer Science 3493*, 1-15.

Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*, Cambridge University Press.

Meyer, B.J.F. (1985). Prose analysis: Purpose, procedures, and problems: Parts I and II. In *Understanding expository text*. Hillsdale, NJ: Lawrence Erlbaum, 269—304.

Niemi, T. (1983). A seven-tuple representation for hierarchical data structures. *Information Systems*. 8(3). 151-157.

Niemi, T., and Järvelin, K. (1996). The processing strategy for the NF^2 relational frc-interface. *Information & Software Technology*. 38(1), 11-24.

Näppilä, T., Järvelin, K., and Niemi, T. (2008). A tool for data cube construction from structurally heterogeneous XML documents. *Journal of the American Society for Information Science and Technology.* 59(3), 435-449.

Ogilvie, P., and Lalmas, M. (2006). Investigating the Exhaustivity Dimension in Content-Oriented XML Element Retrieval Evaluation. In *Proceedings of the 15th ACM International Conference on information and Knowledge Management, CIKM 2006*, 84-93.

Ogilvie, P., and Callan, J. (2006). Parameter Estimation for a Simple Hierarchical Generative Model for XML Retrieval. In *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977,* 211-224.

O'Keefe, R. A., and Trotman, A. (2004). The simplest query language that could possibly work. In *Proceedings of the 2nd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2003*.

O'Neil, P., O'Neil, E., Pal, S., Cseri, I., Schaller, G., and Westbury, N. (2004). ORDPATHs: insertfriendly XML node labels, In *Proceedings of the International Conference on Management of Data, SIGMOD 2004*, 903-908.

Pal, S. (2006). XML Retrieval: A Survey. Technical Report, *CVPR*.

Piwowarski, B., and Dupret, G. (2006). Evaluation in (XML) information retrieval: expected precision-recall with user modelling (EPRUM). In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in information Retrieval, SIGIR 2006*. ACM, New York, NY, 260-267.

Piwowarski, B., and Lalmas, M. (2004). Providing consistent and exhaustive relevance assessments for XML retrieval evaluation. In *Proceedings of the 13th ACM International Conference on information and Knowledge Management, CIKM 2004,* 361–370.

Raghavan, V., Bollmann, P., and Jung, G. (1989). A critical investigation of recall and precision. *ACM Transactions on Information Systems*. 7(3). 205–229.

Robertson, S.E., Zaragoza, H., Taylor, M.J. (2004). Simple BM25 extension to multiple weighted fields. In *Proceedings of the 13th ACM International Conference on information and Knowledge Management, CIKM 2004,* 42-49.

Salton, G., Wong, A., and Yang, C.S. (1975). A Vector Space Model for Automatic Indexing. Commun. ACM 18(11), 613-620.

Saracevic, T. (1996). Relevance Reconsidered. In *Proceedings of the 2nd International Conference on Conceptions of Library and Information Science: Integration in perspective, CoLis 1996*, 201–218.

Schenkel, R., Suchanek, F., and Kasneci, G. (2007). YAWN: A semantically annotated Wikipedia XML corpus, In *Proceedings of Datenbanksysteme in Business, Technologie und Web, BTW 2007*. 277-291.

Schultz, A. (1970). *Reflections on the problem of relevance*. New Haven, CT: Yale University Press.

Sigurbjörnsson, B., and Kamps, J. (2005). The Effect of Structured Queries and Selective Indexing on XML Retrieval, In *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977*, 104-118

SGML. (1986). Information processing - text and office system - standard generalised markup language (sgml), iso/iec 8879. Technical report, International Organization for Standardization (ISO).

Sun, C., Chan, C., and Goenka, A. K. (2007). Multiway SLCA-based keyword search in XML data. In *Proceedings of the 16th International Conference on World Wide Web*, *WWW 2007*, 10 pages.

Skiena, S. (1990). *Partial Orders: §5.4 in Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Reading, MA: Addison-Wesley, 203-209.

Sparck Jones, K., Walker, S. and Robertson S.E. (2000). A probabilistic model of information retrieval: development and comparative experiments. Information Processing and Management 36, Parts 1 & 2, 779-840.

Tatarinov, I., Viglas, S. D., Beyer, K., Shanmugasundaram, J., Shekita, E., and Zhang, C. (2002). Storing and querying ordered XML using a relational database system. In *Proceedings of the International Conference on Management of Data, SIGMOD 2002*. 204-215.

Theobald, M., and Weikum, G. (2002). The index-based XXL search engine for querying XML data with relevance ranking, In *Proceedings of 8th International Conference on Extending Database Technology, EDBT 2002*. 477–495.

Theobald, M., Schenkel, R., and Weikum, G. (2005). An Efficient and Versatile Query Engine for TopX Search, In *Proceedings of the 31st Conference on Very Large Data Bases, VLDB 2005*, 625-636.

Trotman, A. Narrowed Extended XPath I. (2009). *Encyclopedia of Database Systems 2009*. 1876-1880.

Trotman, A., and Sigurbjörnsson, B. (2005). NEXI, now and next. In *Proceedings of the 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Lecture Notes in Computer Science 3493,* 41-53.

Trotman, A. (2009). Processing Structural Constraints. *Encyclopedia of Database Systems 2009*. 2191-2195.

Trotman, A., and Lalmas, M. (2006). Strict and vague interpretation of XML-retrieval queries. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in information Retrieval, SIGIR 2006*, 709-710.

Trotman, A., Pharo, N., and Lehtonen, M. (2007). XML-IR Users and Use Cases. In *Proceedings of the 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006, Lecture Notes in Computer Science 4518*, 400-412.

Voorhees, E., and Harman, D. (2005). *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press.

de Vries, A.P., Kazai, G., and Lalmas, M. (2004). Tolerance to Irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *Proceedings of the Conference on Coupling Approaches, Coupling Media and Coupling Languages for Information Retrieval*, *RIAO 2004*, 463-473.

Witt, A., and Metzing, D. (2010). *Linguistic Modeling of Information and Markup Languages: Contributions to Language Technology*, Springer.

Zhao, L. and Callan, J. (2009). Effective and efficient structured retrieval. In *Proceeding of the 18th ACM Conference on information and Knowledge Management, CIKM 2009*, 1573-1576.

Zobel, J., Moffat, A., and Ramamohanarao, K. (1998). Inverted files versus signature files for text indexing. *ACM Transactions on Database Systems*. 23(4), 453–490.

van Zwol, R., Baas, J., Van Oostendorp, H., and Wiering, F. (2006). Query formulation for XML retrieval with bricks, In *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977*, 80-88.

Study I

Jaana Kekäläinen, Marko Junkkari, Paavo Arvola, Timo Aalto (2005) TRIX 2004 - struggling with the overlap. In *Proceedings of the 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Lecture Notes in Computer Science 3493*, Springer-Verlag Berlin Heidelberg, 127-139.

TRIX 2004 – Struggling with the Overlap

Jaana Kekäläinen[1], Marko Junkkari[2], Paavo Arvola[2], and Timo Aalto[1]

[1] University of Tampere, Department of Information Studies,
33014 University of Tampere, Finland
{jaana.kekäläinen, timo.aalto}@uta.fi
[2] University of Tampere, Department of Computer Sciences,
33014 University of Tampere, Finland
marko.junkkari@cs.uta.fi, paavo.arvola@uta.fi

Abstract. In this paper, we present a new XML retrieval system prototype employing structural indices and a *tf***idf* weighting modification. We test retrieval methods that a) emphasize the *tf* part in weighting and b) allow overlap in run results to different degrees. It seems that increasing the overlap percentage leads to a better performance. Emphasizing the *tf* part enables us to increase exhaustiveness of the returned results.

1 Introduction

In this report, we present an XML retrieval system prototype, TRIX (Tampere retrieval and indexing system for XML), employing structural indices and a *tf***idf* weighting modification based on BM25 [3], [10]. The system is aimed for full scale XML retrieval. Extensibility and generality for heterogeneous XML collections have been the main goals in designing TRIX. This prototype is able to manipulate content_only (CO) queries but not content_and_structure (CAS) queries. However, with the CO approach of TRIX we achieved tolerable ranking for VCAS runs in INEX 2004.

One idea of XML is to distinguish the content (or data) element structure from stylesheet descriptions. From the perspective of information retrieval, stylesheet descriptions are typically irrelevant. However, in the INEX collection these markups are not totally separated. Moreover, some elements are irrelevant for information retrieval. We preprocessed the INEX collection so that we removed the irrelevant parts from the collection. The main goal of the preprocessing of the INEX collection was to achieve a structure in which the content element has a natural interpretation. In the terminology of the present paper, the content element means an element that has own textual content. The ranking in TRIX is based on weighting the words (keys) with a *tf***idf* modification, in which the length normalization and *idf* are based on content elements instead of documents.

The overlap problem is an open question in XML information retrieval. On one hand, it would be ideal that the result list does not contain overlapping elements [7]. On the other hand, the metrics of INEX 2004 encourage for a large overlap among results. In this paper, we introduce how the ranking of runs depends on the degree of overlap. For this, we have three degrees of overlap:

N. Fuhr et al. (Eds.): INEX 2004, LNCS 3493, pp. 127 – 139, 2005.

1. No overlapping is allowed. This means that any element is discarded in the ranking list if its subelement (descendant) or superelement (ancestor) is ranked higher in the result list.
2. Partial overlapping is allowed. The partial overlapping means that the immediate subelements and superelement are not allowed in the result list relating to those elements which have a higher score.
3. Full overlapping is allowed.

In this report we present the performance of two slightly different scoring schemes and three different overlapping degrees for both CO and VCAS tasks. The report is organized as follows: TRIX is described in Section 2, the results are given in Section 3, and discussion and conclusions in Sections 4 and 5 respectively.

2 TRIX 2004

2.1 Indices

The manipulation of XML documents in TRIX is based on the structural indices [4]. In the XML context this way of indexing is known better as Dewey ordering [11]. To our knowledge the first proposal for manipulating hierarchical data structures using structural (or Dewey) indices is found in [9]. The idea of structural indices is that the topmost element is indexed by ⟨1⟩ and its immediate subelements by ⟨1,1⟩, ⟨1,2⟩, ⟨1,3⟩ etc. Further the immediate subelements of ⟨1,1⟩ are labeled by ⟨1,1,1⟩, ⟨1,1,2⟩ ⟨1,1,3⟩ etc. This kind of indexing enables analyzing any hierarchal data structure in a straightforward way. For example, the superelements of the element labeled by ⟨1,3,4,2⟩ are found from indices ⟨1,3,4⟩, ⟨1,3⟩ and ⟨1⟩. In turn, any subelement related to the index ⟨1,3⟩ is labeled by ⟨1,3,ξ⟩ where ξ is a non-empty subscripts of the index.

In TRIX we have utilized structural indices in various tasks. First, documents and elements are identified by them. Second, the structure of the inverted file for elements is based on structural indices. Third, algorithms for degrees of overlapping are based on them. A detailed introduction to Dewey ordering in designing and manipulating inverted index is given in [1].

2.2 Weighting Function and Relevance Scoring

The content element is crucial in our weighing function. In this study, the content element is an element that has own textual content but none of its ancestors possess own textual content. Content elements are index units. For example, if the paragraph level is the highest level in which text is represented then paragraphs are manipulated as content elements and their descendants are not indexed. Content elements are chosen automatically for each document in the indexing process.

In TRIX the weighting of keys is based on a modification of the BM25 weighting function [3], [10]. Related to a single key k in a CO query the weight associated with the element e is calculated as follows:

$$w(k,e) = \frac{kf_e}{kf_e + v \cdot ((1-b) + b \cdot l\_norm(k,e))} \cdot \frac{\log\left(\frac{N}{m}\right)}{\log N} \quad (1)$$

where

- kf_e is the number of times k occurs in element e,
- m is the number of content elements containing k in the collection,
- N is the total number of content elements in the collection,
- v and b are constants for tuning the weighting,
- $l\_norm(k,e)$ is a normalization function defined based on the ratio of the number (ef_c) of all descendant content elements of the element e, and the number (ef_k) of descendant content elements of e containing k. If the element e is a content element then $l\_norm(k,e)$ yields the value 1. Formally, the length normalization function is defined as follows:

$$l\_norm(k,e) = \begin{cases} 1, \text{if } e \text{ is a content element} \\ efc / \sqrt{efk}, \text{otherwise} \end{cases} \quad (2)$$

The weighting formula 1 yields weights scaled into the interval [0,...,1].

TRIX does not support proximity searching for phrases. Instead, we require that each key k_i ($i \in \{1,\ldots,n\}$) in a phrase $p =$"$k_1,\ldots,k_n$" must appear in the same content element. This is a very simple approximation for weighting of phrases but it works well when content elements are short – such as paragraphs and titles.

Related to the element e the weight of the phrase p is is calculated as follows:

$$w(p,e) = \frac{\min(p,e)}{\min(p,e) + v \cdot \left((1-b) + b \cdot lp\_norm(p,e)\right)} \cdot \frac{\log\left(\frac{N}{m_p}\right)}{\log N} \quad (3)$$

where

- $\min(p,e)$ gives the lowest frequency among the keys in p in the element e,
- m_p is the number of content elements containing all the keys in p.
- v, b, N and ef_c have the same interpretation as in formula 1.
- lp_norm where ef_p is the number of descendant content elements of e containing all the keys in p,

$$lp\_norm(p,e) = \begin{cases} 1, \text{if } e \text{ is a content element} \\ efc / \sqrt{efp}, \text{otherwise} \end{cases} \quad (4)$$

In CO queries, a query fragment or sub-query (denoted by sq below) is either a key or phrase with a possible +/- prefix. The '+' prefix in queries is used to emphasize the importance of a search key. In TRIX the weight of the key is increased by taking a square root of the weight:

$$w(+sq,e) = \sqrt{w(sq,e)} \quad (5)$$

The square root increases the weight related to the element *e* and the query fragment *sq* (either *k* or *p*) because the weight of a query fragment is scaled between 0 and 1.

The '-' prefix in queries denotes an unwanted key. In TRIX the weight of such a key is decreased by changing the weight to its negation. For any key or phrase *sq* the minus expression -*sq* is weighted by the negation of the original weight as follows:

$$w(-sq,e) = -w(sq,e) \tag{6}$$

In other words, unwanted query fragments are manipulated in the interval [-1,0].

For combination of query fragments (with a possible +/- subscript) two operation have been implemented: average and a fuzzy operation called Einstein's sum [8]. Using the average the weight $w(q,e)$ related to the CO query $q = sq_1 \ldots sq_n$ is formulated as follows:

$$w(q,e) = \frac{\sum_{i=1}^{n} w(sq_i,e)}{n} \tag{7}$$

The other implemented alternative, Einstein's sum (denoted by $\oplus$), means that two weights w_1 and w_2 are combined as follows:

$$w_1 \oplus w_2 = \frac{w_1 + w_2}{1 + w_1 \cdot w_2} \tag{8}$$

Unlike the average the operation $\oplus$ is associative, i.e. $w_1 \oplus w_2 \oplus w_3 = (w_1 \oplus w_2) \oplus w_3 = w_1 \oplus (w_2 \oplus w_3)$. Thus, the weight (denoted by *w'*) of a CO query $q = sq_1\ sq_2 \ldots sq_n$ can be calculated follows:

$$w'(q,e) = w(sq_1,e) \oplus w(sq_2,e) \oplus \cdots \oplus w(sq_n,e) \tag{9}$$

To illustrate this function we apply it to topic 166 (+"tree edit distance" +xml -image) for an element *e*:

w'(+"tree edit distance" +xml -image, *e*)

First, Equation 9 is applied as follows:

w(+"tree edit distance", *e*) ⊕ *w*(+xml, *e*) ⊕ *w*(-image, *e*)

Then, Equations 5 and 6 are used (sqrt means square root in Equation 5)

sqrt(*w*("tree edit distance", *e*)) ⊕ sqrt(*w*(xml, *e*)) ⊕ -*w*(image, *e*)

Now, *w*("tree edit distance", *e*) is calculated using Equation 3 and the others using Equation 1.

2.3 Implementation

The TRIX is implemented in C++ for Windows/XP but the implementation is aimed for UNIX/LINUX as well. In implementing the present TRIX prototype we have paid

attention for effective manipulation of XML data structures based on structural indices. However, the efficiency has not been the main goal of TRIX.

The TRIX prototype has two modes: online mode and batch mode. In the online mode the user can run CO queries in the default database (XML collection). The batch mode enables running a set of CO queries. In this mode queries are saved in a text file. Running the CO queries of INEX 2004 in the batch mode takes about 40 minutes in a sample PC (Intel Pentium 4, 2.4 GHz, 512MB of RAM). The weights are calculated at query time for every element. The size of the inverted index is 174 MB.

The command-based user interface of the TRIX prototype is tailored for testing various aspects of XML information retrieval. This means that a query can be run with various options. For example, the user can select:

- the method (average or Einstein's sum) used in combining the query term weights,
- the degree of overlap (no overlapping, partial overlapping or full overlapping), and
- the values of the constants.

For example, the command string

```
TRIX –e –o b=0.1 queries2004co.txt
```

means that Einstein's sum is used for combining weights (parameter -e), full overlapping is allowed (parameter -o) and the *b* is 0.1. Finally, queries2004co.txt denotes the file from which the query set, at hand, is found. Actually, there is no assumption of ordering for the parameters of a query. For example, the command string

```
TRIX –o queries2004co.txt b=0.1 –e
```

is equivalent with the previous query.

The online mode of TRIX is chosen by the command

```
TRIX
```

After this command the user may give his/her query, e.g.:

```
+"tree edit distance" +xml -image
```

3 Data and Results

We preprocessed the INEX collection so that from a retrieval point of view irrelevant parts were removed. As irrelevant content we considered elements consisting of non-natural language expressions, e.g. formulas, abbreviations, codes. We classified irrelevant parts into three classes. First, there are elements which possess relevant content but the tags are irrelevant. Tags which only denote styles, such as boldface or italic, inhere in this class. These tags were removed but the content of elements was maintained. Second, there are elements whose content is irrelevant but their tags are necessary in order to maintain the coherent structure of documents. For example we appraised the content of <sgmlmath> and <tmath> elements to inhere in this class. Third, there are elements having irrelevant content whose tags are not necessary in

structural sense. These elements, such as <doi> and <en>, were removed. The selection of the parts to be removed was done by researchers, the removal was automatic.

For INEX 2004 we submitted both CO and VCAS runs though our system actually supports only CO queries. In both cases, the title field was used in automatic query construction. Phrases marked in titles were interpreted as 'TRIX phrases' in queries, i.e. all the phrase components were required to appear in the same element. In addition, all the components were added as single keys to queries. For example, topic 166 is formulated into a query as follows:

```
+"tree edit distance" +xml -image tree edit distance
```

In VCAS queries the structural conditions were neglected and all keys were collected into a flat query. Word form normalization for the INEX collection and queries was Porter stemming, and a stoplist of 419 words was employed.

3.1 Tuning Constants

Setting the values of the constants v and b in the weighting function has an impact on the size of elements retrieved. For analyzing this impact, v was tested with values 1 and 2, and b was varied between 0 and 1. The value $v = 2$ gave better performance than $v = 1$, and the former is thus used as default now on. We ran the CO queries using average scoring, no-overlap and full overlap with different values of b. Then, the result lists were analyzed for the percentage of different element types at document cut-off values (DCV) 100 and 1500. Our categorization was rather coarse; percentages of articles, sections, abstracts, paragraphs and others ware calculated in the result lists. Category *section* contains sections and 'equivalent elements' (see [5]); category *paragraph* contains paragraphs and equivalent elements. Only DCV 100 is reported below because DCV 1500 gave very similar results.

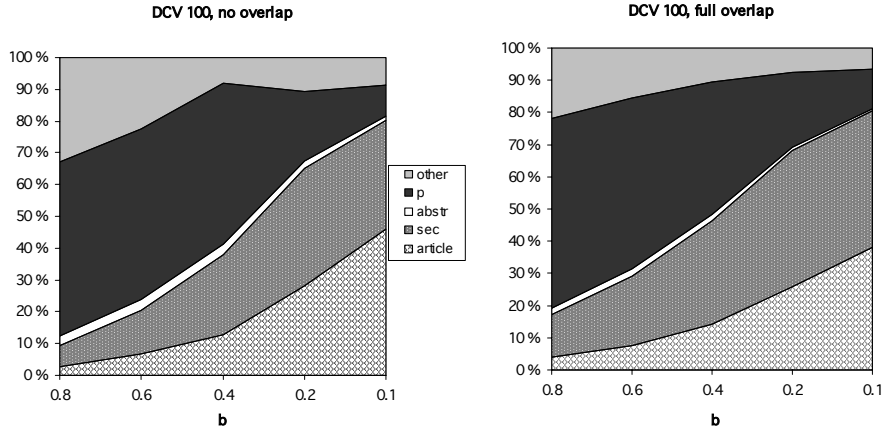

Fig. 1. Percentages of elements of different size in the result sets when b is varied

Figure 1 illustrates the change in the size of retrieved elements when *b* is varied between 0.8 and 0.1. The percentage of the smaller units increases from *b* value 0.1 to *b* value 0.8. In other words large *b* values promote small elements and the small b values promote large elements. This is due to strengthening the *tf* part in the weighting scheme by weakening length normalization. Although our categorization is rough and the category 'other' includes also large elements, the trend is visible. The change is apparent with and without overlap. In our official submissions *b* was 0.4. However, later tests revealed, that *b* value 0.1 gives better performance. Results with both values of *b*, 0.1 and 0.4, are reported in the following sections.

3.2 CO Runs

The evaluation measure used in INEX 2004 was precision at standard recall levels (see [2]) with different quantizations for relevance dimensions (see Relevance Assessment Guide elsewhere in these Proceedings). In the strict quantization only those elements that are highly exhaustive and highly specific are considered relevant, others non-relevant. In other quantization functions elements' degree of relevance is taken into account by crediting elements according to their level of specificity and exhaustiveness. (For details of metrics, see [12] or Evaluation metrics 2004 in these Proceedings.) The results are based on the topic set with relevance assessements for 34 topics. Our official submissions were:

1. CO_avrg: run using *w* weighting (average) with no overlapping when $b = 0.4$,
2. CO_Einstein: run using *w'* weighting (Einstein's sum) with no overlapping when $b = 0.4$,
3. CO_avrg_part_overlap: run using *w* weighting with partial overlapping when $b = 0.4$.

The results for 1 and 2 were so similar that we report the results based on average only. Further, in our official submissions two overlap degrees were tested: no overlapping and partial overlapping. Later on we added the full overlapping case.

Table 1. Mean average precision (MAP) and ranking of CO runs with average scoring

	b	MAP	Rank
No overlapping	0.4	0.0198	45
	0.1	0.0239	42
Partial overlapping	0.4	0.0443	31
	0.1	0.0487	25
Full overlapping	0.4	0.0831	11
	0.1	0.0957	10

Aggregate precision values, given in Table 1, are macro-averages over the different quantizations used in INEX 2004. Table 1 shows the effect of different overlaps and tuning of *b* to aggregate precision and rank. Decreasing *b* has a slight positive effect on the aggregate score and rank. When the different metrics are considered, it is

obvious that small *b* values enhance the dimension of exhaustiveness at specificity's expense. Figures 4 - 5 in Appendix show P-R-curves for CO runs with specificity- and exhaustiveness-oriented quantizations. In case of specificity-oriented quantization (Figures 4a-b and 5 a-b) average precision decreases as *b* decreases. Figures 4c-d and 5c-d in the appendix show an exhaustiveness-oriented quantization, and there average precision increases as *b* decreases. The mean average precision figures with all quantizations for our official submissions are given in Table 2.

Table 2. MAP figures for University of Tampere official CO submissions, $b = 0.4$

	MAP						
	strict	gen.	so	s3_e321	s3_e32	e3_s321	e3_s32
CO_avg	0.022	0.016	0.015	0.016	0.017	0.026	0.026
CO_Einstein	0.023	0.016	0.015	0.014	0.017	0.034	0.029
CO_avg_po	0.044	0.041	0.041	0.039	0.042	0.051	0.054

The effect of overlap is more substantial: allowing the full overlapping changes the aggregate rank from 45th to 11th when $b = 0.4$, or from 42nd to 10th when $b = 0.1$. Figure 2 illustrates the increase in the aggregate score when overlap percentage increases. (No overlap 0%; partial overlap 40%/44%; full overlap 63%/69%. Compare also Figures 4a and 5a, and 4b and 5b, etc. in Appendix). Whether the change in the result lists is desirable from the user's point of view is questionable because it means returning several overlapping elements from the same document in a row.

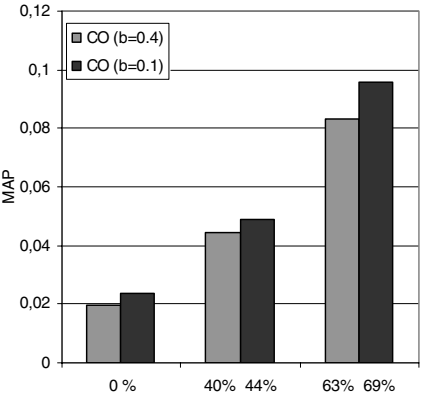

Fig. 2. Mean average precision and overlap percentage of CO runs with average scoring

3.3 VCAS Runs

The results of the VCAS runs are very similar to CO runs. Decreasing *b* value gives better exhaustivity-oriented results but impairs specificity. Increasing the overlap enhances effectiveness. Both these tactics have a positive effect on the aggregate score (see Table 3).

Table 3. Mean average precision and ranking of VCAS runs with average scoring

	b	MAP	Rank
No overlapping	0.4	0.269	30
	0.1	0.031	30
Partial overlapping	0.4	0.038	25
	0.1	0.042	22
Full overlapping	0.4	0.061	11
	0.1	0.075	7

Figure 3 shows the overlap percentages for different VCAS runs. Also here the benefits of allowing the overlap are evident though not as strong as with CO queries.

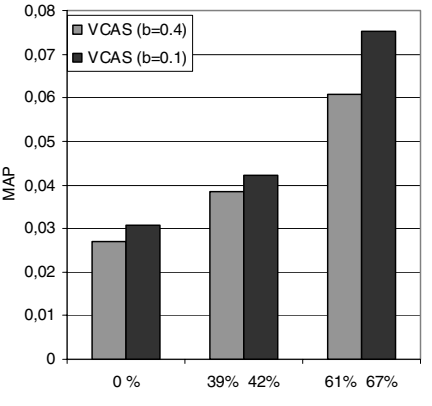

Fig. 3. Mean average precision and overlap percentage of CAS runs with average scoring

4 Discussion

In INEX 2004 University of Tampere group was struggling with the overlap. Our basic design principle was not to allow overlap in the result lists. Because of the structural indices of our system overlap is easy to eliminate. However, the reports of the previous INEX workshop led us to test effectiveness of partial overlap. Because of improved performance, we tested several runs with and without overlap, and allowing full overlap yielded the best performance. Nevertheless, the overlap percentage,

showing the percentage of elements that have either a superelement or a subelement ranked higher in the result list, is almost 70 in case of full overlap. This means that in the result list of 10 elements only 3 elements are 'totally new' for the user. It seems that the INEX metrics encourage returning overlapping elements though this might not be beneficial for the user. Our original idea of eliminating overlap was supported by an alternative measure, addressing the problem of overlapping relevant elements, proposed in [6]. The measure, XCG, ranked our runs without overlap higher than runs with overlap.

Our retrieval system, TRIX, employs a modification of *tf*idf* weighting. The number of content subelements is used in element length normalization. In the present mode, TRIX only supports CO queries but we aim at introducing a query language for content and structure queries. Because only titles of the topics – providing a very terse description of the information need – were allowed in query construction, and we did not expand the queries, a mediocre effectiveness was to be expected. Since TRIX does not support querying with structural conditions we submitted VCAS runs processed similarly as CO runs. Surprisingly our success with the VCAS task was not worse than with the CO task. However, if structural conditions are not considered when assessing the relevance, it is understandable that CO and VCAS tasks resemble each other.

Our further work with TRIX is aimed at introducing a query expansion or enhancing module. Incapability to deal with short content queries is a well-known disadvantage. Also, a CAS query language allowing also document restructuring is under construction.

5 Conclusion

In this paper we presented a *tf*idf* modification for XML retrieval. Instead of normalization based on the length of documents or elements we proposed a normalization function based on the number of content elements. We have shown how the well-known BM25 method, primarily intended to full-text information retrieval, can be applied to favor different sizes of XML elements. This sizing of result elements also has effects on the performance of queries. As our study indicates the performance strongly depends on the degree of overlap when such metrics as in INEX 2004 are used. The redundancy in returned elements might not serve the user. Therefore, if the user point of view is taken into account, nåw measures are needed.

Acknowledgments

This research was supported by the Academy of Finland under grant number 52894.

References

1. Guo, L., Shao, F., Botev, C. and Shanmugasundaram, J.: XRANK: Ranked Keyword Search over XML Documents. In: Proc. of ACM SIGMOD 2003, San Diego, CA (2003) 16-27

2. Gövert, N., and Kazai, G. Overview of the INitiative for the Evaluation of XML retrieval (INEX) 2002. In: Proc. of the First Workshop of the Initiative for the Evaluation of XML Retrieval (INEX), (2002), pp. 1-17. Retrieved 27.1.2005 from http://qmir.dcs.qmul.ac.uk/inex/Workshop.html
3. Hawking, D., Thistlewaite, P., and Craswell, P. ANU/ACSys TREC-6 experiments. In: Proc. of TREC-6, (1998). Retrieved 10.3.2004 from http://trec.nist.gov/pubs/trec6/papers/anu.ps
4. Junkkari, M. PSE: An object-oriented representation for modeling and managing part-of relationships. Journal of Intelligent Information Systems, to appear.
5. Kazai, G. Lalmas, M, and Malik, S. INEX´03 guidelines for topic developmen. In: INEX 2003 Workshop Proc., (2003), pp. 192-199. Retrieved 21.1.2005 from http://inex.is.informatik.uni-duisburg.de:2003/internal/downloads/INEXTopicDevGuide.pdf
6. Kazai, G., Lalmas, M., and de Vries, A.P. Reliability tests for the XCG and INEX-2002 metrics. In INEX 2004 Workshop Pre-Proc. (2004), pp. 158-166. Retrieved 18.1.2005 from http://inex.is.informatik.uni-duisburg.de: 2004/pdf/INEX2004PreProceedings.pdf
7. Kazai, G., Lalmas, M., and de Vries, A.P. The overlap problem in content-oriented XML retrieval evaluation. In Proc. of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK, (2004), pp.72-79.
8. Mattila, J.K. Sumean Logiikan Oppikirja: Johdatusta Sumean Matematiikkaan. Art House, Helsinki, (1998).
9. Niemi, T. A seven-tuple representation for hierarchical data structures. Information systems, 8, 3 (1983), 151-157.
10. Robertson S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., and Gatford, M. Okapi at TREC-3. In: NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3), (1994). Retrieved 21.11.2004 from http://trec.nist.gov/pubs/trec3/papers/city.ps.gz
11. Tatarinov, I., Viglas, S.D., Beyer, K., Shanmugasundaram, J., Shekita, E., and Zhang, C. Storing and querying ordered XML using a relational database system. In Proc. of the SIGMOD Conference, (2002), pp. 204-215.
12. de Vries, A.P, Kazai, G., and Lalmas, M. Evaluation metrics 2004. In INEX 2004 Workshop Pre-proc., (2004), pp. 249-250. Retrieved 18.1.2005 from http://inex.is.informatik.uni-duisburg.de:2004/pdf/INEX2004PreProceedings.pdf

Appendix

Precision-Recall Curves for CO Queries

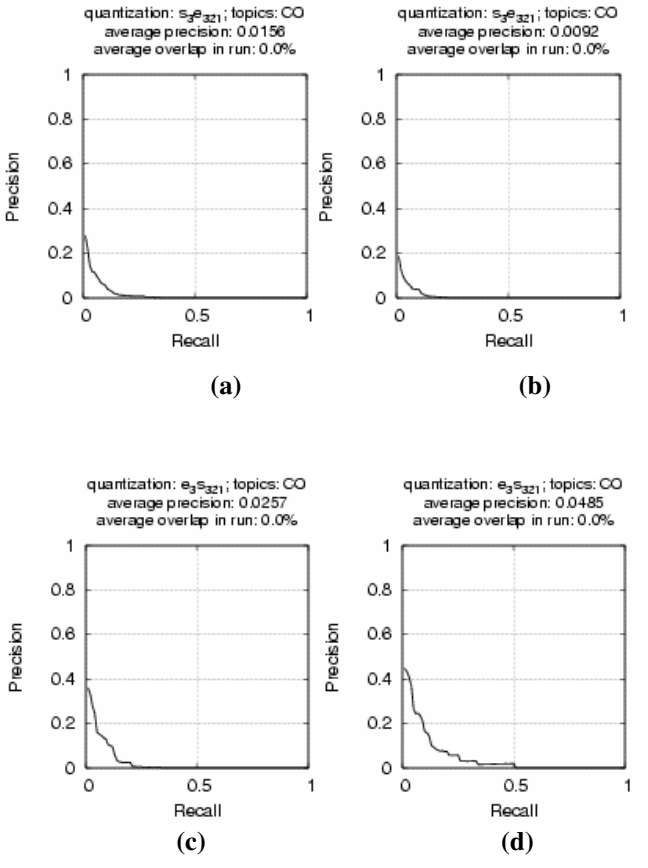

Fig. 4. CO without overlap. Quantization: s_3e_{321} (a) $b = 0.4$, rank 39/70; (b) $b = 0.1$, rank 46/70. Quantization e_3s_{321} (c) $b = 0.4$, rank 45/70 ; (d) $b = 0.1$, rank 39/70

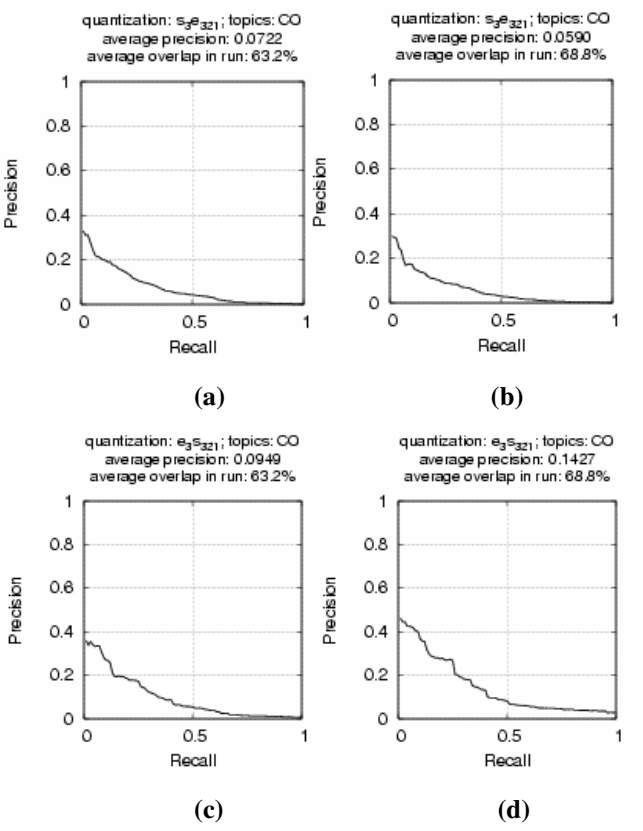

Fig. 5. CO with full overlap. Quantization: s_3e_{321} (a) $b = 0.4$, rank 8/70; (b) $b = 0.1$, rank 12/70. Quantization: e_3s_{321} (c) $b = 0.4$, rank 17/70; (d) $b = 0.1$, rank 11/70

Study II

Paavo Arvola, Marko Junkkari, Jaana Kekäläinen (2005) Generalized contextualization method for XML information retrieval. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management, CIKM 2005,* 20-27.

Generalized Contextualization Method for XML Information Retrieval

Paavo Arvola
Dept. of Information Studies
University of Tampere
Finland
paavo.arvola@uta.fi

Marko Junkkari
Dept. of Computer Sciences
University of Tampere
Finland
marko.junkkari@cs.uta.fi

Jaana Kekäläinen
Dept. of Information Studies
University of Tampere
Finland
jaana.kekalainen@uta.fi

ABSTRACT
A general re-weighting method, called contextualization, for more efficient element ranking in XML retrieval is introduced. Re-weighting is based on the idea of using the ancestors of an element as a context: if the element appears in a good context – good interpreted as probability of relevance – its weight is increased in relevance scoring; if the element appears in a bad context, its weight is decreased. The formal presentation of contextualization is given in a general XML representation and manipulation frame, which is based on utilization of structural indices. This provides a general approach independent of weighting schemas or query languages.

Contextualization is evaluated with the INEX test collection. We tested four runs: no contextualization, parent, root and tower contextualizations. The contextualization runs were significantly better than no contextualization. The root contextualization was the best among the re-weighted runs.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *retrieval models.* H.3.4 [**Information Storage and Retrieval**]: Systems and Software – *performance evaluation.* H.2.1 [**Database Management**]: Logical Design - *data models.* E.1 [**Data**]: Data Structures – *trees*. E.5 [**Data**]: Files – *organization/structure*.

General Terms
Management, Measurement, Performance, Design, Experimentation, Languages.

Keywords
XML, Structured documents, Semi-structured data, Re-weighting, Contextualization, Structural indices, Dewey ordering.

CIKM'05, October 31-November 5, 2005, Bremen, Germany.

1. INTRODUCTION
XML retrieval deals with the possibility to utilize hierarchical document structure in returning more specific text units than whole documents to users [2]. Thus, it is a step towards information retrieval instead of document retrieval. Text units derive part of their meaning from the context in which they appear. The previous and the following passages in a document determine the context of a text passage and give it an interpretative frame. A text passage can be viewed in contexts of different size. Any part of a document which contains the passage is a possible context in which the passage can be viewed. Usually documents, e.g. scientific articles, involve an intrinsic structure in which a document is divided into sections, subsections and paragraphs. This established division gives natural contexts of different size. For example, a paragraph can be viewed in the context of an article, a section, and possible subsections. XML serves as a way to organize structured documents and to manipulate different levels of context. However, in XML retrieval only a little attention has been paid to the context in which the element appears. The context affects the interpretation of the element and gives hints about its relevance. Therefore, we propose a general re-weighting method for XML retrieval that takes into account the specified contexts of elements.

An XML document consists of elements, which in turn may contain smaller elements. If an element x contains another element y then x is called an ancestor of y, whereas y is called a descendant of x. Typically, in weighting of an element the weights of its descendants affect the weight of the element [e.g. 3, 12]. This approach has also been applied to the weighting of text passages in non-XML text retrieval [e.g. 1, 8]. Instead, there are only a few proposals where the weights of the ancestors of an element would be taken into account in weighting of the element [20, 14]. This approach seems reasonable because XML elements are not independent of their ancestors. Sigurbjörnsson, Kamps and de Rijke [20] propose that the weight of an article should affect the weighting of any of its elements. Based on [15] Ogilvie and Callan [18] combine evidence from an element's parent and children in estimation of a language model for the element. We propose a method in which any hierarchy level of ancestors may be taken into account in the weighting of elements. The proposed method supports both increasing and decreasing the weights of elements. Likewise, our approach is not fixed to any collection, or predefined XML structure, or weighting method, or specific query language. We aim at general formal presentation that allows

defining a context for any element and using this context as evidence of the relevance of the element.

The DTD independent manipulation of XML documents requires the capacity of transitive reasoning. In the present paper this is based on the *structural indices,* which have a long history in representing and manipulation of hierarchical data structures. They have been applied in the context of the hierarchical data model [16], the NF2 relational model [17], and composed objects in the object-oriented and deductive object-oriented data models [7]. Similar to these models, the XML data model requires management of hierarchical data structures. Therefore it is not surprising that a similar method based on Dewey decimal indexing has earlier been applied to representing XML structures [e.g. 21]. In this study we use structural indices for designing an inverted file for an XML collection, query evaluation, and re-weighting. In order to avoid ambiguity we refer to a structural index as an index; in contrast, an inverted index, typical for text retrieval, is referred to as an inverted file.

In this paper we develop and present a re-weighting method called *contextualization*. For this we give a general contextualization function and its sample extensions used in the test environment. Our background assumption is that a text passage in a relevant context should be ranked higher than a similar passage in a non-relevant context. Our formal presentation is based on the standard set theory, which is an established and general representation approach. In Section 2 we present our XML retrieval system and in Section 3 the test setting. In Section 4 we give the results obtained with INEX test collection. Discussion and Conclusions are given in Section 5.

2. INDEXING, WEIGHTING AND CONTEXTUALIZING METHODS

2.1 Structural Indices and XML Documents

The idea of structural indices in the context of XML is that the topmost (root) element is indexed by ⟨1⟩ and its children by ⟨1,1⟩, ⟨1,2⟩, ⟨1,3⟩ etc. Further, the children of the element with the index ⟨1,1⟩ are labeled by ⟨1,1,1⟩, ⟨1,1,2⟩, ⟨1,1,3⟩ etc. This kind of indexing enables analyzing of the relationships among elements in a straightforward way. For example, the ancestors of the element labeled by ⟨1,3,4,2⟩ are associated with the indices ⟨1,3,4⟩, ⟨1,3⟩ and ⟨1⟩. In turn, any descendant related to the index ⟨1,3⟩ is labeled by ⟨1,3,ξ⟩ where ξ is a non-empty part of the index.

In the present approach the XML documents in the collection are labeled by positive integers 1, 2, 3, etc. From the perspective of indexing this means that the documents are identified by indices ⟨1⟩, ⟨2⟩, ⟨3⟩, etc., respectively. In other words, each index ⟨*i*⟩ ($i \in \{1,2,3,\ldots\}$) refers to a root element and its descendants are indexed by the way described above. Now each document involves an index structure in which each index is initiated with the document identifier. This means that each element possesses a unique index and we manipulate the XML collection via one index set. For example, let us assume that the following XML document is labeled by 5 then it involves the indices ⟨5⟩ (root index or the index of `article`), ⟨5,1⟩ (the index of `abstract`), ⟨5,2⟩ (the index of `section`), and ⟨5,2,1⟩ (the index of `paragraph`).

```
<article>
 <abstract> This is the content of the
   abstract.</abstract>
 <section>
   <paragraph> Here is the content of this
    paragraph.</paragraph>
 </section>
</article>
```

A basic concept in our system is a *content element*, an element that has own textual content. Here we have made a deliberate practical choice to use the topmost content elements as the least units to index. These are treated as leaves of the XML tree, i.e. their children are not indexed. This solution behaves well in the used test collection (see Section 3.1) where content elements, defined in this way, possess a natural granularity, for example, paragraphs and titles. In other words, this solution prevents division into too small fragments, such as single words or parts of words, mostly result from style-sheet marking. For example the following `abstract` element is interpreted as a content element, i.e. the elements `bold` and `italic` are not indexed. The content of this abstract element is interpreted without tags, or analogously with the abstract element above.

```
<abstract> This is the <bold>content</bold>
of the <italic>abstract</italic>.</abstract>
```

Indexing also gives a sound and efficient foundation for designing an inverted file. Namely, each key in the inverted file may be associated with the set of the structural indices of the elements where the key appears. Actually, no more than the occurrences in content elements must be stored because indirect occurrences can be inferred based on structural indices. For example, if a key has an occurrence in the content element labeled by ⟨1,3,4,2⟩ then it has also an occurrence in the elements labeled by ⟨1,3,4⟩, ⟨1,3⟩, and ⟨1⟩. In the inverted file each key possesses the set of indices of the content elements where the key occurs. In the inverted file an index may involve information on the number of the key occurrences and their positions in the text. In this paper it is assumed that only the number of occurrences is stored. This means that the inverted file can be represented as a binary relation, called *IF*, consisting of tuples ⟨*k*, *I*⟩ where *k* is a key and *I* is the set pairs ⟨ξ, *times*⟩. Here *times* is the number of *k* occurrences of the element indexed by ξ. For example related to the sample article element above the tuple ⟨*content*,{⟨⟨5,1⟩, 1⟩, ⟨⟨5,2,1⟩, 1⟩}⟩ belongs to the set *IF,* assuming that the key *'content'* does not occur in any other element in the collection at hand. Occurrences in elements other than content elements are calculated based on the occurrences of their content elements. In the running example element with the index ⟨5,2⟩ has one '*content'* occurrence and the element with the index ⟨5⟩ has two '*content*' occurrences.

In this paper the following notational conventions are associated with structural indices:

- A structural index (briefly index) is a tuple, denoted between angle brackets, consisting of positive integers ($\mathbb{Z}^+$). The symbol ξ is also used for denoting an index. An element possessing the index ξ is called the *ξ element*.

- The set of indices related to the XML collection at hand is denoted by *IS*.
- The length of an index ξ is denoted by $len(\xi)$. For example $len(\langle 1,2,2,3\rangle)$ is 4.
- The index $\langle i\rangle$ consisting of an integer i (i.e. its length is 1) is called *root index* and it is associated with a whole document.
- Let ξ be an index and i a positive integer then the *cutting operation* $\delta_i(\xi)$ selects the subindex of the index ξ consisting of its i first integers. For example if $\xi = \langle a,b,c\rangle$ then $\delta_2(\xi) = \langle a,b\rangle$. In terms of the cutting operation the root index at hand is denoted by $\delta_1(\xi)$ whereas the index of the parent element can be denoted by $\delta_{len(\xi)-1}(\xi)$. In turn, the index ξ' ($\in$ IS) is associated with a child element of the element with index ξ if $len(\xi') > len(\xi)$ and $\delta_{len(\xi)}(\xi') = \xi$, i.e. $\delta_{len(\xi')-1}(\xi') = \xi$.
- The function $content\_elem(\xi)$ gives the indices of the content elements related to the index ξ when ξ does not itself refer to a content element. If ξ refers to a content element then the function yields the set $\{\xi\}$. In terms of the cutting and length operations $content\_elem(\xi)$ is defined as follows:

$$\begin{cases} \{\xi\}, \text{ if } \neg\exists \xi' \in IS: \delta_{len(\xi')-1}(\xi') = \xi \\ \{\xi' \in IS \mid \exists i \in \mathbb{Z}^+: \delta_i(\xi') = \xi \wedge \neg\exists \xi'' \in IS: \\ \qquad \delta_{len(\xi'')-1}(\xi'') = \xi'\}, \text{ otherwise.} \end{cases}$$

- Let k be a search key then function $num\_of\_keys(k,\xi)$ yields the number of k occurrences in the ξ element. This is the sum of all the k occurrences in the contents of the elements with an index in $content\_elem(\xi)$. In terms of the inverted file *IF* described above, $num\_of\_keys(k,\xi)$ is defined as follows:

$$\sum_{\xi' \in content\_elem(\xi)} times \mid \exists \langle \xi', times\rangle \in I : \langle k, I\rangle \in IF$$

2.2 Contextualization

XML notation does not determine how a document collection should be organized. For example, a collection of documents could be represented as a complex XML element where the collection is the root element. However, de facto, XML collections are organized so that the main referable units, e.g. scientific articles, are represented by root elements. Instead, top hierarchy levels (collection/journal/volume/issue) are manipulated by a directory structure or they are aggregated into an additional XML element where a reference mechanism based on explicit identifiers is used. Further, the representation of documents in XML aims to follow the established structure of documents. For example, a scientific article is typically composed of sections which consist of subsections etc. The lowest level of XML hierarchy is usually designed so that it corresponds to the paragraph level in the source documents. This organization gives a natural starting point for manipulating text passages at the established hierarchy levels of text documents.

The idea of contextualization is based on the assumption that an element in a relevant context should be ranked higher than an identical element in a non-relevant context. Depending on how a collection is organized, an element may be viewed at various levels of context. For example, assuming that documents follow article-section-subsection-paragraph division, then the article, the section and the subsection form different levels of context for a paragraph. Further, a subsection can be viewed in the contexts of the section or the article. The length of the path from the context element to the element at hand determines the level of context. We say that the parent of an element determines the first level context; the ancestor with the path length 2 determines the second level context etc. The root element forms the topmost context. Let the present article and its sample XML representation below illustrate this.

```
<article>
…
<section sec_no="1">…</section>
<section sec_no="2">
…
<sub_section sec_no="2.1">…</sub_section>
…
<sub_section sec_no="2.2">
<title>Contextualization</title>
…
<p>Let us consider the present paragraph. Now Subsection 2.2 forms the first level context and Section 2 second level context of this paragraph. The article is the root element, or it determines the topmost context of this paragraph. In turn, Section 2 forms the first level context, and the article the second level (or topmost) context of Subsection 2.2. The article possesses no context.</p>
…
</sub_section>
</section>
…
</article>
```

Let us consider the present paragraph. Now Subsection 2.2 forms the first level context and Section 2 second level context of this paragraph. The article is the root element, or it determines the topmost context of this paragraph. In turn, Section 2 forms the first level context, and the article the second level (or topmost) context of Subsection 2.2. The article possesses no context.

In contextualization the weight of an element is modified by the basic weight of its context element(s). Contextualization is independent of the used query language and basic weighting schema for elements. Below we assume a basic weighting function $w(q,\xi)$ where q is a query expression and ξ is the index of the element to be weighted. In section 2.3 we specify a query language and a weighting function for the test environment.

We define a general contextualization function C which has the following arguments: q, ξ and g. The arguments q and ξ have the same interpretation as in the context of the basic weighting function w above. The argument g is called *contextualization vector* and set-theoretically it is represented as a tuple, consisting of values by which elements between the root element and ξ

element are weighted in contextualization. The length of g is $len(\xi)$. In referring to the i^{th} position of the contextualization vector g we use the notation $g[i]$. For example, if $g = \langle a,b,c \rangle$ then $g[2] = b$. The value in $g[i]$ relates to the element with the index $\delta_i(\xi)$. For example if $\xi = \langle 2,3,2 \rangle$ then g is the 3-tuple $\langle a,b,c \rangle$ where a is the contextualization weight of the root element (i.e. the element with index $\delta_1(\xi)$), b is the contextualization weight of the $\langle 2,3 \rangle$ element (i.e. the element with index $\delta_2(\xi)$), and c is the weight of the $\langle 2,3,2 \rangle$ element (i.e. the element with the index $\delta_{len(\xi)}(\xi)$). The contextualized weights of elements are calculated by weighted average based on contextualization vector and the index at hand. In the sample case above the contextualized weight is calculated as $(a * w(q, \langle 2 \rangle) + b * w(q, \langle 2,3 \rangle) + c * w(q, \langle 2,3,2 \rangle)) / (a + b + c)$. Contextualization is applied only to those elements whose basic weight is not zero. Next we define the general contextualization function C formally:

$$C(q, \xi, g) = \begin{cases} 0, \text{ if } w(q,\xi) = 0 \\ \dfrac{\sum_{i=1}^{len(\xi)} g[i] \cdot w(q, \delta_i(\xi))}{\sum_{i=1}^{len(\xi)} g[i]}, \text{ otherwise} \end{cases}$$

The values in g are not bound to any range. This means that in term of g, different levels of the context can be weighted in various ways. For example, weighting may increase or decrease towards to the topmost context (root element). In this paper, however, we consider only such cases where g consists of the values 1 and 0. Zero value means that the corresponding element is not taken into account in contextualization. Next we give extensions of the C function based on this weighting for the test setting below.

Related to a query expression q the contextualization based on the first level (parent) context of the ξ element is calculated using the contextualization vector where two last elements have the value 1 and the others zero value. This function is denoted $c_p(q, \xi)$ and it is defined as follows:

$c_p(q, \xi) = C(q,\xi,g)$ where $g =$

$$\begin{cases} g[len(\xi)] = 1 \\ g[len(\xi) - 1] = 1, \text{ when } len(\xi) > 1 \\ \sum_{i=1}^{len(\xi)-2} g[i] = 0, \text{ when } len(\xi) > 2 \end{cases}$$

The contextualization by the topmost context (or by the root element) is denoted by the function symbol c_r and it calculated in terms on the vector where the first and the last element have the weight 1 and the others zero value.

$c_r(q, \xi) = C(q,\xi,g)$ where $g =$

$$\begin{cases} g[len(\xi)] = 1 \\ g[1] = 1 \\ \sum_{i=2}^{len(\xi)-1} g[i] = 0, \text{ when } len(\xi) > 2 \end{cases}$$

The contextualization function c_t is called *tower contextualization* and it means that all the levels of context are taken into account. This is achieved by the contextualization vector where each position is valued by 1.

$c_t(q, \xi) = C(q,\xi,g)$ where $g = \left(\sum_{i=1}^{len(\xi)} g[i] = 1\right)$

When no contextualization is applied, the element gives its basic weight. This is denoted by the function $c_n(q, \xi)$ which is associated with the contextualization vector where the last position has the value 1 and the others the zero value.

$c_n(q, \xi) = C(q,\xi,g)$ where $g =$

$$\begin{cases} g[len(\xi)] = 1 \\ \sum_{i=1}^{len(\xi)-1} g[i] = 0, \text{ when } len(\xi) > 1 \end{cases}$$

We tested the proposed extensions of contextualization in the sample data (INEX 2004, see Section 3). Next, we introduce the used query language and its semantics based on indices; i.e. the weighting method formally.

2.3 Basic weighting schema

The present query language has features typical for query languages in best match retrieval systems. In it, search keys are separated from each other by a space and a phrase can be expressed between quotation marks. A key or phrase may involve +/- prefix to emphasize its importance or avoidance, respectively. The syntax of this language is given in Appendix 1. Next, we introduce the weighting-based semantics of this syntax. We give the weighting function w which is defined following the syntax expressions in Appendix 1. The function w involves two arguments – first for a query expression and second for the index at hand.

The weighting of keys is based on a modification of the BM25 weighting function [5, 19, see also 11]. The weight for the key k related to the index ξ is calculated as follows:

$$w(k, \xi) = \frac{kf_\xi}{kf_\xi + v \cdot \left((1 - b) + b \cdot \frac{\xi f_c}{\xi f_k} \right)} \cdot \frac{log\left(\frac{N}{m}\right)}{log(N)}$$

where

- kf_ξ is the number of times k occurs in the ξ element, i.e. $kf_\xi = num\_of\_keys(k,\xi)$,

- N is the total number of content elements in the collection, i.e. $N =$
$$\left| \bigcup_{\langle i \rangle \in IS} content\_elem(\langle i \rangle) \right|$$
- m is the number of content elements containing k in the collection, i.e. $m =$
$$\left| \bigcup_{\langle i \rangle \in IS} \{\xi' \in content\_elem(\langle i \rangle) \mid num\_of\_keys(k, \xi') \neq 0\} \right|$$
- ξf_c is the number of all descendant content elements of the ξ element, i.e. it is
$$| \; content\_elem(\xi) \; |$$
- ξf_k is the number of descendant content elements of the ξ element containing k, i.e. it is
$$| \; \{\xi' \in content\_elem(\xi) | \; num\_of\_keys(k, \xi') \neq 0)\} \; |$$
- v and b are constants for tuning the weighting. Their effects on performance and the size of returned elements are discussed in [11]. In this study $v = 2$ and $b = 0.1$.

The weighting formula yields weights scaled into the interval [0,...,1].

In queries one may express phrase conditions as a sequence of keys "$k_1 \ldots k_n$". The present weighting schema does not support phrase searching as such. Instead, it supports liberal proximity searching by demanding that all the keys of the phrase appear in the same content element. This approximates phrase searching when content elements are rather short. We manipulate a phrase as a set of keys denoted by KS, i.e. $KS = \{k_1, \ldots, k_n\}$ when "$k_1 \ldots k_n$" is the `phrase_expr` at hand. The weight for a phrase (a `phrase_expr` represented as KS) related to the ξ element, is calculated as follows:

$$w(KS, \xi) = \frac{pf_\xi}{pf_\xi + v \cdot \left((1-b) + b \cdot \frac{\xi f_c}{\xi f_{KS}} \right)} \cdot \frac{log\left(\frac{N}{m_{KS}} \right)}{log(N)}$$

where

- N, v, b and ξf_c have the same interpretation as above.
- pf_ξ gives the sum of the lowest frequencies, among the keys in KS, in the content elements of the ξ element. In other words, when assuming the function $min(S)$ that yields the minimum value of the argument set S consisting of integers, pf_ξ can be defined as follows:
$$\sum_{\xi' \in content\_elem(\xi)} min(\{num\_of\_keys(k, \xi') | \; k \in KS\})$$
- ξf_{KS} is the number of the descendant content elements of the ξ element containing all the keys in KS, i.e.
$$\left| \bigcap_{k \in KS} \{\xi' \in content\_elem(\xi) \mid num\_of\_keys(k, \xi') \neq 0\} \right|$$
- m_{KS} is the number of the content elements containing all the keys in KS in the collection, i.e. it is
$$\left| \bigcup_{\langle i \rangle \in IS} \bigcap_{k \in KS} \{\xi' \in content\_elem(\langle i \rangle) \mid num\_of\_keys(k, \xi') \neq 0\} \right|$$

A query term (a `query_term`), denoted by qt below, is either a single key or a phrase with a possible +/- prefix.

The '+' prefix in queries is used to emphasize the importance of a query term. In our system the weight of the query term is increased by taking a square root of the weight (NB! $\sqrt{x} > x$, when $0 < x < 1$). Related to the ξ element, for any `plus_expr` $+qt$ the weight is calculated as follows:

$$w(+qt, \xi) = \sqrt{w(qt, \xi)}$$

The '-' prefix in queries denotes an unwanted query term. For any `minus_expr` $-qt$ the weight is decreased by changing the weight to its negation:

$$w(-qt, \xi) = -w(qt, \xi)$$

In other words unwanted query terms are manipulated in the interval [-1,0].

In relevance scoring for ranking the weights of query terms are combined by taking an average of the weights. In other words, if $q = qt_1 \ldots qt_n$ (qt_i ($i \in \{1,\ldots,n\}$) is a `query_term`) is a `query_expr` then its weight is calculated as follows:

$$w(q, \xi) = \frac{\sum_{i=1}^{n} w(qt_i, \xi)}{n}$$

This is our basic weighting schema for the element associated with an index ξ.

3. TEST SETTING

We tested our system with the XML collection of INEX, and two sets of ad hoc topics from years 2003 and 2004. Topics of 2003 were used for tuning the parameters, and the results obtained with 2004 topics will be given in the next section. The INEX document collection consists of 12107 XML marked full-text documents with 494 megabytes of data. These documents are scientific articles of the IEEE Computer Society's publications from 12 magazines and 6 transactions. An article contains on average 1532 XML elements (totally over 8 million elements, the average depth of an element is 6.9). However the length and structure of the articles vary. Also the granularities of the elements vary. [2, 4]

INEX participants produce topics every year and also assess the relevance of the elements collected to a result pool from the submissions. There are two types of topics to be evaluated: content_only (CO) and content_and_structure (CAS) topics. The former gives conditions only about the content of elements to be retrieved; the latter gives restrictions about the content and structure of the results. [9, 13] In this evaluation we use only CO topics. In 2003 the number of CO topics with relevance assessments was 32, in 2004 the number was 34. In INEX 2004, only the titles of the topics were allowed in query formulation.

Also in this study search keys and phrases were taken as given in titles. Words included in phrases were added to queries also as single keys.

Relevance of elements was assessed in two dimensions, exhaustiveness and specificity, both on 4-point scale [10, 13]:

- Not exhaustive / specific (0)
- Marginally exhaustive / specific (1)
- Fairly exhaustive / specific (2)
- Highly exhaustive / specific (3)

These dimensions are not totally independent, e.g. a not exhaustive element may not be specific at any level. The evaluation measure used in INEX 2004 was mean average precision with different quantizations for relevance dimensions [4, 10]. In the strict quantization only those elements that are highly exhaustive and highly specific are considered relevant, others non-relevant. In other quantization functions elements' degree of relevance is taken into account by crediting elements according to their level of specificity and exhaustiveness. We will present our results using three official INEX measures: the aggregate mean average precision over all seven INEX quantizations (Aggr. MAP), generalized and specificity oriented mean average precision (General. MAP and SO MAP, see [22, 13]). The first gives an overview of the performance; the second treats exhaustiveness and specificity alike; and the third emphasizes specificity, which is important in XML retrieval. We use the set of relevance assessments with duplicate assessments, referred to as Ass. II in [13].

The quantizations may be expressed as a function

$$f_{quant}(e, s): (\{1, 2, 3\} \times \{1, 2, 3\}) \cup \{\langle 0, 0\rangle\} \rightarrow [0, 1]$$

Then, the generalized quantization is defined as follows:

$$f_{gen}(e,s) = \begin{cases} 1 & ,\text{if } \langle e,s\rangle = \langle 3,3\rangle \\ 0.75 & ,\text{if } \langle e,s\rangle \in \{\langle 2,3\rangle, \langle 3,2\rangle, \langle 3,1\rangle\} \\ 0.5 & ,\text{if } \langle e,s\rangle \in \{\langle 1,3\rangle, \langle 2,2\rangle, \langle 2,1\rangle\} \\ 0.25 & ,\text{if } \langle e,s\rangle \in \{\langle 1,2\rangle, \langle 1,1\rangle\} \\ 0 & ,\text{if } \langle e,s\rangle = \langle 0,0\rangle \end{cases}$$

and specificity oriented quantization is defined as follows:

$$f_{so}(e,s) = \begin{cases} 1 & ,\text{if } \langle e,s\rangle = \langle 3,3\rangle \\ 0.9 & ,\text{if } \langle e,s\rangle = \langle 2,3\rangle \\ 0.75 & ,\text{if } \langle e,s\rangle \in \{\langle 1,3\rangle, \langle 3,2\rangle\} \\ 0.5 & ,\text{if } \langle e,s\rangle = \langle 2,2\rangle \\ 0.25 & ,\text{if } \langle e,s\rangle \in \{\langle 1,2\rangle, \langle 3,1\rangle\} \\ 0.1 & ,\text{if } \langle e,s\rangle \in \{\langle 2,1\rangle, \langle 1,1\rangle\} \\ 0 & ,\text{if } \langle e,s\rangle = \langle 0,0\rangle \end{cases}$$

[22]

4. RESULTS

We tested four different retrieval methods (see Section 2.2):

- *No contextualization, c_n (Baseline)*
- *Parent contextualization, c_p (Parent)*
- *Root contextualization, c_r (Root)*
- *Tower contextualization, c_t (Tower)*

An overview of the four methods is given in Table 1. All contextualization methods improve the performance compared to the baseline. These improvements are also statistically significant (Friedman test). The best mean average precision is obtained with the root contextualization, but the difference between the root and tower contextualization is minor. The parent contextualization yields the smallest difference to the baseline; obviously it offers a too small context. The root contextualization is significantly better than the parent contextualization; with generalized and specificity oriented precisions the root contextualization outperforms the tower contextualization significantly. As a comparison for the aggregate precision values in Table 1 we refer to the best official INEX 2004 aggregate MAP which was 0.139 [13]; INEX has not published the performance of the official submission runs with other measures.

It is worth noting, that the difference in average mean precision does not necessarily imply statistical significance, because Friedman test takes into account the number of topics the method is able to improve rather than the absolute improvement shown in averages.

Table 1. MAP scores for baseline and three contextualization methods

	Aggr. MAP	Diff. to baseline	Diff. to parent	Diff. to root
Baseline	0.106			
Parent	0.129	0.023*		
Root	0.152	0.046**	0.023**	
Tower	0.147	0.041**	0.018	-0.005
	General. MAP	Diff. to baseline	Diff. to parent	Diff. to root
Baseline	0.080			
Parent	0.106	0.026**		
Root	0.134	0.055**	0.028**	
Tower	0.126	0.047**	0.020	-0.008*
	SO MAP	Diff. to baseline	Diff. to parent	Diff. to root
Baseline	0.069			
Parent	0.100	0.031**		
Root	0.139	0.070**	0.039**	
Tower	0.134	0.065**	0.034	-0.005*

Legend: * $p < 0.05$, ** $p < 0.001$

Figures 1-2 illustrate the performance of the methods as precision-recall curves. Regardless of the measure, the average performance of the methods is similar. The parent contextualization gives a clear improvement over the baseline; the performances of the root and tower contextualization are close, and they outperform both the baseline and parent contextualization. Obviously, the root gives the best evidence of

relevance in most cases; the information of the plain tower contextualization is redundant.

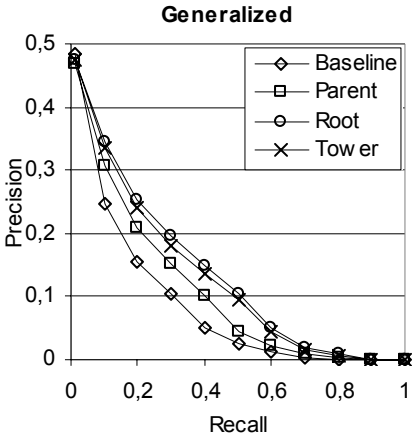

Figure 1. Recall-precision curves with generalized quantization

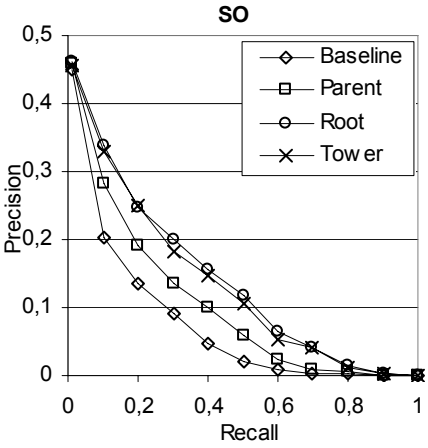

Figure 2 Recall-precision curves with specificity oriented quantization.

5. DISCUSSION AND CONCLUSIONS

In this study we have shown how manipulation – indexing and retrieval – of an XML collection is handled with structural indices. This is a general approach independent of the DTD of the collection, and thus it is applicable to heterogeneous XML collections. The structural indices are used as identifiers for elements. An inverted file stores the occurrences of keys in content elements only; the occurrences in other elements (ancestors) are deduced on the basis of the indices. Structural indices support straightforward manipulation of XML documents – not only element retrieval but also restructuring of documents, which is our forthcoming aim.

In XML retrieval the content elements or leaf nodes tend to be short, which means that the vocabulary problem typical for text retrieval is even worse: all search keys do not appear in the same element and thus there might not be enough evidence of relevance. This problem could be facilitated by seeking evidence from the surrounding context of the element to be weighted. Taking the weight of the root element into account when weighting the element at hand was put forward in [20], also [14, 18] introduce similar approaches. We propose a more general re-weighting method, contextualization, in which not only the root but any context of the element along the hierarchical path may influence the weight of the element. This is achieved through contextualization vectors and indices: the vectors include contextualization weights for each hierarchical level found in indices. The proposed method both increases the weights of elements in probably relevant contexts and decreases the weights of elements in probably not relevant contexts. The contextualization weights need not to be positive, and they may be adjusted according to assumed importance of the context. In the present study we tested only binary weighting (1 or 0).

The effectiveness of contextualization was tested with three basic methods: parent, root and tower contextualization. In this evaluation, the root contextualization proved to be the best. However, it seems that the root might also carry false evidence, and context smaller than root and larger than parent could be effective in re-weighting. This remains to be tested. In the evaluation of this study only one XML collection consisting of scientific articles and one weighting method were used, thus testing with other types of collections and matching models is needed.

6. ACKNOWLEDGMENTS

This research was supported by the Academy of Finland under grant number 52894.

7. REFERENCES

[1] Callan, J.P. Passage-level evidence in document retrieval. Passage-level evidence in document retrieval. In *Proc. of the 17th Annual International ACM SIGIR conference on Research and Development in Information Retrieval*, 1994, 302-310.

[2] Fuhr, N., Malik, S., and Lalmas, M. Overview of the initiative for the evaluation of XML retrieval (INEX) 2003. In *INEX 2003 Workshop Proc.*, 2003, 1-11. Retrieved 13.1.2005 from http://inex.is.informatik.uni-duisburg.de:2003/proceedings.pdf

[3] Gövert, N., Abolhassani, M., Fuhr, N., and Grossjohan, K. Content-oriented XML retrieval with HyRex. In *Proc. of the First Workshop of the Initiative for the Evaluation of XML Retrieval* (INEX), 2002, 26-32. Retrieved 27.1.2005 from http://qmir.dcs.qmul.ac.uk/inex/Workshop.html

[4] Gövert, N., and Kazai, G. Overview of the INitiative for the Evaluation of XML retrieval (INEX) 2002. In *Proc. of the First Workshop of the Initiative for the Evaluation of XML Retrieval* (INEX), 2002, 1-17. Retrieved 27.1.2005 from http://qmir.dcs.qmul.ac.uk/inex/Workshop.html

[5] Hawking, D., Thistlewaite, P., and Craswell, P. ANU/ACSys TREC-6 experiments. In *Proc. of TREC-6*, 1998. Retrieved 10.3.2004 from http://trec.nist.gov/pubs/trec6/papers/anu.ps

[6] ISO/IEC 14977. *International standard ISO/IEC 14977 : 1996(E). Extended BNF*. Draft, 1996.

[7] Junkkari, M. PSE: An object-oriented representation for modeling and managing part-of relationships. *Journal of Intelligent Information Systems*, forthcoming issue, 2005.

[8] Kaszkiel, M., Zobel, J., and Sacks-Davis, R. Efficient passage ranking for document databases. *ACM Transactions on Information Systems*, 17(4), 1999, 406-439.

[9] Kazai, G. Lalmas, M, and Malik, S. INEX´03 guidelines for topic development. In *INEX 2003 Workshop Proc.*, 2003, 192-199. Retrieved 21.1.2005 from http://inex.is.informatik.uni-duisburg.de: 2003/internal/downloads/INEXTopicDevGuide.pdf

[10] Kazai, G., Lalmas, M., and Piwowarski, B. INEX 2004 Relevance Assessment Guide. In *INEX 2004 Workshop Pre-Proc.*, 2004, 241-248. Retrieved 18.1.2005 from http://inex.is.informatik.uni-duisburg.de: 2004/pdf/INEX2004PreProceedings.pdf

[11] Kekäläinen, J., Junkkari, M., Arvola, P., and Aalto, T. TRIX 2004: Struggling with the overlap. In *Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*. LNCS 3493. Springer, Heidelberg, 2005, 127-139.

[12] Liu, S., Zou, Q., and Chu, W.W. Configurable indexing and ranking for XML information retrieval. In *Proc. of the 27th Annual International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, New York, 2004, 88-95.

[13] Malik, S., Lalmas, M., and Fuhr, N. Overview of INEX 2004. In *Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*. LNCS 3493. Springer, Heidelberg, 2005, 1-15.

[14] Mass, Y., and Mandelbrod, M. Component ranking and automatic query refinement for XML retrieval. In *Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*. LNCS 3493. Springer, Heidelberg, 2005, 73-84.

[15] McCallum, A., and Nigam, K. Text classification by bootstrapping with keywords, EM and shrinkage. In *Proc. of ACL 99 Workshop for Unsupervised Learning in Natural Language Processing*. 1999, 52-58.

[16] Niemi, T. A seven-tuple representation for hierarchical data structures. *Information Systems*, 8(3), 1983, 151-157.

[17] Niemi, T., and Järvelin K. The processing strategy for the NF2 relational FRC-interface. *Information & Software Technology*, 38, 1996, 11-24.

[18] Ogilvie, P., and Callan, J. Hierarchical language models for XML component retrieval. In *Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*. LNCS 3493. Springer, Heidelberg, 2005, 224-237.

[19] Robertson S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., and Gatford, M. Okapi at TREC-3. In *NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3)*. 1994. Retrieved 21.11.2004 from http://trec.nist.gov/pubs/trec3/papers/city.ps.gz

[20] Sigurbjörnsson, B., Kamps J., and de Rijke, M. An element-based approach to XML retrieval. In *INEX 2003 Workshop Proc.*, 2003, 19-26.

[21] Tatarinov, I., Viglas, S., Beyer, K.S. Shanmugasundaram, J., Shekita, E.J., and Zhang C. Storing and querying ordered XML using a relational database system. In *Proc. of the 2002 ACM SIGMOD International Conference on Management of Data*, 2002, 204-215.

[22] de Vries, A.P, Kazai, G., and Lalmas, M. Evaluation metrics 2004. In *INEX 2004 Workshop Pre-proc.*, 2004, 249-250. Retrieved 18.1.2005 from http://inex.is.informatik.uni-duisburg.de:2004/pdf/INEX2004PreProceedings.pdf

Appendix 1: Syntax of CO queries (by Extented BNF [6])

```
Query ::= query_term {' ' query_term};
query_term ::= key_expr | plus_expr | minus_expr;
plus_expr ::= '+' key_expr;
minus_expr ::= '-' key_expr;
key_expr ::= k | phrase;
phrase ::= """ k {' ' k }""";
(*k is a search key formed of allowed
    characters*).
```

Study III

Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2008) The effect of contextualization at different granularity levels in content-oriented xml retrieval. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008*, 1491-1492.

The Effect of Contextualization at Different Granularity Levels in Content-Oriented XML Retrieval

Paavo Arvola
Dept. of Information Studies
University of Tampere
Finland
paavo.arvola@uta.fi

Jaana Kekäläinen
Dept. of Information Studies
University of Tampere
Finland
jaana.kekalainen@uta.fi

Marko Junkkari
Dept. of Computer Sciences
University of Tampere
Finland
marko.junkkari@cs.uta.fi

ABSTRACT
In the hierarchical XML structure, the ancestors form the context of an XML element. The process of taking element's context into account in element scoring is called contextualization. The aim of this paper is to separate different granularity levels and test the effect of contextualization on these levels.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information search and retrieval – *Retrieval models, Search process, Selection process*

General Terms
Measurement, Performance, Experimentation

Keywords
Contextualization, Evaluation, Granularity level, XML

1. INTRODUCTION
Passage and XML retrieval allow systems to provide fine grained access to documents, and thus only the most relevant parts of a document can be retrieved. In this kind of content-oriented XML the exact paths and names of the elements are not of interest. Similar to full document retrieval, in content-oriented XML retrieval the elements are typically organized according to their relevance ranking and provided to the user.

This study investigates the effect of contextualization [1] (i.e. utilizing the elements ancestors) in content-oriented XML retrieval. In particular, this study isolates granularities in an XML collection and explores how the contextualization affects the retrieval performance on different granularity levels.

2. CONTEXTUALIZATION
Context is an ambiguous term. In this paper the context of an element refers basically to the ancestor elements. As an illustration, an accustomed text document, such as a book, a newspaper article etc. has a basic hierarchical division, for example an article-section-subsection-paragraph division. This established division gives natural contexts of different size. For example, a paragraph can be viewed in the context of an article, a section, and possible subsections, i.e. the ancestors. In a collection with documents having such a hierarchy, the full documents of a collection belong to the same granularity level with each other. In the same way the paragraphs or sections belong (more or less) to the granularity level of their own. The context defines part of the semantics of the content of an element, and thus it may carry beneficial information for retrieval.

Contextualization means mixing the evidence from an element and its context in matching. In other words, contextualization is a general re-scoring method, where the initial scores of matching elements are combined with their ancestors' scores.[1]

Accordingly, we introduce a general contextualization function. In the function we assume a basic scoring function $Score(q,e)$.

$$CScore(q,e,S,w) = Score(q,e) + w \cdot \frac{\sum_{c \in S} Score(q,c)}{|S|}$$

The argument q is the underlying query, e is the contextualized element, S is the set of the elements whose score is combined with the score of e, and w is a weight in terms of which the power of contextualization can be tuned.

Generally, contextualization has been discovered successful in element matching [1,5,6]. However, there is a lack of more detailed research on how contextualization affects different kind of elements. Hence, we present the following research questions: Does the effectiveness of contextualization vary at different granularity levels? Is the effectiveness of the different contextualization scenarios dependent on the granularity level?

3. TEST SETTINGS
As the core retrieval system, TRIX [1,3], is used to test the effect of contextualization. It was tested against the IEEE collection and a set of 29 content-only topics of the year 2005 [2]. As special cases in this study we investigate four contextualization scenarios, namely *parent, root, 2xroot* and *tower* contextualizations [1]. The scenarios for an element e are defined by parameterizing the basic scoring function with the following arguments.

For the *parent* contextualization we set w=1, and S={*parent*(e)}, for the root S={*root*(e)}, w=1, for the 2xroot S={*root*(e)}, w=2 and for the tower w=1, S=*ancestors*(e). Functions *root*(e) and, *parent*(e) yield the root and parent elements respectively, whereas *ancestors*(e) yields the set of all ancestor elements of the element e.

The various XML retrieval evaluation metrics reward not only the matching, but also the selection of appropriate granularity level [4]. Here, instead, we have extracted three granularity

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.
ACM 978-1-59593-991-3/08/10.

levels from the collection in order to measure the effect of these contextualization scenarios for elements of different granularity, and also to eliminate the effect of selecting elements of appropriate granulation in the results. The first level covers biggish elements, i.e. major sections. The second level covers 'medium size' elements, i.e. minor sections. The smallest level is the content element [3] level consisting of smallest referable units such as paragraphs, headings, list items etc.

None of the granularity levels contains structurally overlapping elements. In addition, for completeness, the selection of elements has been completed so that each level covers all text content of the XML collection. Consequently, for the major section the average text length of an element is 4243 characters, for the minor section 2420 and for the content element 121 characters.

The queries have been executed against each of the three granularity levels with TRIX. In order to evaluate these results, the INEX recall base has been filtered so that relevant elements belonging to each corresponding granularity level has been accepted. Thus, there exist totally three recall bases. For the recall bases we have applied binary relevance criteria. This all enables the usage of traditional IR evaluation metrics. Such an evaluation setting is novel in the field of the XML retrieval research.

4. RESULTS

Test results in MAP (Mean Average Precision) values are given in Table 1. The results show that contextualization is most effective when retrieval is focussed on content elements; the effects of contextualization diminish towards the major granularity level. The effectiveness of the different contextualization methods vary according to the granularity level. The tower and root contextualization methods are the most effective; tower especially for content elements, and root for minor and major elements.

Table 1. MAP values for different contextualization methods at different granularity levels. (The statistical significance of the differences between the baseline and the contextualization methods was tested withthe non-parametric Friedman test.)

Normal rel.	**MAP** Baseline	Parent	Root	2xRoot	Tower
Content	0.102	0.115	0.116	0.108	**0.127**
Minor	0.214	0.236	**0.252**	0.244	0.245
Major	0.250	0.274	**0.288**	0.285	0.280
		Change % (X-Baseline)			
		Parent-Basel.	Root-Basel.	2xRoot-Basel.	Tower-Basel.
Content		13.25*	13.84*	6.08*	**24.73***
Minor		10.58*	**17.98***	14.23*	14.65*
Major		9.40	**15.24***	13.80*	11.88*

*=p<0.05

Comparison between the contextualization methods shows some statistically significant differences although the differences in MAP are not striking. The best performing method (tower/root) outperforms parent and double root contextualization methods for content and major elements. For minor elements there are no significant differences between the methods.

5. DISCUSSION AND CONCLUSIONS

The results verify the fact, that contextualization benefits on using all levels of context on any element. Referring to the former studies [1,5,6] it is not surprising that the results for large elements improve the most with the root level contextualization. However, the difference between various contextualization methods is small. This is most likely due to the lower number of context levels for major sections.

Similarly, the experiments show that contextualization in general improves the effectiveness most on deep and small elements. This is understandable while it is known that they possess scant textual evidence. Interestingly, utilizing the near context delivers better results than the root for the small elements. It seems like the root is not an enough focussed context for small elements. Instead context levels in between are more informative in this sense. This is rather intuitive, because the content of, for example, the paragraphs and headings is more related to the section rather than the whole article

Consequently, our *CScore* function gives a better base on finer grained investigation of the near context than former approaches based on sole ancestor elements [5,6]. This is because it is based on a general set of elements, which may include a deliberate selection of following and preceding siblings as a context, instead of atomic ancestors. Testing this is a matter of an ongoing study.

6. ACKNOWLEDGEMENTS

This study was funded by the Academy of Finland under grant number 115480.

7. REFERENCES

[1] Arvola, P., Junkkari, M., and Kekäläinen, J. 2005. Generalized contextualization method for XML information retrieval. In *Proc. of CIKM '05*, 20-27.

[2] INEX 2005 homepage, INEX 2005. Available at: http://inex.is.informatik.uni-duisburg.de/2005/ [Cited 2008-5-31].

[3] Kekäläinen, J., Junkkari, M., Arvola, P., and Aalto, T. 2005. TRIX 2004: Struggling with the Overlap. In *Advances in XML Information Retrieval and Evaluation, INEX 2004*, LNCS 3493, 127-139.

[4] Lalmas, M. and Tombros, A. 2007. Evaluating XML retrieval effectiveness at INEX. *SIGIR Forum* 41, 1, 40-57.

[5] Mass, Y., and Mandelbrod, M. 2005. Component ranking and automatic query refinement for XML retrieval. In *Advances in XML Information Retrieval and Evaluation, INEX 2004*, LNCS 3493, 73-84.

[6] Sigurbjörnsson, B., Kamps, J., and de Rijke, M. 2004. An element-based approach to XML retrieval. In *INEX 2003 Workshop Proceedings*, *INEX 2003*. 19-26.

Study IV

Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2011) Contextualization Models for XML Retrieval, Information Processing & Management, Article in Press doi:10.1016/j.ipm.2011.02.006, Elsevier, 15 pages.

Information Processing and Management xxx (2011) xxx–xxx

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Contextualization models for XML retrieval

Paavo Arvola *, Jaana Kekäläinen, Marko Junkkari

University of Tampere, School of Information Sciences, FIN-33014 University of Tampere, Finland

ARTICLE INFO

Article history:
Received 5 September 2008
Received in revised form 17 February 2011
Accepted 20 February 2011
Available online xxxx

Keywords:
Contextualization
Evaluation
Granularity level
Granulation
Semi-structured data
Structured documents
Content element
XML

ABSTRACT

In a hierarchical XML structure, surrounding elements form the context of an XML element. In document-oriented XML, the context is a part of the semantics of the element and augments its textual information. The process of taking the context of the element into account in element scoring is called contextualization. This study extends the concept of contextualization and presents a classification of contextualization models. In an XML collection, elements are of different granularity, i.e. lower level elements are shorter and carry less textual information. Thus, it seems credible that contextualization interacts differently with diverse elements. Even if it is known that contextualization leads to improved effectiveness in element retrieval, the improvement on different granularity levels has not been investigated. This study explores the effect of contextualization on these levels. Further, a parameterized framework for testing contextualization is presented.

The empirical part of the study is carried out in a traditional laboratory setting, where an XML collection is granulated. This is necessary in order to measure performance separately at different hierarchy levels. The results confirm the effectiveness of contextualization, and show how the elements of different granularities benefit from contextualization.

1. Introduction

In traditional information retrieval (IR) a retrievable unit is a document. With mark-up languages, XML at the head, the structure of a document can be represented. Accordingly, XML information retrieval allows IR systems to provide focused access to documents, and only the relevant parts (elements) of documents are retrieved. A successful XML retrieval system is capable of delivering focused elements, optimal in length and covering exhaustively the topic.

Text matching methods in IR rely on the textual content of the retrievable unit. Compared with full document IR, one problem in XML IR matching is that, especially in short elements, the textual evidence for matching is scant and the meaning of an element arises partly from its context. Hence, matching, based solely on the textual content of an element, does not seem to deliver the best possible results in XML IR. It is necessary to apply supplementary methods to improve the retrieval of focused elements in particular. For instance, in order to remedy the poor matching, auxiliary evidence from the surroundings (i.e. context) of the element can be collected. We call this method contextualization following Kekäläinen and others (2009).

Contextualization relies on the explicit structure of documents. In a document two main structural dimensions can be distinguished: First, a document has a hierarchy. For example a newspaper article has a basic hierarchical division of sections subsections and paragraphs. This gives natural contexts of different sizes. For example, a paragraph can be viewed in the context of an article, a section, and a possible subsection. Second, the parts of a document follow a sequential order, often referred to as the document order, where text passages follow each other consecutively. The nearby passages are supposed to

* Corresponding author.
E-mail addresses: paavo.arvola@uta.fi (P. Arvola), jaana.kekalainen@uta.fi (J. Kekäläinen), marko.junkkari@cs.uta.fi (M. Junkkari).

0306-4573/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2011.02.006

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

form the most definitive context but the further passages in the document should be taken into account in contextualization as well.

In the present study, we develop contextualization models and present a classification for them. The main focus is to explore the effect of different contextualization models on different hierarchical levels. We are interested in improving the challenging retrieval of short and focused elements in particular, and we hypothesize that the retrieval of such elements would benefit from contextualization more than the retrieval of broader elements. We propose a general function for contextualization and show the robustness of the function by testing contextualization models in two XML test beds.

In mainstream XML IR evaluation (Lalmas & Tombros, 2007) heterogeneous result lists are produced and evaluated. In this type of evaluation the task of a retrieval system includes figuring out the proper granularity level (e.g. the paragraph or section level) in addition to good element ranking. In this study, we are interested in ranking elements of specific granularity levels only, so the heterogeneous result list evaluation setting is too complex for our purposes. Therefore, we have developed a specific evaluation setting, which is based on granulation of an XML collection. In granulation, the level of an XML hierarchy is specified in advance and the retrieval is focused on the set of elements belonging to that level. More specifically, the effect of contextualization is investigated against different granularity levels. With respect to the soundness of evaluation, it is desirable to obtain full recall at all granularity levels. Thus, the levels are specified so that each of the levels covers the whole textual content of the collection (full coverage). In addition, within each level there are no overlapping elements (no overlap). The result of granulation is a flat list of elements enabling a laboratory setting and the usage of traditional IR evaluation metrics, e.g. precision/recall.

Shortly, the main contributions of this study include

- the classification of contextualization models,
- their applications and a general contextualization function (Section 2),
- testing the effect of contextualization models on three granularity levels (Section 4) with a,
- tailored test setting (Section 3).

Section 5 concludes the article.

2. Contextualization

Contextualization is a method exploiting features in the context of an element (Arvola, Junkkari, & Kekäläinen, 2005; Kekäläinen et al., 2009). It means mixing the evidence from an element and its context in matching. The context of an element consists of *contextualizing* elements, which have a relationship and a distance to the *contextualized* element. The relationship refers to the place of the contextualizing element in the XML hierarchy with regard to the contextualized element. The distance refers to a structural remoteness between elements. These features affect the weight each of the contextualizing elements has in contextualization. Contextualization is a re-scoring method, where the initial score of an element is combined with the weighted scores of the contextualizing elements.

2.1. Classification of contextualization models

The classification we propose is based on the relationships among elements in an XML hierarchy. In Fig. 1 the hierarchical or tree structure of an XML document is depicted. Element e1 is the root whereas the elements e2, e4, e7, e8 and e9 are the leaves. The relationships among the elements are given assuming element e5 as a starting point. Elements e3 and e6 are the

	Element ids
Roles of elements	
root	e1
leaf	e2, e4, e7, e8, e9
Relationships to the element e5	
parent	e3
child	e6
descendant	e6, e7
sibling	e4, e8
ancestor	e1, e3
horizontal relatives	e2, e4, e8, e9
context	e1, e2, e3, e4, e8, e9

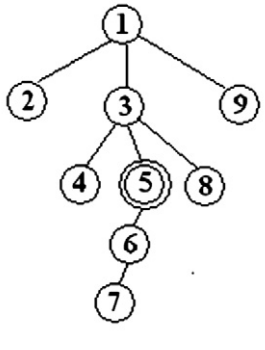

Fig. 1. A sample XML document hierarchy and the relationships of the elements in the hierarchy.

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

parent and child of element e5, respectively. Elements e1 and e3 are the ancestors of element e5 whereas elements e6 and e7 are its descendants. A descendant-ancestor pair involves a *vertical distance* and it is the number of parent–child steps (the length of the path) between them. For example the vertical distance between elements e5 and e7 is 2. In vertical relationships, the ancestors form the contexts of an element, i.e. elements e1 and e3 form the vertical contexts of element e5.

Horizontal relationships are more complex than vertical ones. Siblings form their simplest case. They are elements at the same hierarchy level (measured from the root). For example, elements e4 and e8 are the (following and preceding) siblings of element e5. Siblings of the ancestors of an element are also horizontal relatives (possible contextualizing elements) of an element, as well as the descendants of these siblings. In our example, the horizontal relatives of element e5 are e2, e4, e8 and e9.

Unlike vertical distance, a *horizontal distance* between two elements is equivocal, because it must be defined via indirect relationships. For example in Fig. 1 the horizontal distance between elements e2 and e9 can be two or five measured by sibling steps or via leaves respectively. Instead, the horizontal distance is unequivocal in a set of elements where no two elements are in an ancestor–descendant relationship, and there is one element from each path from the root to the leaves. In Fig. 1, {e1}, {e2, e3, e9} and {e2, e4, e5, e8, e9}, for example, are such sets. Now the horizontal distance can be defined based on the sequential order of the elements. For example the sequential order of the set {e2, e4, e5, e8, e9} is ⟨e2, e4, e5, e8, e9⟩ and the distance of elements e4 and e9 is 3. This kind of element set is called a *granularity level* of an XML document. Granulation is further discussed in Section 3.2.

From now on we use the following functional notations for relationships among elements:

- *root*(x) yields the root of the element x,
- *parent*(x) yields the parent of x,
- *descendants*(x) yields all descendants of x,
- *ancestors*(x) yields all ancestors of x,
- *depth*(x) yields the vertical distance from the root to x,
- *h_distance*(x, y, S) yields the horizontal distance between the elements x and y in the set S. $x, y \in S$, and in S there is no ancestor–descendant relationship and there is an element from each path from the root to the leaves.

As introduced above, the context of elements can be viewed in vertical and horizontal directions. In general, we distinguish three types of contextualization.

Vertical (hierarchical) contextualization is a well-known and common contextualization model in XML retrieval (Arvola et al., 2005; Kekäläinen et al., 2009; Mass & Mandelbrod, 2005; Sigurbjörnsson, Kamps, & de Rijke, 2004). In vertical contextualization, a rough classification of different context levels can be formed according to the vertical distance of ancestor elements. The nearest context of the element is the parent element. Likewise, the furthermost context is the whole document, i.e. the root element. These documents are considered forming the root levels. Thus, the root element possesses no explicit context.

Horizontal contextualization. Apart from the vertical order, the elements have a document order. In the document order the elements form a chain from the first element to the last one, where each element is preceding or following another. The document order does not allow preceding and following elements to overlap (e.g. Clark & DeRose, 1999), hence any two elements have some horizontal distance between them. These preceding and following elements form the context. Thus, in horizontal contextualization the contextualizing elements are independent of the contextualized element, whereas in vertical contextualization the contextualized element contributes to the weight of its contextualizing elements. In addition, horizontal contextualization can be used also in the event that there is no explicit hierarchy present, thus it is applicable to passage retrieval. To our knowledge, horizontal contextualization has not been studied earlier.

Ad hoc *contextualization* contains a number of other contextualization methods, where the contextualizing elements are selected from a known structure or even from another document (e.g. via links). A typical usage of this kind of contextualization is query specific. For example queries containing source element constraints can actually be considered as a form of contextualization. This is illustrated by a NEXI (Trotman & Sigurbjrnsson, 2004) expression

```
//article[about(.//abstract, contextualization)]//paragraph[about(., systems)]
```

which explicitly requires a specific context for any paragraph about systems with the abstract of the corresponding article. The interpretation of Content-and-Structure queries (e.g. Kamps, Marx, de Rijke, & Sigurbjörnsson, 2005) is a representative example of ad hoc contextualixation.

These three contextualization models and their combinations can be used to improve performance in element retrieval. The effectiveness of vertical contextualization has been proven for heterogeneous element lists. Sigurbjörnsson and others (2004) showed a significant improvement in results by taking the root level into account in element scoring. Mass and Mandelbrod (2005) scaled the final score of an element in XML IR. The scaling is based on the document pivot factor, the score of the root element, and the score of the element at hand. The mentioned studies hint that taking the root into account with a double score delivers the best results. Apart from root contextualization, Arvola and others (2005) generalized the context to include other ancestor levels as well. They suggested contextualization functions based on the usage of hierarchical context levels, namely the root, parent and all ancestors (root, parent and tower contextualization respectively).

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Ogilvie and Callan (2005) applied contextualization in hierarchical language modeling. There, contextualization is applied for the score of each keyword of the query rather than the score of the element. As they do not include descendants in the direct estimation of the model of the element, they utilize children to smooth up parents (smooth up tree). Since we have not regarded descendants as a context but rather included them in elements' primary scoring, there is no direct counterpart to the smooth up tree in our categorization. In the contextualization phase, parents serve as contextualizing elements for children (smooth down tree or shrinkage). The process goes through the XML hierarchy, thus the contextualization corresponds to the tower contextualization proposed by Arvola and others (2005), and is a type of vertical contextualization. In the hierarchical language modeling approach the strength of the contextualization is adjusted by parameters, which can be chosen in different ways, e.g. parameters may depend on the length or type of the element in the smooth up tree (Ogilvie & Callan, 2005).

2.2. General re-scoring function

Arvola and others (2005) generalized vertical contextualization, whereas now we give a more general re-scoring function *RS*, which allows any (contextualizing) scores to be added to the initial score. Formally the function is defined as follows:

$$RS(x, f, D, g) = sx + f \cdot \frac{\sum_{y \in D} sy \cdot g(x, y)}{\sum_{y \in D} g(x, y)} \tag{1}$$

where

- s_x is the initial score for the contextualized element x.
- f is a constant for the context weighting, i.e. it determines the weight of the context as a whole.
- D is a set of contextualizing elements of x, i.e. $D \subseteq descendants(root(x)) - (descendants(x) \cup \{x\})$.
- g is a function that maps x with its contextualizing elements and yields the weights associated with the related contextualization.

The g function determines the importance of a contextualizing element. In vertical and horizontal contextualization, this can be based on vertical and horizontal distances between contextualized and contextualizing elements, whereas in ad hoc contextualization it can be based on explicit conditions. In ad hoc contextualization the importance of elements would be given explicitly by a g function having e.g. marking-up conditions. In this study, we concentrate on vertical and horizontal contextualization.

2.3. Vertical contextualization

The role and relation of a contextualizing element are operationalized by giving the element a contextualizing weight. For this purpose, Arvola and others (2005) defined a contextualization vector for vertical contextualization. For the general contextualization function *RS* we must reformulate their idea so that the contextualization vector is represented explicitly by a g function.

In vertical contextualization we concentrate on three levels, namely, *parent*, *root* and *ancestor* other than the parent or root. For representing the level of vertical contextualization, we define a 3-tuple $par = \langle p, a, r \rangle$, where p stands for the weight of the parent element, r stands for the weight of the root and a stands for the weight of the other ancestors. Accordingly, it is agreed that when there is a single ancestor, only r is taken into account and when there are two, both r and p are taken into account. When there are multiple levels of ancestors, the ancestors in between the root and parent are treated as a single pseudo element by taking the average of all of these element scores multiplied with a. This way the cumulating effect of numerous context levels is excluded, and the special role of the root and parent elements as contexts is acknowledged. In terms of the *par* tuple, the g function can be defined as follows:

$$g(x, y) = \begin{cases} 0, & \text{if } y \notin ancestors(x) \\ r, & \text{if } y = root(x) \\ p, & \text{if } y = parent(x) \text{ and } depth(x) > 1 \\ \frac{a}{depth(x)-2}, & \text{otherwise} \end{cases} \tag{2}$$

For example, let us initialize the tuple *par* to be $\langle 2, 5, 3 \rangle$ and consider element e7 in Fig. 1. Now $depth(e7) = 4$, $D = ancestors(e7) = \{e1, e3, e5, e6\}$, and g is

$$g(e7, e1) = 2$$
$$g(e7, e3) = 2.5$$
$$g(e7, e5) = 2.5$$
$$g(e7, e6) = 3$$

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Further, let $s_{e7} = 0.4$, $s_{e6} = 0.4$, $s_{e5} = 0.4$, $s_{e3} = 0.3$, and $s_{e1} = 0.2$, and $f = 1$. Now, the contextualized weight for e7 is calculated as follows:

$$RS(e7, 1, \{e1, e3, e5, e6\}, g) = 0.4 + 1 \cdot \frac{2 \cdot 0.4 + 2.5 \cdot 0.4 + 2.5 \cdot 0.3 + 3 \cdot 0.2}{2 + 2.5 + 2.5 + 3} \approx 0.7$$

In Section 5, the values of *par* and *f* are optimized and tested for different sizes of elements, i.e. granularity levels.

2.4. Horizontal contextualization

The nature of horizontal contextualization differs from the vertical one. First, vertical contextualization is single-directional but horizontal contextualization is bi-directional based on the preceding and following elements of the contextualized element. Second, there are typically more contextualizing elements in horizontal contextualization. Third, in vertical contextualization the contextualizing elements overlap, but in horizontal contextualization a meaningful requirement is that they do not overlap with each other or with the contextualized element.

As in vertical approach, in horizontal contextualization the weight of a contextualizing element is a function of distance. We assume that the weight ought to be the lower the further away the contextualizing element is from the contextualized element. Thus the neighbors are expected to form the most important context. In this study, we set the weight array to follow a zero centered parabola and we define the weight of a contextualizing element as follows:

$$g(x, y) = max(-\alpha d^2 + \gamma, 0), \quad \text{so that } d = h_{distance}(x, y, S) \tag{3}$$

where α and γ are the parabola parameters to be tuned. The function $h\_distance(x, y, S)$ is defined in Section 2.1.

Fig. 2 illustrates the effect of distance on weight while tuning the parameters (α and γ).

In order to illustrate the horizontal contextualization let us consider element e2 within the element set {e2, e4, e5, e8, e9} related to Fig. 1. Now D = {e4, e5, e8, e9} and the distances between contextualised and contextualising element are:

$$h\_distance(e2, e4, \{e2, e4, e5, e8, e9\}) = 1$$
$$h\_distance(e2, e5, \{e2, e4, e5, e8, e9\}) = 2$$
$$h\_distance(e2, e8, \{e2, e4, e5, e8, e9\}) = 3$$
$$h\_distance(e2, e9, \{e2, e4, e5, e8, e9\}) = 4$$

The scores of elements are assumed to be $s_{e2} = 0.2$, $s_{e4} = 0.9$, $s_{e5} = 0$, $s_{e8} = 0$, $s_{e9} = 0.1$.

The g function is defined by the parabola where $\alpha = 0.04$ and $\gamma = 1$ as follows:

$$g(e2, e4) = 0.96$$
$$g(e2, e5) = 0.84$$
$$g(e2, e8) = 0.64$$
$$g(e2, e9) = 0.36$$

The total score *RS* for element e is defined as follows:

$$RS(e4, 1, \{e2, e5, e8, e9\}, g) = 0.2 + 1 \cdot \frac{0.9 \cdot 0.96 + 0 + 0 + 0.1 \cdot 0.36}{0.96 + 0.84 + 0.64 + 0.36} \approx 0.521$$

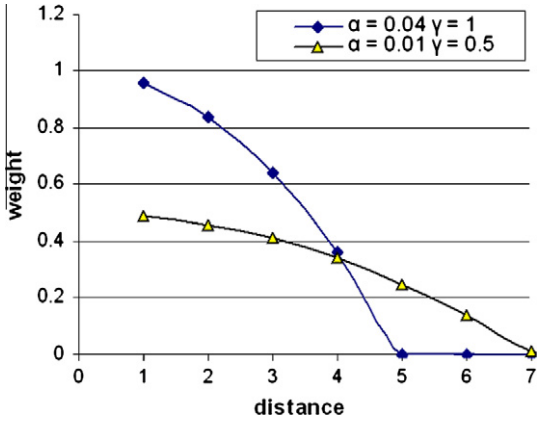

Fig. 2. The effect of distance on weight with distinct values of α and γ.

As a comparative example, the parabola with the parameter values $\alpha = 0.01$ and $\gamma = 0.5$ give the following scores for the function g:

$$g(e2, e4) = 0.49$$
$$g(e2, e5) = 0.46$$
$$g(e2, e8) = 0.41$$
$$g(e2, e9) = 0.34$$

This gives slightly less relative weight for the nearest context in the RS function and the result with the same element weight, context and contextualization magnitude (1) is approximately 0.479.

3. Test collection, relevance assessments and granulation

In this section, we first present the test collection utilized in the present study and discuss the notions of relevance in XML IR. Then we explain how the collection is granulated to test the effects of contextualization on the elements of different hierarchy levels.

3.1. INEX test collection

INEX (the INitiative for the Evaluation for XML retrieval) provides a test bed for XML IR evaluation (Malik, Lalmas, & Fuhr, 2005). This includes a document collection, topics, relevance assessments and metrics. The initiative has been running since 2002 and several changes have been made during the years, including the collection, as well as the metrics and the relevance assessment process.

In XML documents, elements overlap with each other. In XML IR, this is a challenge in the result presentation, because when retrieving, say, two overlapping elements, part of the content is retrieved twice. A straightforward solution to prevent this kind of redundancy is to exclude the ancestors and descendants of a retrieved element from the results. This kind of result list is still heterogeneous, while it may contain elements of any granularity ranging from a small text element to the root.

This heterogeneity in result lists challenges the notion of topical relevance as the criterion for retrieval quality. Namely, if an element containing any amount of relevant text is relevant, an element is necessarily at least as relevant as any of its descendants. Such interpretation of relevance leads paradoxically to poor performance for systems returning short and focused elements, which, for their part, motivate XML IR. Therefore, in the early INEX evaluation methodology two measures, exhaustivity and specificity, were introduced. "Exhaustivity is defined as a measure of how exhaustively a document component discusses the topic of request, while specificity is defined as a measure of how focused the component is on the topic of request (i.e. discusses no other, irrelevant topics)" (Kazai, Lalmas, & de Vries, 2004, p. 73).

In the early INEX ad hoc tracks (2002–2004) assessments were done element wise, so that each element in the assessment pool was judged for exhaustivity and specificity. Both of the dimensions have a four point scale with 0 meaning not exhaustive/specific and 3 meaning very exhaustive/specific. (Kazai et al., 2004; Malik et al., 2005). This kind of assessment process gives an explicit relevance value for each element, but is considered laborious from the assessors' perspective (Ogilvie & Lalmas, 2006). Hence, in 2005 to decrease assessment effort, a highlighting procedure was introduced: "In the first pass, assessors highlight text fragments that contain only relevant information. In the second pass, assessors judge the exhaustivity level of any elements that have highlighted parts. As a result of this process, any elements that have been fully highlighted will be automatically labelled as fully specific. The main advantage of this highlighting approach is that assessors will now only have to judge the exhaustivity level of the elements that have highlighted parts (in the second phase). The specificity of any other (partially highlighted) elements will be calculated automatically as some function of the contained relevant and irrelevant content (e.g. in the simplest case as the ratio of relevant content to all content, measured in number of words or characters)." (Lalmas & Piwowarski, 2005, p. 391).

The aim of XML IR is to retrieve not only document components about the subject of the topic but also those at the optimal hierarchy level (Malik et al., 2005; Pehcevski & Piwowarski, 2009). Hence, the retrieval quality is measured as a trade-off of exhaustivity and specificity. This has lead to a number of rather complex evaluation metrics (see Lalmas & Tombros, 2007; Pehcevski & Piwowarski, 2009), where the optimal hierarchy level is often obtained post hoc (e.g. Kamps, Koolen, & Lalmas, 2008). In other words, it is discovered that it is reasonable to return e.g. full documents only.

Another alternative is to use flat, non-overlapping result lists, where the set of retrievable elements is pre-defined, and the elements belonging to the set are considered to be (more or less) at the same hierarchy level. This approach suits the goals of the present study better, because we are not interested in selecting the optimal level. Instead, we want to measure performance at pre-set hierarchy levels. Therefore, for measuring the effect of contextualization at different hierarchy levels we aim to distinguish three levels of different nature in the collection(s): smallish elements, moderate elements and large elements. For each of these levels, we tailor a separate set of relevance assessments (recall base), based on the relevance assessments in the collection of the INEX. Further, this enables applying TREC style binary relevance – an element is relevant if it contains relevant text; otherwise the element is not relevant – and standard evaluation measures. In case of small elements, binary relevance is applicable because the variation in specificity is

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

limited; in case of larger elements, binary relevance does not distinguish the obviously greater variation. To overcome this problem we also apply graded relevance based on the combination of exhaustivity and specificity figures in INEX collection.

3.2. Granulating the IEEE collection

In order to measure the effectiveness of contextualization on a specific granularity level, the set of elements belonging to the level ought to be carefully defined. In our experiments, we use the INEX 2004 and 2005 collections and related topics with relevance assessments. The INEX 2004 collection consists of 12 107 XML marked full-text documents, whereas the INEX 2005 document collection includes these plus additional 4712 documents, totalling in 764 megabytes of data. These are scientific documents of the IEEE Computer Society's publications from 12 magazines and 6 transactions. In the INEX (IEEE) collection the granularity levels are relatively easy to distinguish by providing reasonably clear and standard division of article-section-subsection-paragraph levels, similar to many other XML standards for structured text.

The requirements for each of the three granularity levels are that the retrievable units are structurally non-overlapping and cover all of the text content in the collection. The former requirement enables the usage of conventional evaluation metrics. This is because

(1) The elements are assumed to be independent of each other.
(2) There is no granularity level selection in the retrieval (it is forced).

Since the set of retrievable elements is known in advance, these elements can be treated in evaluation as if they were documents. The full coverage of the collection's content naturally means that every bit of (relevant) text can be retrieved.

The definition of a granularity level is highly contractual because of the semi-structured nature of XML and complexity in element naming, In order to find appropriate granularity levels, we have to analyze the schema and the common structure of the collection. However, there is a possibility to achieve an unambiguous granularity level of an XML document starting from the leaves. Namely, selecting those elements that are parents of text elements and whose ancestors do not contain a text element as an immediate component. These kinds of elements are called content elements (Kekäläinen, Junkkari, Arvola, & Aalto, 2005).

The mark-up in the IEEE collection is high-quality, and thus the content element granulation corresponds well to the paragraph level, covering other small logical text units such as headings and list items. For the section and subsection granularity levels we use more contractual selections. These can be specified by explicitly defining the set of retrievable units using XPath expressions. The objective of the granulation process is illustrated in Fig. 3.

Next we define the pre-set granularity levels for the collection.

- *Paragraph/content* element granulation: set of lowermost non-overlapping elements having 100% text coverage of the collection.
- *Minor section (i.e. subsection) granulation*: set of lowermost (non-overlapping) sections having 100% text coverage of the collection.
- *Major section granulation*: set of uppermost (non-overlapping) sections having 100% text coverage of the collection.

In practise minor and major section granulations require explicit definitions following the schema of the underlying collection. That is because of their insufficient coverage of the collection's text and a variety of element name aliases. The exact XPath definition for the minor and major sections for the IEEE collection is given in Appendix A.

It is practically unavoidable to distinguish the granularity levels so that they cover the whole collection's text, contain no overlap, and still do not share the same elements. For instance, if a section does not contain any sub- or super-sections, its text has still to be taken into account in subsection granulation. Therefore we label the section granulations as minor and major sections, which both contain sections without sub- or super-sections, if such exist. However, generally the profiles of granulation vary by average element length. For the major section, the average text length of an element is 4243 characters, for the minor section 2420 and for the content element 121 characters (in the 2005 collection).

The experiments in the next section are based on INEX data with 29 Content-only (CO) topics from 2005 and 34 CO topics from 2004. The collection sizes and recall base characteristics are shown in Table 1. Corresponding recall bases are built in the following fashion: First, the relevance assessments are made binary, so that any element containing relevant text is considered relevant. Second, as the INEX CO recall base contains elements of practically any kind, only the elements that belong to the selected granularity level are selected. For example, the recall base for the content element retrieval contains content elements only. Thus, there are three different recall bases, one for each granularity level.

Besides binary relevance, we applied graded relevance in evaluation. The contemporary INEX exhaustivity interpretation is liberal considering every element containing relevant text exhaustive. However, in INEX 2005 the exhaustivity dimension is assessed with a four graded scale having "too small" (yet relevant) elements as a special case of exhaustivity. In the following the "too small" elements are given a score 1, exhaustive elements a score 2 and highly exhaustive elements a score 3. For the 2004 collection, we adopt the exhaustivity and specificity scores as they are assessed, i.e. both having a scale 0–3. The specificity dimension in the 2004 collection is assessed, but in 2005 collection it calculated as the proportion of relevant text

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

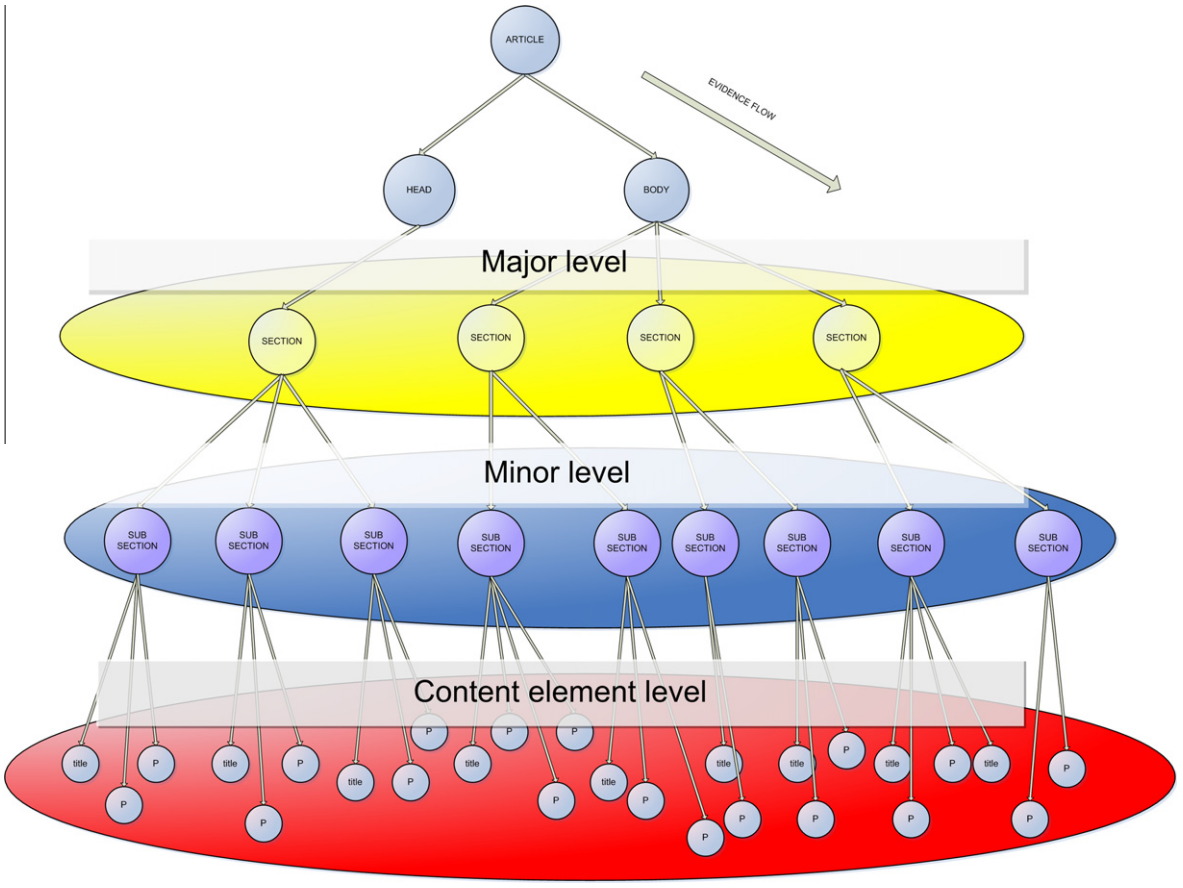

Fig. 3. Three granularity levels in a document.

Table 1
Collection and recall base characteristics on each granularity level.

Year	Collection size (elems)		Avg. relevant elements/topic		Avg. relevant documents/topic	
	2004	2005	2004	2005	2004	2005
Content element	32,15,852	45,28,958	283	691	37	50
Minor section	146,529	205,553	99	126	37	50
Major section	91,517	128,303	84	99	37	50

(i.e. relevance density (Arvola, Kekäläinen, & Junkkari, 2010)). Then we combine exhaustivity and specificity values into one graded relevance figure by multiplying exhaustivity by specificity.

4. Experiments

In this section we parameterize the general re-scoring function (RS) for the vertical and horizontal contextualization models. For vertical contextualization we test the optimized *f* and *par* parameters (see Section 2.2) on the three selected granularity levels specified in the previous section. For horizontal contextualization, in turn, we set the *a* and *c* parameters for the weights (Section 2.3). In horizontal contextualization, we make a deliberate choice testing the content element level only, because the meaningfulness of horizontal distance depends on the number of elements in a granularity level. The optimal values for the parameters (i.e. training) are obtained with the 2005 and 2004 data and tested with the other data. That is, the best trained parameters with 2004 topics and collection are tested with the 2005 topics and collection and vice versa. We use binary relevance criterion and report the results with mean average precision (MAP). In addition, we evaluate with graded relevance based on exhaustivity and specificity.

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

4.1. Retrieval system

The core retrieval system, TRIX (Tampere Retrieval and Indexing for XML; Arvola, Junkkari, & Kekäläinen, 2006; Arvola et al., 2005; Kekäläinen et al., 2005), is used to test the effect of contextualization. From the perspective of the present study, the retrieval system is secondary, because the findings can be falsified or verified with any equivalent partial match XML IR system. However, the good performance of the system entitles its usage to set the baseline high enough. For instance, TRIX was the best performing system within the INEX content-and-structure task with strict interpretation for the target element (SSCAS, VSCAS) in 2005 (Arvola et al., 2006) and with CO-task 2004 when measured with normalized extended cumulated gain (Kazai, Lalmas, & de Vries, 2005).

A basic concept in the system is the content element, which is the uppermost element containing text (see Section 3). As such the system is meaningful in XML-collections where the textual content is principally in the leaves, i.e. paragraphs, headings etc., like in the IEEE collection. Otherwise the definition of the content element should be somewhat modified. In the weighting of keys, which is basically a tf*idf modification, document length normalization is replaced by element 'length' normalization based on the number of descendant content elements and all descendant elements. In many document retrieval applications, the idf part is calculated on the basis of the number of documents including the key in the collection, and the size of the whole collection. However, some XML collections are not organized according to documents, thus the number of elements is used instead. The weight for element e in relation to key t is calculated as follows:

$$tw(t,e) = \frac{tf_e}{tf_e + 2 \cdot (0.9 + 0.1 \cdot \frac{\text{c\_elems}(e,*)}{\text{c\_elems}(e,t)})} \cdot \frac{\log(\frac{N}{n})}{\log(N)} \quad (4)$$

in which:

- $tw(t, e)$ is the weight for key t in element e,
- tf_e is the number of times search key t occurs in e element,
- N is the total number of content elements in the collection,
- n is the number of content elements containing t in the collection,
- $c\_elems(e, t)$ yields the number of content elements in the descendants (or self) of e containing t,
- $c\_elems(e,*)$ yields the total number of content elements in the descendants (or self) of e.

The constants $(2, 0.9, 0.1)$ have been discovered good in various settings (Arvola et al., 2005, 2006; Kekäläinen et al., 2005). Eventually, the score of an element is the sum of term weights:

$$Score(q,e) = \sum_{t \in q} tw(t,e) \quad (5)$$

4.2. Vertical contextualization

The training of the system was done by reasonable extensive testing for all the data (2004 & 2005) with a number of different contextualization parameters. For testing the parameters were applied to the data of the other year. As defined in this study, contextualization has two general dimensions: the overall magnitude of contextualization and the proportions the individual contextualizing elements have. In the vertical contextualization the proportions are related to the hierarchical positions of the elements. We have simplified the adjustment of the parameters by using one figure for each of the two dimensions; that is, one for the overall magnitude (the f –parameter as such) and the other for the roles of the hierarchy levels (the *par* parameter).

The magnitude is adopted by using the f parameter directly. However, using one parameter for the roles of the hierarchy levels means truncating the three parameters $\langle p, a, r \rangle$ into one slider. In other words, instead of trying a set of different values for the p, a, and r parameters, for simplicity a single parameter x (ranging from -1 to 1) controls their different values. The value of x determines whether the important context is towards the root ($x = -1$) or close to the contextualized element ($x = 1$). This is a tolerable simplification, because we might state that the contextualization bias is either more on the parent side or on the root side or then balanced between these two. Tables 6–10 in Appendix B are interpreted so that if the hierarchy variable is negative, the contextualization balance is on the parent side and if positive the balance is on the root side. The maximum and minimum values mean that only root or parent is represented in contextualization respectively.

According to the collection schemas, the major level has typically only two context levels (*bdy* and *article*), whereas the content element level consists of elements of varying depth. The minor section level is in between, but closer to the major section level. Therefore, we adopt the sliders gradually, so that for the major section we apply root and use only the contextualization magnitude (i.e. f). For the minor and content element levels, we adopt the hierarchy dimension so that for the minor sections p and r are applied, and in the content level p, a, and r are taken into account. The hierarchy values in Tables 7–10 in Appendix B are interpreted as p, a, and r values as follows:

$$p = max(0, min(x+1, 0) - max(x-1, -1))$$
$$a = max(0, min(x+1, 0.5) - max(x-1, -0.5))$$
$$r = max(0, min(x+1, 1) - max(x-1, 0))$$

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Table 2
MAP for vertical contextualization at different granularity levels, training and test results.

MAP	Content element		Minor section		Major section	
	2004 $N = 34$	2005 $N = 29$	2004 $N = 34$	2005 $N = 29$	2004 $N = 34$	2005 $N = 29$
Baseline	12.1	10.0	18.5	21.4	22.6	25.0
Best trained	19.7	12.7	26.9	24.7	29.6	28.2
Best tested	19.2	11.3	26.4	24.3	29.4	27.9
Improvement% wrt baseline						
Best tested	59.2***	12.0	42.4**	13.8	29.8**	11.8*

t-Test: contextualization versus baseline.
* $p < 0.05$.
** $p < 0.01$.
*** $p < 0.001$.

In the formulas, x denotes the hierarchy value in the table. For instance, if $x = -0.5$, then $p = 1$, $a = 1$ and $r = 0.5$. In other words parent has double contextualization weight in comparison to the root.

Table 2 presents the effect of vertical contextualization. The roles of the collections 2004 and 2005 are swapped mutually so that the best trained values of one collection are tested with the other. The best trained values at the content element level were $f = 1.75$, $par = \langle 0, 0.25, 0.75 \rangle$ for the 2004 collection, and $f = 1.25$, $par = \langle 0.5, 1, 1 \rangle$ for the 2005 collection (see Tables 9 and 10 in Appendix B); at the minor level $f = 1.75$, $par = \langle 0, 0, 1 \rangle$ for the 2004 collection, and $f = 1.25$, $par = \langle 0.25, 0, 1 \rangle$ for the 2005 collection (see Tables 7 and 8); at the major level $f = 2$ for the 2004 collection, and $f = 1.25$ for the 2005 collection (see Tables 6). Generally, vertical contextualization improves the results significantly (statistical significance tested with *t*-test; the best tested vertical contextualization versus baseline) in the 2004 data, especially at the small granularity levels. However, when tested with the 2005 data, the improvement was notable, but not significant, except with the major level. A low number of topics (29) may have an influence on the result of the statistical test. As Tables 9 and 10 show, many different parameter combinations yield quite similar results, having the bias heavily on the root side.

We tested also the robustness of training by comparing the results of the best trained and best tested runs *in the same collection*. There are no statistically significant differences between the results (*t*-test $p < 0.05$) at any granularity level, thus the method appears robust.

4.3. Horizontal contextualization

Horizontal contextualization was tested on the content element level only. The horizontal contextualization model presented in this study is a parabola, in which the roles of individual elements are present. The contextualization is tested with tuning the α and γ parameters using contextualization magnitude 1. Table 3 shows the best tested and trained values and we report extensive results with varying α and γ values in Tables 11 and 12 in Appendix C.

In training, the greatest benefit was obtained with $\alpha = 0.00001$, $\gamma = 0.025$ using the 2004 data and with $\alpha = 0.00005$, $\gamma = 0.035$ using the 2005 data. The testing was done for both collections with the values obtained with the training data. Naturally, the baseline is the same as with the vertical contextualization. The performance of the presented horizontal contextualization improves the baseline significantly with the 2004 data (statistical significance tested with *t*-test $p < 0.001$; best contextualization method versus the baseline). The improvement is notable also with the 2005 data, but the improvement is not statistically significant.

Horizontal contextualization does not quite reach the level of vertical contextualization. We compared the best vertical contextualization method to the best horizontal contextualization method. The difference between the two methods is statistically significant only with the 2004 data ($p < 0.01$ for the average precision with *t*-test). We also tested the robustness of training by comparing the results of the best trained and best tested runs *in the same collection*. There are no statistically significant differences between the results (*t*-test $p < 0.05$), thus the method appears robust.

Table 3
MAP for horizontal contextualization at content element level, training and test results.

MAP	2004 $N = 34$	2005 $N = 29$
Baseline	12.1	10.0
Best trained	15.4	10.9
Best tested	15.3	10.7
Improvement% wrt baseline		
Best tested	26.4***	6.0

t-Test: contextualization versus baseline.
*** $p < 0.001$.

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Table 4
nDCG at different granularity levels, vertical and horizontal contextualization, 2004 and 2005 data.

	Content element			Minor section			Major section		
	@10	@100	@1500	@10	@100	@1500	@10	@100	@1500
2004 nDCG (34 topics)									
Baseline	0.303	0.241	0.327	0.280	0.296	0.435	0.348	0.361	0.499
Vertical: best tested	0.375	0.329	0.414	0.357	0.396	0.511	0.406	0.434	0.562
Improvement%	23.4**	27.7***	16.8***	36.3***	33.9***	20.3***	26.9**	17.5***	12.4***
Horizontal: best tested	0.322	0.280	0.371	–	–	–	–	–	–
Improvement%	6.2	16.2**	13.9***	–	–	–	–	–	–
2005 nDCG (29 topics)									
Baseline	0.337	0.237	0.261	0.202	0.254	0.355	0.177	0.250	0.350
Vertical: best tested	0.356	0.252	0.299	0.259	0.312	0.420	0.200	0.312	0.399
Improvement%	5.6	6.7	14.6**	28.3	22.8*	18.4**	12.8	25.0**	14.1**
Horizontal: best tested	0.340	0.248	0.277	–	–	–	–	–	–
Improvement%	0.8	5.0	6.2	–	–	–	–	–	–

t-Test: contextualization versus baseline.
* $p < 0.05$.
** $p < 0.01$.
*** $p < 0.001$.

Table 5
Number of topics (%) classified by the degree of difference to the baseline performance, vertical and horizontal contextualizations.

Year	2004 (34 topics)			2005 (29 topics)		
Difference	$\geqslant$5%	0% $\geqslant$ – < 5%	<0%	$\geqslant$5%	0% $\geqslant$ – < 5%	<0%
Vertical contextualization						
Avg. prec.						
Content	16 (47%)	12 (35%)	6 (18%)	8 (28%)	14 (48%)	7 (24%)
Minor	18 (53%)	13 (38%)	3 (9%)	11 (38%)	10 (34%)	8 (28%)
Major	12 (35%)	18 (53%)	4 (12%)	11 (38%)	8 (28%)	10 (34%)
nDCG@10						
Content	16 (47%)	8 (24%)	10 (29%)	9 (31%)	8 (28%)	12 (41%)
Minor	17 (50%)	10 (29%)	7 (21%)	15 (52%)	8 (28%)	6 (20%)
Major	16 (47%)	11 (32%)	7 (21%)	12 (41%)	9 (31%)	8 (28%)
nDCG@100						
Content	18 (53%)	9 (26%)	7 (21%)	6 (20%)	15 (52%)	8 (28%)
Minor	22 (65%)	10 (29%)	2 (6%)	13 (45%)	7 (24%)	9 (31%)
Major	19 (56%)	10 (29%)	5 (15%)	14 (48%)	8 (28%)	7 (24%)
nDCG@1500						
Content	22 (65%)	5 (15%)	7 (21%)	12 (41%)	12 (41%)	5 (18%)
Minor	22 (65%)	10 (29%)	2 (6%)	17 (58%)	5 (18%)	7 (24%)
Major	20 (59%)	9 (26%)	5 (15%)	12 (41%)	12 (41%)	5 (18%)
Horizontal contextualization (Content element only)						
Avg. prec	7 (21%)	17 (50%)	10 (29%)	2 (7%)	16 (55%)	11 (38%)
nDCG@10	11 (32%)	10 (29%)	13 (38%)	6 (20%)	9 (31%)	14 (48%)
nDCG@100	12 (35%)	13 (38%)	9 (27%)	6 (20%)	12 (41%)	11 (38%)
nDCG@1500	15 (44%)	10 (29%)	9 (27%)	6 (20%)	11 (38%)	12 (41%)

4.4. Graded relevance and topic-by-topic analysis

In Table 4 the normalized discounted cumulative gain (nDCG, Järvelin & Kekäläinen, 2002) figures are given for the baseline, and vertical and horizontal contextualizations. The nDCG results are in line compared with the MAP results: both show that vertical contextualization enhances performance with the 2004 data, even significantly. With the 2005 data the improvement is notable and statistically significant at nDCG@1500 for all granulation levels. Minor and major sections benefit significantly even at cut-off 100. Horizontal contextualization is less effective, but still notable in 2004 data; however, improvement in 2005 data is insignificant.

The average figures do not reveal the benefit of contextualization for individual topics; therefore a topic-by-topic analysis was performed. The topics are classified according to the percentage unit difference in performance compared between the best tested contextualization and baseline (best – baseline).

For vertical contextualization, the percentage of topics with difference greater than or equal to 5% (e.g. notable improvement) ranges from about 20% to 65%. In any case, over half of the topics (59–94%) benefit from vertical contextualization (see

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Table 5, columns 2 + 3 and 5 + 6). Minor section level measured with nDCG seems to benefit most of this contextualization. Horizontal contextualization is less effective than vertical contextualization: 51–73% of the topics benefit from it. Only content level was tested.

5. Discussion and conclusions

Contextualization is a re-ranking method utilizing the context of an element in scoring. In this study, contextualization is calculated as a linear combination of the weights of an element itself and its contextualizing elements. We developed the contextualization methodology by introducing a classification of three contextualization models: vertical, horizontal and ad hoc contextualization, among which horizontal contextualization is a novel contextualization model. In horizontal contextualization, the context is based on the document order instead of the hierarchy. Thus, horizontal contextualization is more versatile in a sense that it does not require a hierarchy, and can be used slightly modified in non-structured passage retrieval.

We introduced a general contextualization function as an umbrella function for all presented contextualization models. Within the vertical and horizontal models, we introduced implementing methods, through which we tested the effect of contextualization on retrieval performance. For vertical contextualization, we separated three granularity levels for which we tested contextualization, namely content element, minor and major sections. Horizontal contextualization was tested with the content element level only, because the number of elements on that level allowed investigating the effect of horizontal distance. We experimented with INEX 2004 and 2005 test collections, swapping their roles as training and test collections.

The experiments show that utilizing the context enhances the retrieval of elements on any of the granularity levels. The improvements measured with MAP and nDCG are notable and the results of most topics are improved by contextualization. The results between the 2004 and 2005 collections show some inconsistency, because the small elements benefit in the 2004 collection but in the 2005 collection the larger elements benefit. The XML document collections were not fundamentally different in 2004 and 2005, and cannot be seen as the distinguishing factor of the slightly contradictory results. The features of the recall bases of the two collections are rather consistent as well (see Table 1); at least no dependencies between the number of relevant elements per topic (per relevant document) and the effectiveness of contextualization can be found. The 2005 collection seems more difficult, as the baseline result is clearly lower than the baseline of the 2004 collection. The only obvious difference between the collections is in the relevance assessment process. However, since the recall bases do not differ notably with regard to the number of relevant elements per topic, it is difficult to explain how the assessment process would affect contextualization. Nevertheless, it seems that the training of parameters was robust since the results obtained with the parameters trained in the other collection (test results) are very close to the results obtained with the parameters trained in the same collection (trained results).

The baseline results demonstrate that, in general, getting good performance is more challenging for short elements than for larger elements, at least with the average measures (MAP, nDCG@1500). We see two reasons for that. First, the textual evidence is scanter for the shorter elements. Second, it requires more of a system to accurately point out the small relevant

Table 6
MAP for different vertical contextualization magnitudes at the major section granularity level, obtained with INEX 2004 and 2005 data. The best results are in bold face.

	f								
	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5
Yr 2004	27.65	28.47	29.05	29.38	29.42	29.42	**29.64**	29.58	29.55
Yr 2005	27.59	28.04	28.11	**28.25**	28.22	28.18	27.95	27.85	27.74

Table 7
MAP for different vertical contextualization magnitudes and hierarchies at the minor section granularity level, obtained with INEX 2004 data. The best result is in bold face.

Hierarchy			*f*								
	p	*r*	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5
−1.00	1.00	0.00	22.19	23.01	23.19	23.25	22.98	22.83	23.25	22.58	22.37
−0.75	1.00	0.25	22.28	23.55	24.08	24.10	24.08	23.96	24.67	23.83	23.65
−0.50	1.00	0.50	23.04	24.08	24.44	24.75	24.72	24.76	25.38	24.50	24.34
−0.25	1.00	0.75	23.30	24.26	24.75	25.05	25.26	25.17	25.89	25.11	25.04
0.00	1.00	1.00	23.73	24.79	25.38	25.62	25.55	25.57	25.66	25.61	25.04
0.25	0.75	1.00	23.58	24.66	25.31	25.70	25.86	25.80	25.70	25.74	25.68
0.50	0.50	1.00	23.82	24.99	25.67	26.02	26.23	26.26	25.50	26.08	25.98
0.75	0.25	1.00	23.39	25.41	25.96	26.39	26.56	26.51	26.41	26.48	26.51
1.00	0.00	1.00	24.63	25.79	26.39	26.60	26.86	**26.92**	26.88	26.88	26.87

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Table 8
MAP for different vertical contextualization magnitudes and hierarchies at the minor section granularity level, obtained with INEX 2005 data. The best result is in bold face.

Hierarchy			*f*								
	p	*r*	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5
−1.00	1.00	0.00	23.01	23.36	23.19	23.15	23.12	23.00	23.06	22.63	22.47
−0.75	1.00	0.25	23.44	23.81	23.81	23.71	23.56	23.45	23.55	23.28	23.24
−0.50	1.00	0.50	23.69	24.06	23.98	23.88	23.80	23.78	23.82	23.64	23.63
−0.25	1.00	0.75	23.83	24.03	24.17	24.08	23.49	23.87	23.76	23.83	23.67
0.00	1.00	1.00	23.88	24.38	24.57	24.45	24.12	24.07	24.25	23.99	24.25
0.25	0.75	1.00	23.87	24.19	25.38	24.36	24.18	24.12	24.30	24.04	24.02
0.50	0.50	1.00	23.93	24.28	24.61	24.51	24.40	24.38	24.60	24.31	24.13
0.75	0.25	1.00	24.16	24.45	24.51	**24.66**	24.57	24.40	24.62	24.23	24.17
1.00	0.00	1.00	24.30	24.44	24.54	24.59	24.43	24.34	24.46	24.00	23.92

Table 9
MAP for different vertical contextualization magnitudes and hierarchies at the content element granularity level, obtained with INEX 2004 data. The best result is in bold face.

Hierarchy				*f*								
	p	*a*	*r*	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5
−1.50	0.50	0.00	0.00	14.98	15.18	15.01	14.76	14.42	14.19	13.38	13.57	13.39
−1.25	0.75	0.25	0.00	15.62	16.05	16.15	16.05	15.83	15.58	15.3	15.12	14.98
−1.00	1.00	0.50	0.00	15.84	16.38	16.55	16.47	16.28	16.10	15.79	15.58	15.43
−0.75	1.00	0.75	0.25	16.70	17.38	17.6	17.62	17.49	17.33	17.14	16.95	16.78
−0.50	1.00	1.00	0.50	17.09	17.91	18.22	18.23	18.17	18.07	17.88	17.73	17.60
−0.25	1.00	1.00	0.75	17.35	18.13	18.49	18.52	18.46	18.38	18.25	18.01	17.89
0.00	1.00	1.00	1.00	17.53	18.23	18.74	18.79	18.70	18.6	18.42	18.32	18.16
0.25	0.75	1.00	1.00	17.74	18.63	18.94	19.03	18.98	18.86	18.69	18.58	18.48
0.50	0.50	1.00	1.00	17.98	18.8	19.19	19.27	19.22	19.08	18.95	18.83	18.77
0.75	0.25	0.75	1.00	18.21	19.04	19.40	19.48	19.47	19.35	19.31	19.19	19.09
1.00	0.00	0.50	1.00	18.54	19.24	19.54	19.66	19.66	19.65	19.63	19.55	19.47
1.25	0.00	0.25	0.75	18.61	19.24	19.53	19.59	19.63	**19.71**	19.67	19.61	19.48
1.50	0.00	0.00	0.50	18.73	19.23	19.40	19.54	19.61	19.6	19.47	19.36	19.28

Table 10
MAP for different vertical contextualization magnitudes and hierarchies at the content element granularity level, obtained with INEX 2005 data. The best result is in bold face.

Hierarchy				*f*								
	p	*a*	*r*	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5
−1.50	0.50	0.00	0.00	10.82	11.36	11.72	11.76	11.24	10.80	10.48	10.2	10.04
−1.25	0.75	0.25	0.00	11.36	11.52	11.88	12.01	11.58	11.22	10.88	10.67	10.48
−1.00	1.00	0.50	0.00	11.36	11.62	11.97	12.05	11.70	11.36	11.06	10.83	10.63
−0.75	1.00	0.75	0.25	11.39	11.75	12.26	12.40	12.06	11.78	11.49	11.25	11.03
−0.50	1.00	1.00	0.50	11.75	12.08	12.26	12.51	12.26	11.97	11.59	11.29	11.10
−0.25	1.00	1.00	0.75	12.01	12.17	12.49	12.63	12.35	11.95	11.57	11.32	11.13
0.00	1.00	1.00	1.00	11.75	11.79	12.52	12.67	12.32	11.98	11.64	11.29	11.09
0.25	0.75	1.00	1.00	11.99	12.10	12.64	12.71	12.34	11.99	11.64	11.28	11.06
0.50	0.50	1.00	1.00	11.79	12.51	12.66	**12.75**	12.34	11.93	11.53	11.26	11.01
0.75	0.25	0.75	1.00	12.14	12.50	12.66	12.74	12.24	11.75	11.37	11.06	10.83
1.00	0.00	0.50	1.00	12.05	12.01	12.62	12.64	11.96	11.41	10.94	10.65	10.40
1.25	0.00	0.25	0.75	12.14	12.36	12.58	12.55	11.85	11.25	10.79	10.49	10.25
1.50	0.00	0.00	0.50	11.48	11.49	12.13	12.24	11.39	10.70	10.29	10.01	9.81

elements, while in larger elements the spotting of relevance is left for the user. The latter aspect becomes evident when looking at the ratios between the number of relevant elements and the total number of elements in Table 1. In other words, there are more non-relevant elements per a relevant element at lower granularity levels. A summary of the findings in this study is that the lack of textual evidence can be complemented with contextualization. Vertical contextualization is more effective than horizontal contextualization. It is worth noting that the benefit of contextualization is collection dependent so that the topical coherence between elements affects the impact. For instance, topically heterogeneous elements e.g. in a general encyclopedia do not benefit from contextualization.

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Table 11
MAP for different horizontal contextualization at the content element granularity level, obtained with INEX 2004 data. The best result is in bold face.

γ (10^{-2})	α (10^{-5})							
	0.8	1.0	1.5	2.0	4.0	6.0	8.0	10.0
1.0	13.81	13.81	13.60	13.46	13.11	12.95	12.90	12.87
1.5	14.65	14.65	14.40	14.20	13.80	13.59	13.43	13.30
2.0	15.24	15.24	15.01	14.85	14.39	14.12	13.94	13.80
2.5	**15.41**	**15.41**	15.39	15.24	14.85	14.58	14.39	14.24
3.0	15.39	15.39	15.36	15.29	15.23	14.96	14.77	14.63
3.5	15.23	15.23	15.28	15.30	15.23	15.15	15.06	14.93
4.0	14.97	14.97	15.09	15.24	15.26	15.23	15.21	15.07
4.5	14.79	14.79	14.93	15.00	15.20	15.21	15.21	15.20
5.0	14.43	14.72	14.85	15.08	15.21	15.19	15.22	14.31

Table 12
MAP for different horizontal contextualization at the content element granularity level, obtained with INEX 2005 data. The best result is in bold face.

γ (10^{-2})	α (10^{-5})							
	0.8	1.0	1.5	2.0	4.0	6.0	8.0	10.0
1.0	10.77	10.57	10.51	10.48	10.41	10.36	10.34	10.35
1.5	10.81	10.72	10.73	10.72	10.59	10.53	10.47	10.44
2.0	10.83	10.87	10.81	10.79	10.77	10.69	10.65	10.61
2.5	10.70	10.86	10.82	10.82	10.81	10.76	10.71	10.72
3.0	10.56	10.76	10.87	10.86	10.83	10.81	10.77	10.76
3.5	10.30	10.64	10.78	**10.90**	10.87	10.87	10.77	10.80
4.0	10.60	10.43	10.67	10.75	10.87	10.88	10.82	10.80
4.5	10.06	10.23	10.47	10.62	10.86	10.88	10.87	10.85
5.0	9.74	9.89	10.28	10.46	10.81	10.81	10.87	10.86

Acknowledgements

This study was funded by the Academy of Finland under Grants #140315, #115480 and #130482. The authors wish to thank the anonymous reviewers for their helpful comments.

Appendix A

The full XPath queries for minor and major section retrieval are presented here in this order. All the aliases for sections (sec,ss1,ss2) and back- and frontmatters (bm,fm) are defined as sections. Moreover, in order to get full coverage of the collection we added some rare elements (direct children of the bdy element):

A.1. Minor section

//*[self::fm or self::bm or self::sec or self::ss1 or self::ss2][not(.//sec or .//ss1 or .//ss2)] | //bdy/Fig. | //bdy/ip2 | //bdy/p | //bdy/ip1 | //bdy/index | //bdy/bq | //bdy/bib | //bdy/bib | //bdy/figw | //bdy/ack | //bdy/fn | //bdy/vt | //bdy/au | //bdy/snm | //bdy/dialog | //bdy/reviewer | //bdy/reviewers | //bdy/list.

A.2. Major section

//*[self::fm or self::bm or self::sec or self::ss1 or self::ss2][not(ancestor::fm or ancestor::bm or ancestor::sec or ancestor::ss1 or ancestor::ss2)] | //bdy/Fig. | //bdy/ip2 | //bdy/p | //bdy/ip1 | //bdy/index | //bdy/bq | //bdy/bib | //bdy/bib | //bdy/figw | //bdy/ack | //bdy/fn | //bdy/vt | //bdy/au | //bdy/snm | //bdy/dialog | //bdy/reviewer | //bdy/reviewers | //bdy/list.

Appendix B

See Tables 6–10.

Appendix C

See Tables 11 and 12.

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

References

Arvola, P., Junkkari, M., & Kekäläinen, J. (2005). Generalized contextualization method for XML information retrieval. In *Proceedings of the 14th ACM international conference on information and knowledge management, CIKM '05* (pp. 20–27). New York, NY: ACM.

Arvola, P., Junkkari, M., & Kekäläinen, J. (2006). Query evaluation with structural indices. In *Advances in XML information retrieval and evaluation, INEX 2005, LNCS 3977* (pp. 134–145). Berlin: Springer.

Arvola, P., Kekäläinen, J., & Junkkari, M. (2010). Focused access to sparsely and densely relevant documents. In *Proceedings of the 33rd annual international ACM SIGIR conference on research and development in information retrieval* (pp. 781–782). New York, NY: ACM.

Clark, J., & DeRose, S. (1999). XML path language (XPath). W3C Recommendation. <http://www.w3.org/TR/xpath>.

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems, 20*(4), 442–446.

Kamps, J., Koolen, M., & Lalmas, M. (2008). Locating relevant text within XML documents. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 847–848). New York, NY: ACM.

Kamps, J., Marx, M., de Rijke, M., & Sigurbjörnsson, B. (2005). Structured queries in XML retrieval. In *Proceedings of the 14th ACM international conference on information and knowledge management, CIKM '05* (pp. 4–11). New York, NY: ACM.

Kazai, G., Lalmas, M., & de Vries, A. P. (2004). The overlap problem in content-oriented XML retrieval evaluation. In *Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 72–79). New York, NY: ACM.

Kazai, G., Lalmas, M., & de Vries, A. P. (2005). Reliability tests for the XCG and inex-2002 metrics. In *Advances in XML information retrieval and evaluation, INEX 2004, LNCS 3493* (pp. 60–72). Berlin: Springer.

Kekäläinen, J., Arvola, P., & Junkkari, M. (2009). Contextualization. In *Encyclopedia of database systems* (pp. 174–178). USA: Springer.

Kekäläinen, J., Junkkari, M., Arvola, P., & Aalto, T. (2005). TRIX 2004: Struggling with the overlap. In *Advances in XML information retrieval and evaluation, INEX 2004, LNCS 3493* (pp. 127–139). Berlin: Springer.

Lalmas, M., & Piwowarski, B. (2005). INEX relevance assessment guide. In *INEX 2005 workshop pre-proceedings*. November 28–30, 2005, Schloss Dagstuhl (pp. 391–400). <http://inex.is.informatik.uni-duisburg.de/2005/pdf/inex-2005-preproceedings.pdf>.

Lalmas, M., & Tombros, A. (2007). Evaluating XML retrieval effectiveness at INEX. *SIGIR Forum, 41*(1), 40–57.

Malik, S., Lalmas, M., & Fuhr, N. (2005). Overview of INEX 2004. In *Advances in XML information retrieval, INEX 2004, LNCS 3493* (pp. 1–15). Berlin: Springer.

Mass, Y., & Mandelbrod, M. (2005). Component ranking and automatic query refinement for XML retrieval. In *Advances in XML information retrieval: Third international workshop of the initiative for the evaluation of XML retrieval, INEX 2004, LNCS 3493* (pp. 73–84). Berlin: Springer.

Ogilvie, P., & Callan, J. (2005). Hierarchical language models for XML component retrieval. In *Advances in XML information retrieval and evaluation, INEX 2004, LNCS 3493* (pp. 224–237). Berlin: Springer.

Ogilvie, P., & Lalmas, M. (2006). Investigating the exhaustivity dimension in content-oriented XML element retrieval evaluation. In *Proceedings of the 15th ACM international conference on information and knowledge management, CIKM '06* (pp. 84–93). New York, NY: ACM.

Pehcevski, J., & Piwowarski, B. (2009). Evaluation metrics for semi-structured text retrieval. In L. Liu & T. Özsu (Eds.), *Encyclopedia of database systems*. USA: Springer.

Sigurbjörnsson, B., Kamps, J., & de Rijke, M. (2004). An element-based approach to XML retrieval. In *INEX 2003 workshop proceedings, INEX 2003* (pp. 19–26).

Trotman, A., & Sigurbjrnsson, B. (2004). Narrowed extended XPath I (NEXI). In *Advances in XML information retrieval, INEX 2004, LNCS 3493* (pp. 16–40). Berlin: Springer.

Please cite this article in press as: Arvola, P., et al. Contextualization models for XML retrieval. *Information Processing and Management* (2011), doi:10.1016/j.ipm.2011.02.006

Study V

Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2006) Query evaluation with structural indices. In *Proceedings of 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005, Lecture Notes in Computer Science 3977,* Springer-Verlag Berlin Heidelberg, 134-145.

Query Evaluation with Structural Indices

Paavo Arvola[1], Jaana Kekäläinen[2], and Marko Junkkari[1]

[1] Department of Computer Sciences, Kanslerinrinne 1,
33014 University of Tampere, Finland
junken@cs.uta.fi, paavo.arvola@uta.fi
[2] Deparment of Information Studies, Kanslerinrinne 1,
33014 University of Tampere, Finland
jaana.kekalainen@uta.fi

Abstract. This paper describes the retrieval methods of TRIX system based on structural indices utilizing the natural tree structure of XML. We show how these indices can be employed in the processing of CO as well as CAS queries, which makes it easy for variations of CAS queries to be processed. Results at INEX 2005 are discussed including the following tasks: CO.Focussed, CO.FetchBrowse, CO.Thorough and all of the CAS tasks. While creating result lists, two different overlapping models have been applied according to task. The weights of the ancestors of an element have been taken into account in re-weighting in order to get more evidence about relevance.

1 Introduction to TRIX Retrieval System

The present study comprises of retrieval experiments conducted within the INEX 2005 framework addressing the following research questions: ranking of elements of 'best size' for CO queries, query expansion, and handling of structural conditions in CAS queries. In INEX 2005 we submitted runs for the following tasks: CO.focussed, CO.thorough, CO.FetchBrowse, and all of the CAS tasks.

Next we introduce the TRIX (Tampere information retrieval and indexing of XML) approach for indexing, weighting and re-weighting. Then, Section 2 describes the processing of CAS queries and in Section 3 the results of INEX 2005 are presented and analyzed. Finally conclusions are given in Section 4. Graphical representations of our official results are given in the appendix.

1.1 Structural Indices and Basic Weighting Schema

In TRIX the management of structural aspects is based on the *structural indices* [2,4,5,8]. The idea of structural indices in the context of XML is that the topmost (root) element is indexed by ⟨1⟩ and its children by ⟨1,1⟩, ⟨1,2⟩, ⟨1,3⟩ etc. Further, the children of the element with the index ⟨1,1⟩ are labeled by ⟨1,1,1⟩, ⟨1,1,2⟩, ⟨1,1,3⟩ etc. This kind of indexing enables analyzing of the relationships among elements in a straightforward way. For example, the ancestors of the element labeled by ⟨1,3,4,2⟩ are associated with the indices ⟨1,3,4⟩, ⟨1,3⟩ and ⟨1⟩. In turn, any descendant related to the index ⟨1,3⟩ is labeled by ⟨1,3,ξ⟩ where ξ is a non-empty

N. Fuhr et al. (Eds.): INEX 2005, LNCS 3977, pp. 134 – 145, 2006.

part of the index. In the present approach the XML documents in the collection are labeled by positive integers 1, 2, 3, etc. From the perspective of indexing this means that the documents are identified by indices ⟨1⟩, ⟨2⟩, ⟨3⟩, etc., respectively. The length of an index ξ is denoted by $len(\xi)$. For example $len(\langle 1,2,2,3\rangle)$ is 4. *Cutting operation* $\delta_i(\xi)$ selects the subindex of the index ξ consisting of its i first integers. For example if $\xi = \langle a,b,c\rangle$ then $\delta_2(\xi) = \langle a,b\rangle$. In terms of the cutting operation the root index at hand is denoted by $\delta_1(\xi)$ whereas the index of the parent element can be denoted by $\delta_{len(\xi)-1}(\xi)$.

The retrieval system, TRIX, is developed further from the version used in the 2004 ad hoc track [3] and its basic weighting scheme for a key k is slightly simplified from the previous year:

$$w(k,\xi) = \frac{kf_\xi}{kf_\xi + v \cdot \left((1-b) + b \cdot \frac{\xi f_c}{\xi f_k}\right)} \cdot \frac{log\left(\frac{N}{m}\right)}{log(N)} \tag{1}$$

where

- kf_ξ is the number of times k occurs in the ξ element,
- N is the total number of content elements in the collection,
- m is the number of content elements containing k in the collection,
- ξf_c is the number of all descendant content elements of the ξ element
- ξf_k is the number of descendant content elements of the ξ element containing k,
- v and b are constants for tuning the weighting.

This formula is utilized only for such elements where kf_ξ is greater than 0. This ensures that the ξf_c and ξf_k are equal or greater than 1, because we define that the lowermost referable element, the content element, contains itself. Otherwise the weight of an element for the key k is 0. The constants v and b allow us to affect the 'length normalization component' ($\xi f_c / \xi f_k$) or LNC and tune the typical element size in the result set. In our runs for INEX 2005 b is used for tuning, while v is set to 2. Small values of b (0-0.1) yield more large elements, whereas big values (0.8-1) yield more small elements. This is because the LNC tends to be large in small matching elements; it is likely that the smaller the ξf_k value is the bigger is the LNC. A large b value emphasizes the LNC component, whereas a small one the key frequency. While b is set to 0, the system considers the root element always to be the best one in a document, because in case of two overlapping elements have the same weight, the ancestor one is privileged. Table 1 shows the average distribution of top 100 elements in our result lists (Content only), when b is set to 0.1 and to 0.9. Testing the parameters in INEX collections has shown that value 2 for v gives a smooth overall performance and ranging b allows tuning the size of the elements in the result list. The underline overlap percentage is 0. In the table the '+' sign means all the equivalent tags. E.g. p+ means all paragraph tags: p, ip1, ip2 etc.

Table 1. The average distribution of top 100 elements, when b is set to 0.1 and to 0.9

	p+	sec+	bdy	article
b = 0.9	31.7	14.1	19.3	0.4
b = 0.1	8.8	13.1	42.2	8.6

The weighting formula yields weights scaled into the semi-closed interval (0,1]. The weighting of phrases and the operations for + and - prefixes have the same property. They are introduced in detail in [3]. A *query term* is a key or phrase with a possible prefix + or -. A CO query q is a sequence of query terms $k_1, \ldots, k_n$. In relevance scoring for ranking the weights of the query terms are combined by taking the average of the weights:

$$w(q,\xi) = \frac{\sum_{i=1}^{n} w(k_i,\xi)}{n} \tag{2}$$

After this basic calculation elements' weights can be re-weighted. Next we consider the used re-weighting method, called contextualization.

1.2 Contextualization

In our runs we use a method called contextualization to rank elements in more effective way in XML retrieval [1, see also 7]. Re-weighting is based on the idea of using the ancestors of an element as a context. In terms of a contextualization schema the context levels can be taken into account in different ways. Here we applied four different contextualization schemata.

1) Root (denotation: $c_{r1.5}(q, \xi)$)
2) Parent (denotation: $c_p(q, \xi)$)
3) Tower (denotation: $c_t(q, \xi)$)
4) Root + Tower (denotation: $c_{rt}(q, \xi)$)

A contextualized weight is calculated using weighted average of the basic weights of target element and its ancestor(s), if exists. Root contextualization means that the contextualized weight of an element is a combination of the weight of an element and its root. In our runs the root is weighted by the value 1.5. This is calculated as follows:

$$c_{r1.5}(q, \xi) = \frac{w(q,\xi) + 1.5 * w(q,\delta_1(\xi))}{2.5} \tag{3}$$

Parent contextualization for an element is an average of the weights of the element and its parent.

$$c_p(q, \xi) = \frac{w(q,\xi) + w(q, \delta_{len(\xi)-1}(\xi))}{2} \tag{4}$$

Tower contextualization is an average of the weights of an element and all its ancestors.

$$c_t(q, \xi) = \frac{\sum_{i=1}^{len(\xi)} w(q, \delta_i(\xi))}{len(\xi)} \tag{5}$$

So called Root + Tower contextualizaton means the plain tower contextualization with root multiplied by two. This can be seen as a combination of parent and root contextualizations.

$$c_{rt}(q, \xi) = \frac{w(q, \delta_1(\xi)) + \sum_{i=1}^{len(\xi)} w(q, \delta_i(\xi))}{len(\xi) + 1} \tag{6}$$

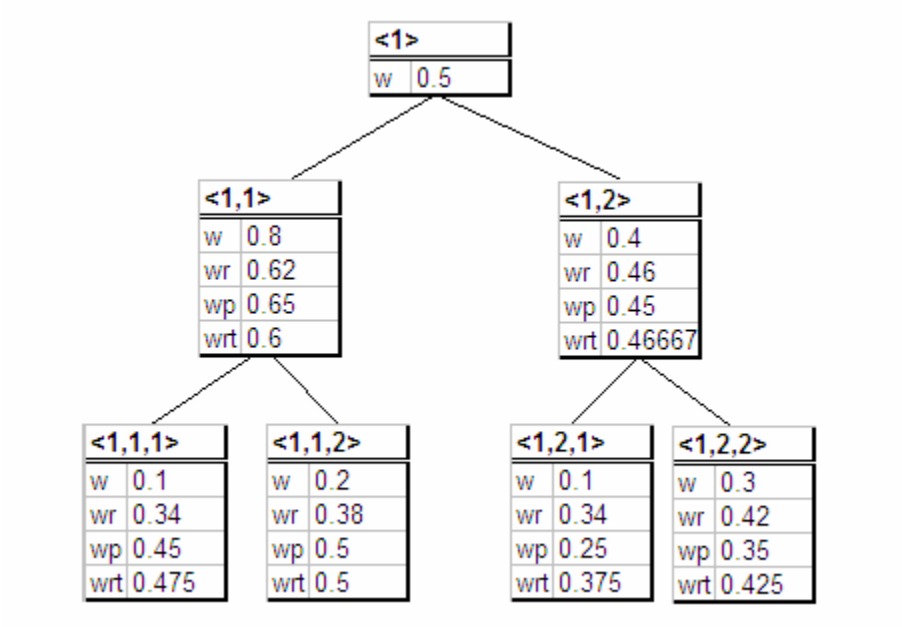

Fig. 1. A tree presentation of an XML document illustrating different contextualization schemata

In Figure 1 the effects of the present contextualization schemata are illustrated. The basic weights are only sample values. In it, XML tree with elements assigned initial weights (w) and contextualized weights: Root (wr), Parent (wp) and Root + Tower (wrt) is given. For instance, element with index ⟨1,1,2⟩ has an basic weight of 0.2. Parent contextualization means an average weight of ⟨1,1,2⟩ and ⟨1,1⟩. Root is the weighted average of ⟨1,1,2⟩ and ⟨1⟩ where the weight of ⟨1⟩ has been multiplied by 1.5. Root + Tower is the weighted average of weights of ⟨1⟩ , ⟨1,1⟩ and ⟨1,1,2⟩, where the weight of ⟨1⟩ has been calculated twice.

In [1] we have discovered that a root element carries the best evidence related to the topics and assessments of INEX 2004. However, contextualizing the root only has an effect on the order of elements in the result list, and it does not change the order of elements within a document. Generally, if we contextualize the weights of elements x and y with the weight of their ancestor z, the order of x and y will not change in the result list. Further, the mutual order of x, y and z will not change if no re-weighting (i.e. contextualization) method is applied to element z. The root element (article) possesses no context in our approach. Hence in the CO.FetchBrowse task, where documents have to be ordered first, the Root contextualization will not have an effect on the rankings of other elements. However, within a document there are still several other context levels, and by utilizing those levels, it is possible to re-rank elements within a document. This finding has been utilized in the CO.FetchBrowse task.

1.3 Handling the Overlap in Results

In Figure 2 two overlap models, which our system supports, are illustrated. First, an element to be returned is marked with a letter P. On the left side there is a situation where all overlapping elements are excluded from the result list, even if their weight would be sufficient, but smaller than P. In other words, P has higher score than its descendants or ancestors. The model indicates that the overlap percentage is 0. On the right side all elements can be accepted, regardless of their structural position in the document.

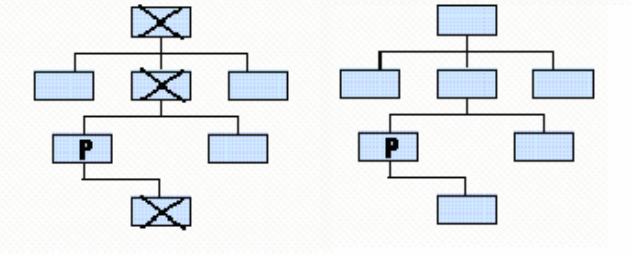

Fig. 2. Two overlap models

We have used the former (left) model in the CO.Focussed and CO.FetchBrowse tasks and the latter (right) model in the CO.Thorough and all of the CAS tasks. Next, we introduce the overall processing of CAS and the structural constraints involved.

2 Processing CAS-Queries

In the CAS queries an element may have constraints concerning itself, its ancestors or descendants. These constraints may be only structural, or structural with content. For instance in query

//A[about(.,x)]//B[about(.//C,y)]

B is the structural constraint of a target element itself. A is a structural constraint of a target element's ancestor, and is C target element's descendants. All of these structural constraints have also content constraints, namely *x* or *y*. So, to be selected to a result list, an element must fulfil these constraints. The processing of CAS queries can be divided into four steps:

- First step: Generate a tree according to the target element's content constraint, and weight elements, which fulfil the target element's structural constraint.
- Second step: Discard all the target elements which do not fulfil the structural ancestor and descendant constraints. Due to the nature of hierarchical data, ancestors are always about the same issue as their descendants, i.e. they share the descendants' keys. So the content constraints of descendant elements are taken into account here as well.
- Third step: Generate trees according to each ancestor element's content constraint. Discard elements, where the structural descendant and ancestor content constraint are not fulfilled, i.e. corresponding elements do not exist in any sub tree.
- Fourth step: Collect the indices of elements left in the third step fulfilling the ancestor structural constraint, and discard all of the target elements, which do not have such indices among ancestor elements.

To clarify this, processing of a CAS query can be demonstrated with a sufficiently complex example.

A query:

//article[about(.//abs, logic programming)]//bdy//sec[about(.//p, prolog)]

breaks down into following parts:

- an element with structural constraint **sec** is the target element with content constraint *prolog*
- **p** is a structural descendant constraint of the target element with the same content constraint as **sec :** *prolog*
- **article** is a structural ancestor constraint of the target element with a content constraint *logic programming*
- **abs** is a structural descendant constraint of **article** with the same content constraint *logic programming*
- **bdy** is a structural ancestor constraint of the target element without any content constraints

In the first step, shown in Figure 3, we form a tree of elements with non-zero weights according to the query *prolog*. In other words all the elements with zero weights are discarded from an XML tree structure.

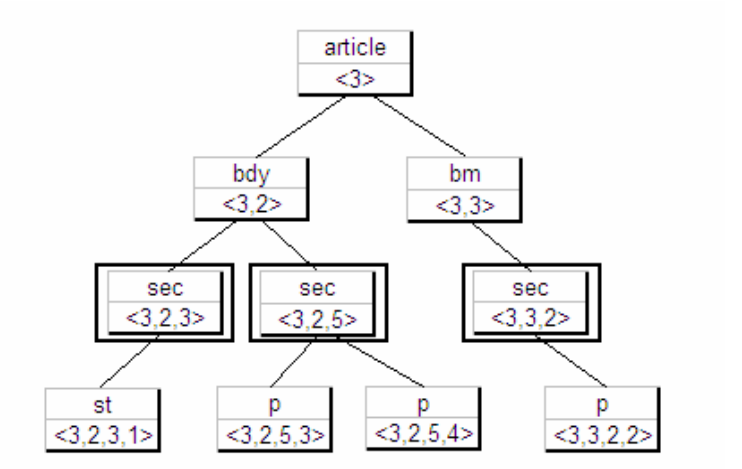

Fig. 3. A tree presentation of a sample XML document having only elements with a weight greater than 0 according to the query *prolog*

In the second step (Figure 4), we exclude target element ⟨3,3,2⟩, because the structural ancestor constraint **bdy** is not fulfilled. Element ⟨3,2,3⟩ is also to be excluded, because the descendant constraint **p** is not fulfilled.

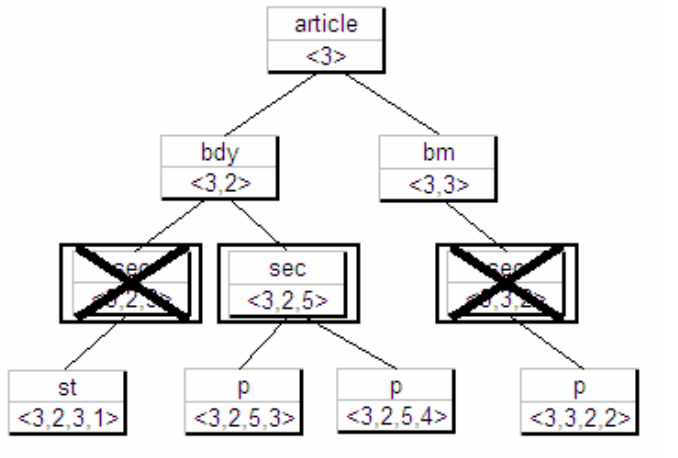

Fig. 4. A tree presentation of a sample XML document having only elements with a weight greater than 0 according to the query *prolog*, where target elements not fulfilling the constraints are excluded

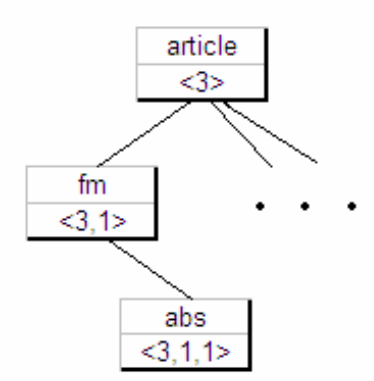

Fig. 5. A tree presentation of a sample XML document having only elements with a weight greater than 0 according to the query logic programming

In the third step we form a tree with non-zero weights according to the query *logic programming*, as seen in Figure 5.

In the tree, there is an **abs** element as a descendant of **article**, so both of the structural and content constraints are fulfilled. Hence, we take the index of the article: ⟨3⟩, and see that the index belongs to a descendant of the remaining target element ⟨3,2,5⟩. So, this and only this element is to be returned from this document.

2.1 Taking Vagueness into Account in CAS

In the current evaluations there are four different kinds of interpretations for structural constraints for processing NEXI, in our approach the structural constraints are interpreted strictly. However for SVCAS, VSCAS and VVCAS the query has been modified. Our system handles vague interpretation so that the corresponding element names have been ignored. In NEXI language this can be implemented by replacing the names with a star. Thus we have modified CAS queries as follows:

The initial CAS query (and SSCAS):
//A[about(.,x)]//B[about(.,y)]

SVCAS:
//*[about(.,x)]//B[about(.,y)]

VSCAS:
//A[about(.,x)]//*[about(.,y)]

VVCAS would then logically correspond to:
//*[about(.,x)]//*[about(.,y)]

For simplification we have processed VVCAS like a CO query. In the present example VVCAS corresponds to the query: //*[about(.,x y)].

3 Results

3.1 CAS Runs

In the content and structure queries, only elements which fulfil the constraints are accepted to the results. The ranking of the elements has been done according to the target element's textual content. Besides the target element, other content constraints have been taken into account as a full match constraint without any weighting. This full match content constraint within a structural constraint has been interpreted in disjunctive way. It means, that only one occurrence of any of the keys in a sub query is sufficient enough to fulfil the condition. For instance in the query

```
//A[about(.,x y z)]//B[about(.//C,w)]
```

for B to be returned, it is sufficient that the A element includes only one of the keys *x, y* or *z*. Naturally the B element should be about *w*, and also have a descendant C about *w*. This approach, the CAS processing with structural indices and the TRIX matching methods lead to fairly good results with the generalized quantization in all of the CAS tasks and especially in the SSCAS task (see Figure 6 in Appendix). Also the CO-style run in the VVCAS task (Figure 9) worked out fairly well.

There was a slight error in our submissions of results. Accidentally we sent runs intended for SVCAS for VSCAS, and vice versa. Figures 7 and 8 show the situation, where the "*should have been*"-runs are the thick upper ones in the nXCG curves. The overload of elements of wrong type led to a quite rotten score in SVCAS. Surprisingly, despite the error, VSCAS results proved to be quite satisfactory. Especially, according to the early precision of our runs, the ranking was as high as 3rd and 4th in the generalized quantization and 3rd and 8th in the strict quantization of the nXCG metrics. However, in general the right interpretation of both of those tasks leads to a substantial improvement of effectiveness (see Figures 7 and 8).

3.2 CO Runs

In the CO runs we have used Root+Tower contextualization (Tampere_..._tower), and Root contextualization (Tampere_..._root). In addition we have applied a query expansion method from Robertson [6], taking 5 or 10 expansion words from 7 top documents from the first result set (Corresponding runs: Tampere_exp5_b09_root, Tampere_exp10_b01_root). Figure 10 shows the slight improvement of the expanded run compared with a similar run without any expansion.

Because of the prevention of overlapping elements, promoting large elements may not be wise in the focussed task. That is because if a large element is returned, then every descendant is excluded from the results. However, in thorough task promoting large elements is not that risky. Hence, we used small *b* values for the thorough and large values for the focussed runs. Favouring small elements might have caused another kind of problem, though. In the relevance assessments many of the paragraph sized elements are marked as too small. That leads to a situation, where a whole relevant branch is paralyzed, when a too small leaf element is returned.

In the topic 229 there is a spelling error "latent semantic anlysis", which in our system would lead to a poor score. To minimize the error rate and also to improve recall, we have opened the phrases in all of the queries. For instance, query *"latent semantic anlysis*" would become *"latent semantic anlysis" latent semantic anlysis*. A manual correction of the mistake improves overall performance by 1-2 percentage depending on the task. These features and also the effect of the contextualization improve recall and scores in the generalized quantization, although the early precision suffers slightly (see Figures 10 and 11). A run without contextualization improves the early precision from 0,1657 to 0,2401 in CO.Focussed task with generalized quantization (nxCG@10). Accordingly the ep/gr value improves slightly as well.

4 Conclusions

This paper presents our experiments and results at INEX 2005. The results for the CO task show that Root contextualization is not generally better than Root + Tower, except for the early precision. In general, our approach is in many runs quite recall oriented, and we also do better in the generalized than strict quantization. Therefore, improving top precision in all tasks and quantizations remains as one of our primary goals.

This was the first time we participated in (strict) CAS task. The analyzing power of structural indices enables a straightforward processing of CAS queries. In addition, results in INEX 2005 give a good baseline for future development.

References

1. Arvola, P., Junkkari, M., and Kekäläinen, J.: Generalized Contextualization Method for XML Information Retrieval. In Proceedings of ACM Fourteenth Conference on Information and Knowledge Management (CIKM'2005), (2005) 20-27.
2. Junkkari, M.: PSE: An object-oriented representation for modeling and managing part-of relationships. Journal of Intelligent Information Systems, 25(2), (2005) 131-157.
3. Kekäläinen, J., Junkkari, M., Arvola, P., and Aalto, T.: TRIX 2004: Struggling with the overlap. In Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004. LNCS 3493. Springer, Heidelberg, (2005) 127-139.
4. Knuth, D.: Fundamental Algorithms: The Art of Computer Programming. Vol. 1, Addison Wesley, (1968).
5. Niemi, T.: A Seven-Tuple Representation for Hierarchical Data Structures. Information Systems, 8(3), (1983) 151-157.
6. Robertson, S.E. and Walker, S.: Okapi/Keenbow at TREC-8, Proc. NIST Special Publication 500-246: The Eighth Text Retrieval Conference Text (TREC), (1999) 151-162.
7. Sigurbjörnsson, B., Kamps J., and de Rijke, M.: An Element-Based Approach to XML Retrieval. In INEX 2003 Workshop Proceedings (2003) 19-26.
8. Tatarinov, I., Viglas, S., Beyer, K.S. Shanmugasundaram, J., Shekita, E.J., and Zhang C.: Storing and Querying Ordered XML Using a Relational Database System. In Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, (2002) 204-215.

Appendix

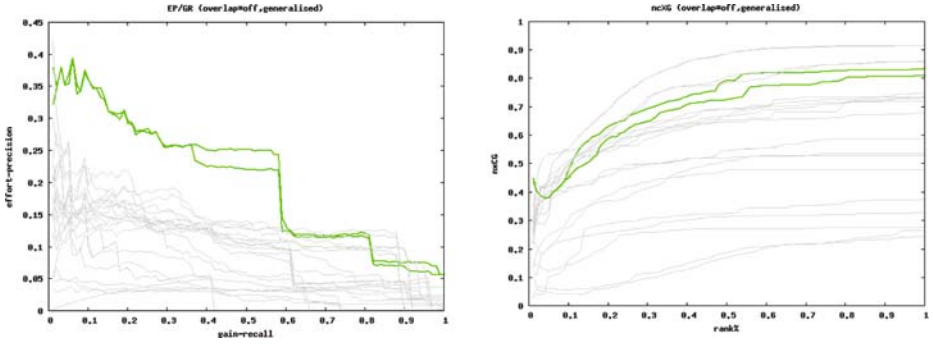

Fig. 6. SSCAS: The EP/GR and nXCG curves of the generalized quantization

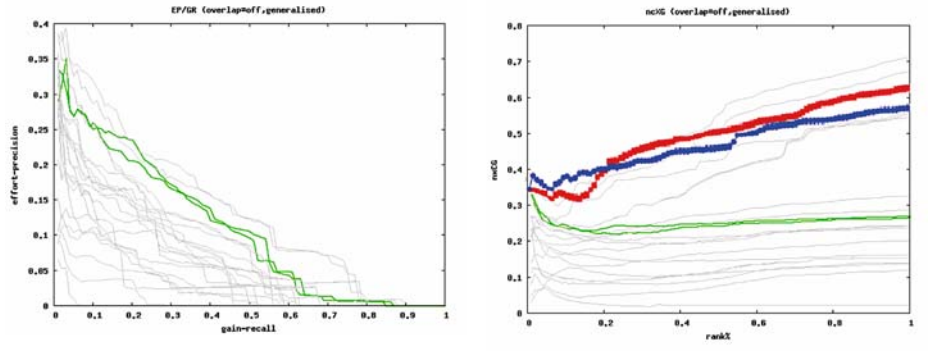

Fig. 7. VSCAS: The EP/GR and nXCG curves of the generalized quantization

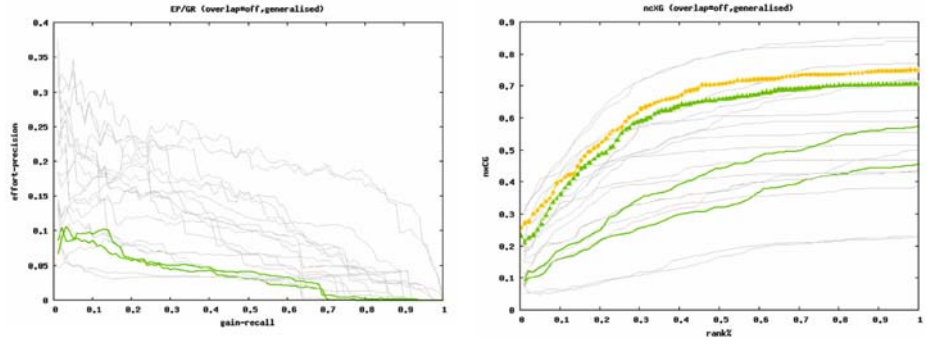

Fig. 8. SVCAS: The EP/GR and nXCG curves of the generalized quantization

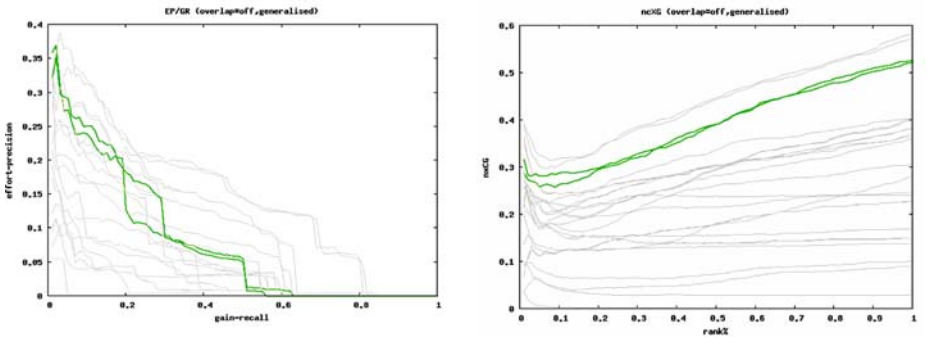

Fig. 9. VVCAS: The EP/GR and nXCG curves of the generalized quantization

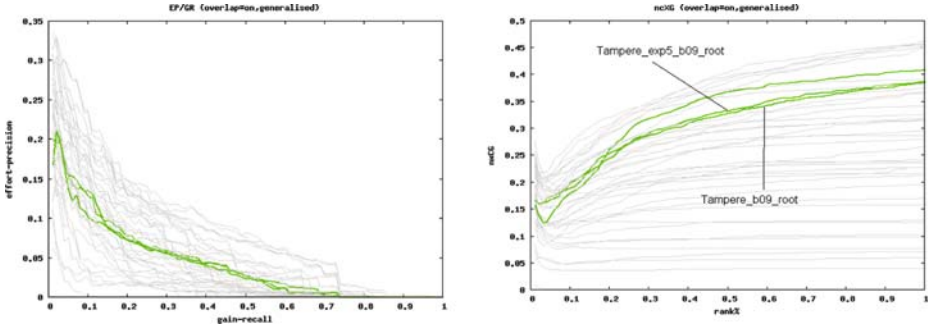

Fig. 10. CO.Focussed: The EP/GR and nXCG curves of the generalized quantization

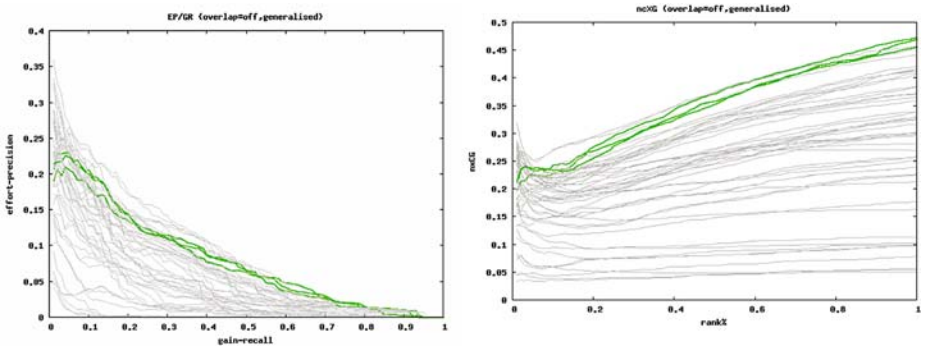

Fig. 11. CO.Thorough: The EP/GR and nXCG curves of the generalized quantization

Study VI

Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2010) Expected Reading Effort in Focused IR Evaluation, *Information Retrieval*, Volume 13, Number 5, Springer Science+Business Media, 460-484.

Inf Retrieval (2010) 13:460–484
DOI 10.1007/s10791-010-9133-9

Expected reading effort in focused retrieval evaluation

Paavo Arvola · Jaana Kekäläinen · Marko Junkkari

Received: 1 May 2009 / Accepted: 14 April 2010 / Published online: 6 May 2010

Abstract This study introduces a novel framework for evaluating passage and XML retrieval. The framework focuses on a user's effort to localize relevant content in a result document. Measuring the effort is based on a system guided reading order of documents. The effort is calculated as the quantity of text the user is expected to browse through. More specifically, this study seeks evaluation metrics for retrieval methods following a specific fetch and browse approach, where in the fetch phase documents are ranked in decreasing order according to their document score, like in document retrieval. In the browse phase, for each retrieved document, a set of non-overlapping passages representing the relevant text within the document is retrieved. In other words, the passages of the document are re-organized, so that the best matching passages are read first in sequential order. We introduce an application scenario motivating the framework, and propose sample metrics based on the framework. These metrics give a basis for the comparison of effectiveness between traditional document retrieval and passage/XML retrieval and illuminate the benefit of passage/XML retrieval.

Keywords Passage retrieval · XML retrieval · Evaluation · Metrics · Small screen devices

1 Introduction

The traditional information retrieval (IR) considers a document to be an atomic retrievable unit. Since not all content of a document is relevant according to a query, it is useful to retrieve smaller parts e.g. with an XML retrieval system or a system retrieving arbitrary passages. This enables a more specific retrieval strategy and allows a system to focus on

P. Arvola (✉) · J. Kekäläinen
Department of Information Studies and Interactive Media, University of Tampere, Tampere, Finland
e-mail: paavo.arvola@uta.fi

J. Kekäläinen
e-mail: jaana.kekalainen@uta.fi

M. Junkkari
Department of Computer Sciences, University of Tampere, Tampere, Finland
e-mail: junken@cs.uta.fi

parts of documents. Thus, content-oriented XML retrieval and passage retrieval are beneficial in reducing a user's effort in finding the best parts of a document.

From the evaluation perspective, the fundamental difference between content-oriented XML retrieval and passage retrieval is that in XML retrieval the passages are marked-up as elements, i.e. text is between the element's start and end tags, whereas in passage retrieval the passages are not dependent on element boundaries. In this study the term passage retrieval is extended to concern content-oriented XML retrieval as well.

The retrieved passages can be grouped in many ways. This study follows a specific fetch and browse approach (Chiaramella et al. 1996). In the fetch phase documents are ranked in decreasing order according to their document score, just like in the traditional document retrieval. In the browse phase, a set of non-overlapping passages representing the relevant text within a document is retrieved, and the retrieval system interface turns the searcher's attention to the relevant parts of the documents. The matching method, including the selection of appropriate (best matching) passages, defines *what* the user is expected to browse. The user interface, in turn, specifies *how* the user is expected to browse the content. The co-operative action affects the reading order of the passages, and the effort the user has to spend in localizing the relevant content in the document. This effort can be measured by the quantity of text the user is expected to browse through.

The amount of text can be measured e.g. with words, sentences or windows of characters. We have chosen to use characters as the measurable units of the user effort. Characters are the smallest atomic units of text to read, retrieve and evaluate, and we assume a character to be read with a constant effort. This is tolerable while by comparison in document retrieval the effort of reading a whole document is treated as a constant regardless of the size of the document and other qualities.

Considering characters to be retrievable units, any text document can be modeled as a character position list, starting basically from the first character (at position 1) and ending at the last character (at position n) of the document (n is the length of the document). Other characters are in sequential order in between.

This order can typically be considered as the author's intended reading order of the document. In other words, the author 'expects' the reader to follow the sequential order of the document. Thus, the corresponding character position list for the intended reading order is $\langle 1,2,3,\ldots,n \rangle$. Figure 1 illustrates this.

However, in reality this order applies only for a person who is a strict sequential reader; other kinds of readers probably follow some other order of their own, based on numerous uncontrollable factors beyond the author's intention (see Hyönä and Nurminen 2006). Yet, any reading order can also be modeled as a character position list, but the order of the list items follows rather a temporal pattern of the reader's behavior and the list may differ more or less from the intended reading order character position list.

The user behavior is not a totally independent variable. Namely, the usage of a passage retrieval system in co-operation with a user interface provides means to guide browsing within a document, and to break the intended reading order and re-organize the expected browsing order. This kind of system guided reading order is supposed to allow more focused access to the relevant content in a document. Thus, the aim of this study is to provide an evaluation framework based on the guided reading order within a document.

In addition to the expected reading order, one might expect that the user is not willing to browse through all irrelevant material, if a lot of such material is met. Instead, the user will stop at some point and move onto the next document (or reformulate the query). This means that not all relevant content of a document, if any, may be encountered in the event the user has to read a lot of irrelevant material.

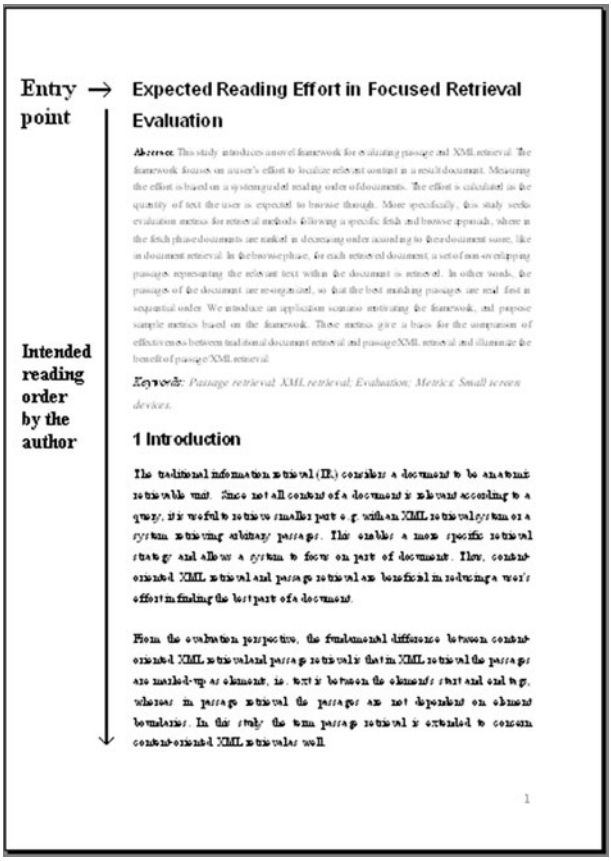

Fig. 1 Intended reading order of a document by the author (Greek and Latin based scripts)

The benefit of the evaluation framework is twofold: First, since the user effort is based on characters to be read, it is credible and adjustable for various user interfaces and user scenarios. Second, it gives a basis for effectiveness comparison between traditional document retrieval and passage/XML retrieval. The latter illuminates the benefit of passage/XML retrieval, which has been questioned by Kamps et al. (2008a).

For an IR system's tasks in browsing a document, we propose two approaches: (1) to quickly assess the document to be relevant (or not relevant) and (2) to browse through the relevant content of a document effectively. For the two tasks we introduce two metrics in Sect. 4, namely *cumulated effort* for the first and *character precision-recall* for the other.

To motivate and clarify the evaluation framework, we present a sample user interface scenario in Sect. 2. However, it is worth noting that the presented framework is independent of the sample scenario. Section 3 reviews related studies. In Sect. 4 we introduce two metrics with sample measures, and test results on Wikipedia data in Sect. 5.

2 Motivating sample scenario

The reading order of a document depends on the co-operative action of the user interface and the passage matching method. Therefore, as a sample scenario and a basis of evaluations, we introduce an interface, which is a slightly simplified variant of the interface

described by Arvola et al. (2006). This scenario motivates the present study and is the basis of the experiments, but the evaluation framework is not bound to any of its features (incl. screen size or other technical details). It is necessary to emphasize that this sample scenario is only illuminating.

Passage and element retrieval provide focused access to documents. The size of passages/elements may vary but they are always accessed through a window, size of which depends on the media and device. This feature is stressed when using a device with limited screen space, such as a mobile phone. The small screen forces the user's attention to the position the screen is showing, and thus the expected user behavior is more predictable.

A small screen is one of the major constraints for a mobile device. Because of that, several approaches in preventing horizontal scrolling are introduced (Buyukkokten et al. 2000; Jones et al. 1999). Our sample scenario follows the Opera Mini browser (Opera 2006) outline, where the textual content of a document is rendered in one column. Nevertheless, conventional browsing through a long text document with such a device requires a lot of vertical scrolling.

In our interface the effort of finding relevant content is reduced by inserting hyperlinks into anchors that, in turn, are placed at the matching locations of the document according to the initial query expression. The user is directed to the supposedly relevant parts of the document.

This user interface not only reduces the user's effort in reduced vertical scrolling but also preserves the original document order and the structure of the document. In other words the document is represented without breaking up the continuity of the initial textual content presentation of the document. The conventional browsing methods within the document are also available. Consequently, the user is expected to navigate through the document in the fashion described in Sects. 2.1 and 2.2.

The anchoring of passages is done simultaneously with rendering documents to the standard XHTML format for viewing and browsing in a device independent way. In XML retrieval, our method is especially suited for content-oriented online XML collections, such as the Wikipedia XML collection used in INEX 2008 (Denoyer and Gallinari 2006). Next, we give a detailed view of the system.

2.1 Interface overview

In a search process the user inserts keywords with available text input methods in order to perform a search. In order to retrieve the best matching passages from the best matching documents, the retrieval follows the fetch and browse approach. In a nutshell, documents are first sorted according to their retrieval status value, and after that the passages are clustered by documents. Phase 1 is plain full document retrieval where, according to the query expression, the system presents a result list with links to the documents in a matching order. Figure 2 illustrates this phase. Phase 2 is element retrieval and it is done for a single document selected by the user.

The usage of the interface follows strictly the fetch and browse approach. In Fig. 2 the query: 'matching method' is forwarded into the IR system. This launches the full document retrieval phase and the system presents a result list, with the current document on top of it. Preferably, the user selects this document from the result list by clicking a pointing link. Clicking the link triggers Phase 2 (Fig. 3). In this phase, the system marks up the best matching parts of the document. Thereafter the system renders the resulting document into an XHTML document for viewing and browsing. This includes also inserting anchors into the beginning of the best matching parts. Finally, the browser shows the beginning of the result document to the user.

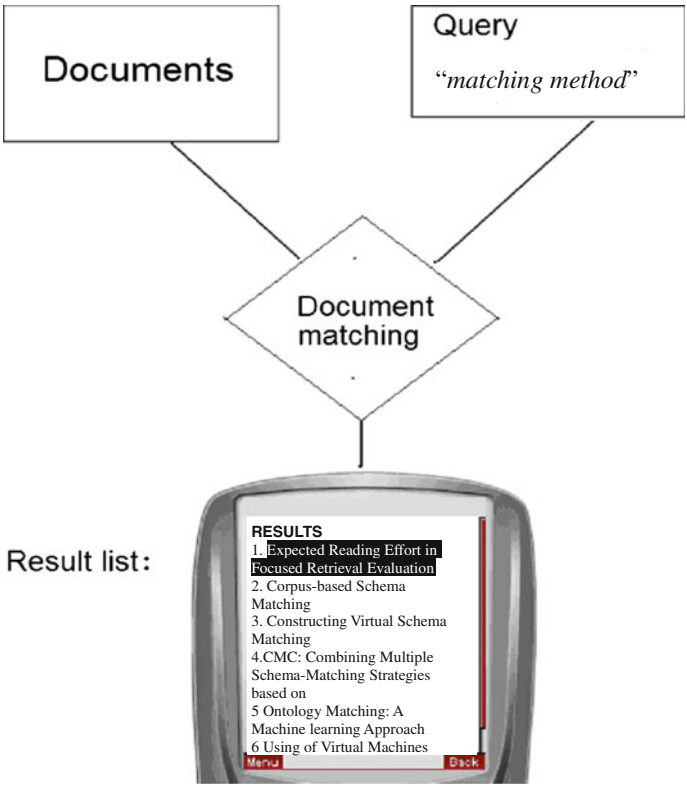

Fig. 2 Phase 1 (fetch)—document ranking

2.2 Creating a matching chain by linking the Best parts with anchors

In the present user interface scenario, there are two arrow icons at the beginning of the resulting document. The first one, an arrow down, is a link to the first anchor at the beginning of the first matching element. The arrow left is a link to getting back to the result list. The arrow down hyperlink is for relevance browsing within the document. By clicking it, the user ends up to the point the anchor is at. At the end of the matching passage, the arrows are presented again. Now the arrow down is a link to the next matching (not overlapping) part of the document.

For instance in Fig. 3, the user selects the current document from the result list. The system places two arrow hyperlinks to the top left corner of the result document. In Fig. 4 the current user interface focus is on the first hyperlink, which is represented by an arrow down hyperlink. By clicking the link the user moves down to the place the anchor is at. If the matching works perfectly the anchor is just before a relevant part in the result document. Now the user scrolls and reads the whole section, which is estimated to be relevant by the retrieval system. Because there are no further relevant parts in the document at hand, at the end of the section there are only two hyperlinks: back to results and back to the beginning of the document. In case there were other relevant passages further down in the document, there would be an arrow down hyperlink to the start of the next matching passage and so on.

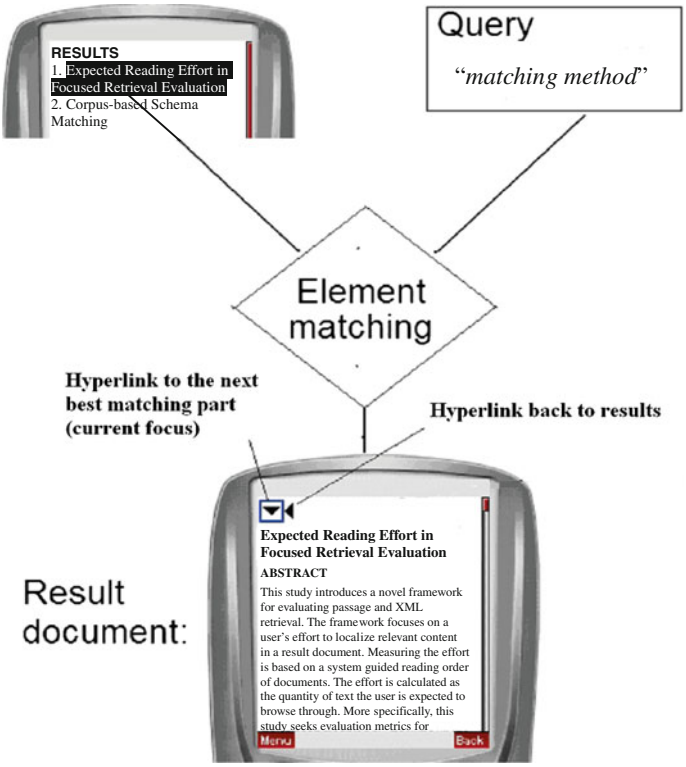

Fig. 3 Phase 2 (browse)—relevant in document retrieval

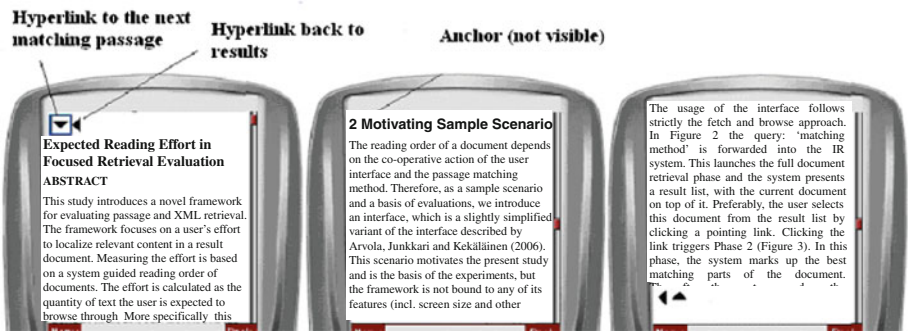

Fig. 4 Matching chain

When a user has read the retrieved passages and is still willing to read more within the document, we have to consider options of how the user proceeds after seeing the last retrieved passage. The extreme options for further reading are that the remaining relevant text is read immediately after the last retrieved passage (best case), or after all other non-relevant text is read (worst case). The best case can be discarded, since it is too easy to deliver good results with that. Instead, we define a third case in between (natural case),

where the reader clicks the hyperlink back to the beginning of the document and reads the remaining parts in document order.

The scenario affects the reading order but it does not tell us how long the user is willing to browse the document. We assume that the browsing continues until the user's *tolerance to irrelevance* (de Vries et al. 2004) has been reached. At that cut-off point the reader is assumed to be bored with the irrelevant material and moves onto the next document in the result list.

Consequently, evaluating the results delivered by a retrieval system can be based on this scenario. In other words, the user is expected to follow the matching chain, i.e. read the matching passages in document order. Whether the interface is easy to use, or whether every user is willing to utilize the features in this scenario are important usability issues but not a concern of the present study. In Sect. 5 we illustrate, with the metrics described in Sect. 4, how different retrieval systems perform within this sample scenario and how systems perform without the anchor hyperlink structure (i.e. document retrieval). In other words, what is the improvement rate, when using passage retrieval and the presented user interface compared with traditional document retrieval?

3 Related work

The present study combines measuring passage/XML retrieval and the concept of expected user effort. Accordingly, in Sect. 3.1 we review the current metrics evaluating fetch and browse style XML/passage retrieval and in Sect. 3.2 we address existing approaches in measuring the user effort at the retrieval.

3.1 Passage retrieval and relevant-in-context in INEX

The evaluation framework presented in this study is focused on a reading order within a single document and the approach is close to passage or XML retrieval and the work done within this context. Therefore the outcome of the present study relates to INEX (Initiative of the Evaluation of XML, INEX 2009) and the evaluation testbed provided by it. INEX is a prominent forum for the evaluation of XML retrieval offering a test collection with topics and corresponding relevance assessments, as well as various evaluation metrics. Currently, aside evaluating element retrieval, passage retrieval evaluation is also supported in INEX. That is because the relevance assessments are executed so that the assessors have marked up the relevant passages regardless of any element boundaries (Piwowarski and Lalmas 2004). Similar relevance assessments are available also in TREC Hard Track's passage retrieval (Allan 2004). In terms of the present study, a recall base for a document consists of a set of character positions.

The evaluation of the fetch and browse approach is an essential issue in content-oriented XML retrieval. This is mainly because the Relevant-in-Context (RiC) task of the INEX's tasks is considered the most credible from the users' perspective (Trotman et al. 2007; Tombros et al. 2005). The task corresponds fully to the fetch and browse approach. Because of the complex nature of the task, the evaluation measures are constantly evolving. There has also been a concern that full document retrieval would be very competitive in XML retrieval (Kamps et al. 2008a). According to the fetch and browse approach, in the official metric of the RiC task, separate scores are calculated for each individual retrieved document d as a *document score* ($S(d)$) in the browse part, and the

document result list as a *list score* in the fetch part. Next we introduce the current official metric for RiC in detail.

3.1.1 List score

The list score is calculated over a ranked list of documents based on document scores. A generalized precision is calculated as the sum of document scores up to an article-rank divided by the article-rank. Similarly generalized recall is the number of relevant articles retrieved up to an article rank, divided by the total number of relevant articles. Formally generalized precision (gP) at rank r is defined as follows:

$$gP[r] = \frac{\sum_{i=1}^{r} S(d_i)}{r},$$

and similarly the generalized recall:

$$gR[r] = \frac{\sum_{i=1}^{r} isrel(d_i)}{Trel},$$

where *Trel* denotes the total number of relevant documents and *isrel* is a binary function of the relevance at a given point. With these equations, we are able to calculate the average generalized precision for the result list:

$$AgP = \frac{\sum_{r=1}^{D} (isrel(d_r) \times gP[r])}{Trel},$$

where D is the ranked list of documents. Mean average generalized precision ($MAgP$) is calculated basically as the mean of the values of individual topics. Further details can be found in (Kekäläinen and Järvelin 2002; Kamps et al. 2007, 2008b, c).

The list score is general in a sense that the calculation of document score ($S(d)$) is not predefined, except that the values range is [0,1]. We adopt the list score for our evaluation metrics and replace later the document score with our own formula. Next, we introduce the official document measure used in INEX.

3.1.2 Document score in INEX

The official INEX measure for the document score is an *F-Score* of the retrieved set of character positions (Kamps et al. 2008b; see also Allan 2004). The *F-Score* is calculated for each retrieved document d as follows (Kamps et al. 2008a):

$$F_\alpha(d) = \frac{(1+\alpha^2) \times P(d) \times R(d)}{\alpha^2 \times P(d) + R(d)},$$

The α value is used to tune the role of the precision in the formula. It determines the power of which the precision is taken into account in the evaluations. $P(d)$ (the document precision) is the number of retrieved relevant characters divided by the number of retrieved characters. $R(d)$ (the document recall), accordingly, is the number of characters assessed to be relevant that is retrieved divided by the total number of relevant characters as follows:

$$P(d) = \frac{|rel(d) \cap ret(d)|}{|ret(d)|}$$

$$R(d) = \frac{|rel(d) \cap ret(d)|}{|rel(d)|}$$

In other words, the retrieval performance of a system is based solely on the set of character positions within the retrieved passages, whereas our approach takes the reading order and the tolerance of irrelevance into account as well.

The aim of using the *F-Score* of retrieved passages is to measure effectiveness in the relevant in context task, where "focused retrieval answers are grouped per document, in their original document order, providing access through further navigational means" (Kamps et al. 2008b). However, since the official *F-Score* measure treats the retrieved character positions as a set, the reading order dimension remains unjustified. With the framework of this study we take the document order (i.e. reading order) into account. Based on the framework we present novel measures as alternatives to the *F-Score* of retrieved passages in Sect. 4.

Since the final score of a system is a combination of document and list scores, we denote the combined measure as *list_score\document_score*. For example the official INEX measure, mean average generalized precision list score over $F_{0.25}$ document scores is denoted as $MAgP\backslash F_{0.25}$.

3.2 User effort in evaluation metrics

Most evaluation metrics of IR effectiveness are based on topical relevance (Saracevic 1996) and include explicit or implicit user models. Besides relevance, evaluation metrics have tried to encompass other aspects of user behavior affecting information retrieval, most prominently satisfaction and effort. Next we review metrics related to our study.

The implicit user model of the traditional (laboratory) full document retrieval evaluation assumes that the user reads the documents of the result list one by one starting from the beginning, and stopping when the last relevant document is passed. This might not be a realistic assumption of the user behavior, but has been considered adequate for evaluation purposes for decades. A further elaboration (Robertson 2008) interprets *average precision* with an assumption that users stop at a relevant document in the ranked list after their information need is satisfied. If stopping is uniformly distributed across relevant documents, average precision may be interpreted according to this simple user model.

Expected search length (ESL, Cooper 1968) takes the expected user effort into account as the average number of documents the user has to browse in order to retrieve a given number of relevant documents. ESL has inspired other metrics, like *expected search duration* (Dunlop 1997) and *tolerance to irrelevance* (T2I, de Vries et al. 2004). Instead of search length, expected search duration measures the time that users need to view documents in the ranked result list to find the required number of relevant documents. Predicted user effort is incorporated with the interface and search engine effects into one evaluation model.

A user's tolerance to irrelevant information is a central notion in T2I. It is aimed at retrieval environments without predefined retrieval units; in other words, passages of documents (or other information storage units) are retrieved instead of whole documents. The user model of T2I assumes that a retrieved passage acts as an entry point to the document: if the user does not find relevant information in the document before his or her tolerance to irrelevance is reached, he or she will move to the next item in the result list. de Vries et al. (2004) combine user effort with the model, and propose measuring it as time spent on inspecting irrelevant information. Moreover, they mention that in XML IR words or sentences could be used as well. As actual evaluation measures the authors propose a T2I variant of average precision of document cut-off values, i.e. average over precisions after

given T2I points in time, and an ESL variant, i.e. "the user effort wasted while inspecting the system's result list … augmented with the effort needed to find the remaining relevant items by random search through the collection" (de Vries et al. 2004, 470).

Other metrics combining user effort with retrieval evaluation are *expected precision-recall with user modelling* (EPRUM, Piwowarski and Dupret 2006; Piwowarski 2006) and *effort precision—generalized recall* (EP/GR, Kazai and Lalmas 2006). EPRUM considers returned result items as entry points to the collection from which points the user can navigate to relevant items. EP/GR measures the amount of effort as the number of visited ranks that the user has to spend when browsing a system's ranked result list, compared to the effort an ideal ranking would take in order to reach a given level of gain.

Like T2I our framework can be applied not only to XML elements but also to arbitrary passages. Our framework shares the notion of effort with the earlier measures; however we interpret effort at character level but in principle share the idea of measuring effort as the amount of text. Further, the best retrieved passage acts as an entry point to the document like in T2I and EPRUM. We also utilize tolerance to irrelevance as a stopping rule within a document. However, our framework exploits the guided reading order of the retrieved document and scoring of each document is based on this order. Consequently measuring effectiveness at document level differs from other measures.

In our framework, the reading order is considered within a document, not in a hierarchical element structure. The idea behind the browsing model is somewhat different than in EPRUM (and also Ali et al. 2008), where the browsing is based on hierarchical and linked items. The proposed framework differs from the earlier work by combining the character level evaluation to the system guided reading order.

4 Metrics based on expected browsing effort

The character level evaluation can be associated with the traditional document retrieval evaluation in a sense that the retrievable unit is a character and is treated as if it was a document. When considering the result as a set of retrieved character positions, document's precision and recall and document's *F-Score* correspond to the full match evaluation measures. Our approach, however, resembles partial match evaluation on the document level. Instead of treating the result as a set of character positions, we treat the result as a list of character positions. The order of the characters in the list depends on their browsing order. Clearly, treating the retrievable units as a list instead of a set broadens the number of alternatives for the retrieval performance measures. In addition, the list approach brings on the temporal dimension in browsing, and thus enables the exploitation of the T2I approach.

We present two metrics based on the reading order: *character precision-recall* (*ChPR*) and *cumulated effort* (*CE*). For both metrics we assume that some text within the retrieved relevant document is assessed to be relevant. In other words there exists a recall base, like the INEX recall base, containing the character positions of relevant characters. These characters are then compared with the expected order of reading. The metrics follow the underlying evaluation framework and the reading order is not bound to any specific user or interface scenario.

In *ChPR* the relevance score values scale between 0 and 1, and the list score is calculated analogously to generalized precision-recall, whereas in *CE* the document scoring is looser and the list score is calculated by cumulated effort, which has evolved from the cumulated gain metric (Järvelin and Kekäläinen 2002).

4.1 Character precision-recall

In our framework characters are units to retrieve and they are expected to be read in some order. This simple reading model along with the character position wise relevance assessments enables the usage of the standard precision-recall metric for the document score, and thus all the related measures are available. The list score, in turn, is calculated by the generalized precision-recall metric as given in Sect. 3 (Kamps et al. 2007).

As a basic measure of this metric, we define the character average precision for a document, *aveChP*(*d*), which is similar to the traditional average precision measure. The only difference is that documents are replaced here with characters.

$$aveChP(d) = \frac{\sum_{p=1}^{|d|} (P_d(p) \times RL_d(p))}{NRC_d}$$

In the formula, p is the character position from the point the reading starts, *RL* a binary valued function on the relevance of a given position, *NRC* the number of relevant characters in document d, and P precision at a given position in d. We set *aveChP* as the document score for calculating the list score with the generalized precision-recall metric (see Sect. 3.1.1). Note that *aveChP* is calculated for a relevant document only. For non-relevant documents *aveChP*, and other measures within the character precision-recall metric the document score is 0.

The *aveChP* can be considered a somewhat system-oriented measure, since it does not take a stand on when the user stops reading the document. However, it rewards systems that organize the expected reading order in an optimal way. Naturally, instead of using *aveChP* a number of cut-off measures can be used, for instance precision can be calculated when a chunk of 600 characters is read (i.e. *ChPR*@600). Apart from this kind of basic cut-off point, a user oriented cut-off point, like T2I, can be utilized. In T2I the reading of a document is supposed to end when a pre-set tolerance to irrelevance has been reached (or the whole document is read through). For instance the T2I measure *T2Iprec* (2000) means that the reading ends, when the user has seen 2000 non-relevant characters, and then document's precision is calculated. In other words the T2I is a cut-off measure, where the cut-off point varies according to the read irrelevant material. In addition to precision, also recall (*T2Irecall*) and their harmonic mean *F-Score* ($T2If_\alpha$) are viable measures to combine with T2I. Note that this time the *F-Score* is calculated according to the read characters, not to the retrieved.

A couple of toy examples illustrate *aveChP*; we calculate some sample values for a 'mini document'. In the example the reading order is based on the scenario given in Sect. 2 and the natural case reading order for the non-retrieved passages. In Table 1, there is a character position list for a sample mini document.

For each example the characters assessed as relevant are in bold face and the retrieved characters are underlined. The two examples are the following:

Example 1: "**relevant content is in bold** and retrieved is underlined"
Example 2: "**relevant content is in bold** and retrieved is underlined"

Example 1: The system has found a relevant document (value as such), but is unable to identify the relevant content within the document. The expected reading order is ⟨33, …, 55, 1, 2, …, 31, 32⟩ and the recall base is the set {1, 2, …, 27}. Thus $aveChP = 0.35$. The *F-Score* (of the retrieved characters, $\alpha = 1$) does not give any value to this result, and thus the document corresponds to a non-relevant document: i.e. $F\text{-}Score = 0$. However, in this case the passage retrieval system is not helpful, because the relevant content is at the

Table 1 Character position list of a mini document (line break is nr. 28)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
r	e	l	e	v	a	n	t		c	o	n	t	e	n	t		i	s		i	n		b	o	l	d
29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55
a	n	d		r	e	t	r	i	e	v	e	d		i	s		u	n	d	e	r	l	i	n	e	d

beginning of the document and no guidance is needed. Thus, a system retrieving full documents would deliver the following scores: $aveChP = 1$, $F\text{-}Score = 0.66$.

Example 2: The passage retrieval system does not identify all relevant content, but the retrieved and relevant passages are partially overlapping. This example illustrates the early precision biased nature of the *aveChP* measure. Here, *F-Score* is 0.16. The reading order is: ⟨24, 25, …, 44, 45, 1, 2,…, 23, 46, …, 55⟩, and $aveChP = 0.53$.

Comparing our reading order based approach to the *F-Score* shows the benefit of combining the amount of read text and reading order in evaluations. For example *F-Score*, would give the same score to a long document, with a relevant latter half, as with a relevant first half, even though it requires more effort to reach the latter half, assuming that the browsing starts at the beginning of the matching passage, in this case the whole document.

In addition, the *F-Score* calculation involves a hidden assumption that the user stops reading the document after the retrieved passages. This holds even if there were still relevant passages elsewhere in the document to be read. Thus, the browse phase for reaching more relevant content is incomplete, and the passages of the next document in the result list are prioritized over the ones in the document the user is currently looking at. These passages will then never be reached. This seems a rather simple user model.

4.2 Cumulated effort

Instead of the gain the user receives by reading the documents in the result list, cumulated effort (*CE*) focuses on the effort the user has to spend while looking for relevant content. The effort-oriented metric should fulfill the following aims: (1) to model the increase of the expected effort, when the user is reading the document list further; (2) to ensure that minimal effort produces no increase to the effort value; (3) to allow different effort scales.

4.2.1 Document score

For calculating *CE*, an effort score for each ranked document d, $ES(d)$, is needed. The values of $ES(d)$ should increase with the effort; in other words the lower the score the better. There are different possibilities for assigning effort scores for documents. Next, we propose a solution motivated by the evaluation framework.

We assume that the system's task is to point out that the retrieved document is relevant by guiding the user to relevant content. As soon as the user's attention is focused on the relevant spot, the systems mission is accomplished. The document score represents how much expected effort it takes to find relevant text within the document. The scoring depends directly on (non-relevant) characters read before finding the first relevant passage or element. For that we define:

- d' is the expected reading order of document d
- $r_{d'}$ is the position of the first relevant character with the reading order d'

- function $LE(r_{d'})$ gives the localizing effort score based on a chosen window size.

The document effort score $ES(d)$ is the score, that the LE function gives after the relevant text within the document is yielded. For non-relevant documents we assume a default effort score NR:

$$ES(d) = \begin{cases} LE(r_{d'}), & \text{if } d \text{ is relevant} \\ NR, & \text{otherwise} \end{cases}$$

We do not give any default implementation for the LE function. Instead, we introduce sample quantizations in Sect. 5, motivated by the small screen scenario of Sect. 2.

4.2.2 List score

After having defined effort scores for documents in a ranked result list, we can cumulate the effort over the list up to a given cut-off point. Cumulated effort (vector CE) is defined as follows:

$$CE[i] = \sum_{j=1}^{i} \frac{ES(d_j)}{\min ES} - 1$$

where i is a position in the result list and $minES$ denotes the absolute minimum value the function ES delivers. This is obtained when the relevant material is found immediately. The formula ensures that when the effort is minimal the effort value does not increase (cumulate). For instance, let us consider a result list of documents $\langle d_1, d_2, d_3, d_4, d_5 \rangle$ with a vector of corresponding scores $\langle ES(d_1), ES(d_2), ES(d_3), ES(d_4), ES(d_5) \rangle = \langle 1, 2, 5, 1, 5 \rangle$. Moreover, let us assume that the range set of $LE(r_{d'})$ is $\{1, 2, 3, 4\}$ and $NR = 5$, then $minES = 1$. Now $CE = \langle 0, 1, 5, 5, 9 \rangle$.

Normalized cumulated effort (vector NCE) is needed for averaging over multiple topics. It is defined as follows:

$$NCE[i] = \sum_{j=1}^{i} \frac{ES(d_j)}{IE[j]} - 1$$

where IE is the vector representing the ideal performance for the topic. As an example we take the values from the previous example and in addition we state that total number of relevant documents is three, i.e. $Trel = 3$, thus $IE = \langle 1, 1, 1, 5, 5, \ldots \rangle$ and $NCE = \langle 0, 1, 5, 4.2, 4.2, \ldots \rangle$. A normalized optimal run produces a curve having zero values only.

Often it is necessary to have one effectiveness value for the whole result list or a run. An average at a given cut-off point for normalized cumulated effort is calculated as follows:

$$ANCE[i] = \frac{\sum_{j=1}^{i} NCE[j]}{i}$$

where i is the cut-off point. Analogously to mean average precision, mean average normalized cumulated effort ($MANCE[i]$) may be calculated over a set of topics. It is worth noting, that the curves presenting cumulated effort represent the better effectiveness the closer they are to x-axis.

5 Experiments

Next, we illustrate the use of the CE and ChP metrics in testing runs from the RiC task of the INEX 2008 ad hoc track. The RiC task contains 70 topics with character-wise

relevance assessments, and the test collection covers around 660,000 XML marked articles in the English Wikipedia collection (Denoyer and Gallinari 2006). The official results were measured with *F-Score* having alpha value 0.25 (INEX 2009; Kamps et al. 2008c).

Aside from presenting sample results of our metrics and comparing these with the $F_{0.25}$-*Score* metric we aim to study the benefit of using passage retrieval for more effective browsing within the retrieved documents. This is done by comparing the focused fetch and browse strategy with plain full document retrieval. In the document retrieval baseline, the reading starts from the beginning of the document and continues until a relevant passage is met in the *CE* metric, and all relevant passages are read consecutively in *ChPR* metric. This baseline is compared with the corresponding element run.

In Sect. 5.3, based on "Appendix 2", we give a comparative summary of 38 official INEX 2008 runs. First, as a special focus, we report the results of three best performing participants of the RiC task, namely *GPX1CORICe* from the University of Queensland (in Kamps et al. 2008c) and *RICBest* from the University of Waterloo (Itakura and Clarke 2009). For comparison, we selected the best performing full-document run of the task: *manualQEIndri* from the University of Lyon (Ibekwe-SanJuan and SanJuan 2009). Further, we constructed additional runs by transforming GPX1CORICe and RICBest so that the browse phase was discarded, i.e. full documents were returned instead of sets of passages. These runs are labelled as *GPX1CORICe_doc* and *RICBest_doc*.

5.1 Results with character precision-recall

For the Character Precision Recall metric we report results obtained with the following measures: *aveChP* and two T2I based measures, namely $T2If_1(300)$ and $T2If_1(2000)$, where the tolerance of irrelevance is 300 and 2000 characters respectively. In case of the T2I measures the document score is calculated with *F-Score* (note that this is different from *F-Score* of retrieved passages). The α value is 1. In all measures we assume the natural reading order after retrieved passages, i.e. the reading continues from the beginning of the retrieved document after reading the retrieved passages. The *gPr* (list score) curves for the runs GPX1CORICe, RICBest, GPX1CORICe_doc, RICBest_doc and manualQEIndri are shown in Figs. 6, 7 and 8. For comparison, these runs measured with $F_{0.25}$-*Score* of retrieved passages (i.e. the official INEX metric) are shown in Fig. 5. The related *MAgP* values can be found in "Appendix 2".

In Figs. 5, 6, 7 and 8 the superiority of manualQEIndri at early ranks is obvious. It outperforms the element runs, but the comparisons of manualQEIndri with the full document runs (GPX1CORICe_doc and RICBest_doc) show that its better performance is more due to the good document ranking than to full document retrieval competitiveness in focused retrieval (see Kamps et al. 2008a). Adopting focused retrieval clearly gives a boost for RicBest and GPXCORICe in comparison to their full document baselines. This comes especially evident when assuming a lower tolerance to irrelevance, where with the $T2If_1(300)$ measure, the element runs ($MAgP\backslash T2If_1(300)$ 0.187, 0.163, resp.) beat the manualQEIndri (0.151) in addition to their document baselines (0.136, 0.133, resp.). Note that the figures show cut-off results. The differences between document and corresponding element runs by all $MAgP\backslash ChP$ measures are statistically significant ($p < 0.001$, t-test).

5.2 Results with cumulated effort

Measuring the effort on finding relevant content is done with the localizing effort metric for the document score and cumulated effort for the list score. As a basis for calculating the

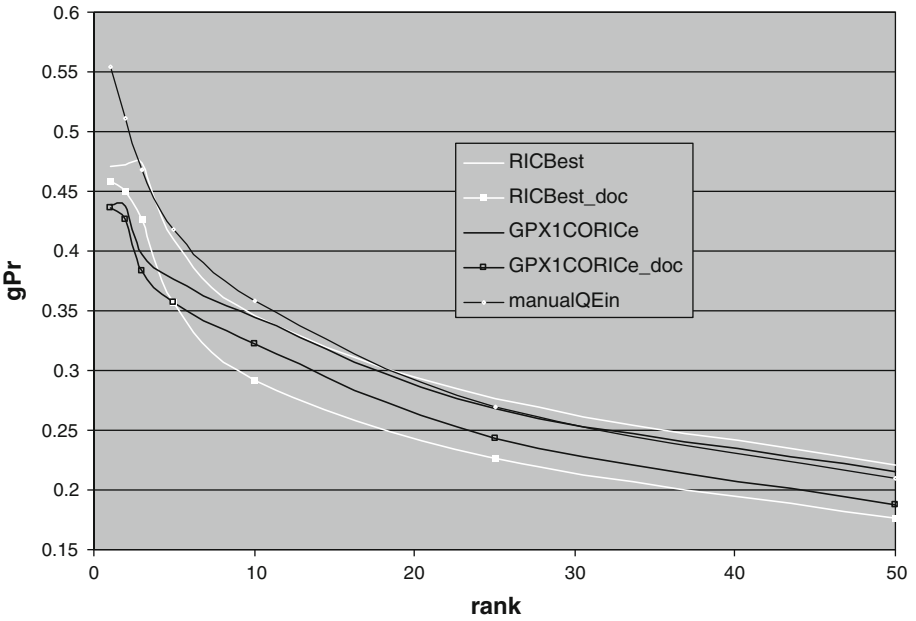

Fig. 5 Generalized precision at cut-off points ($gPr[i]/F_{0.25}$), where the document score is measured with the $F_{0.25}$-*Score*

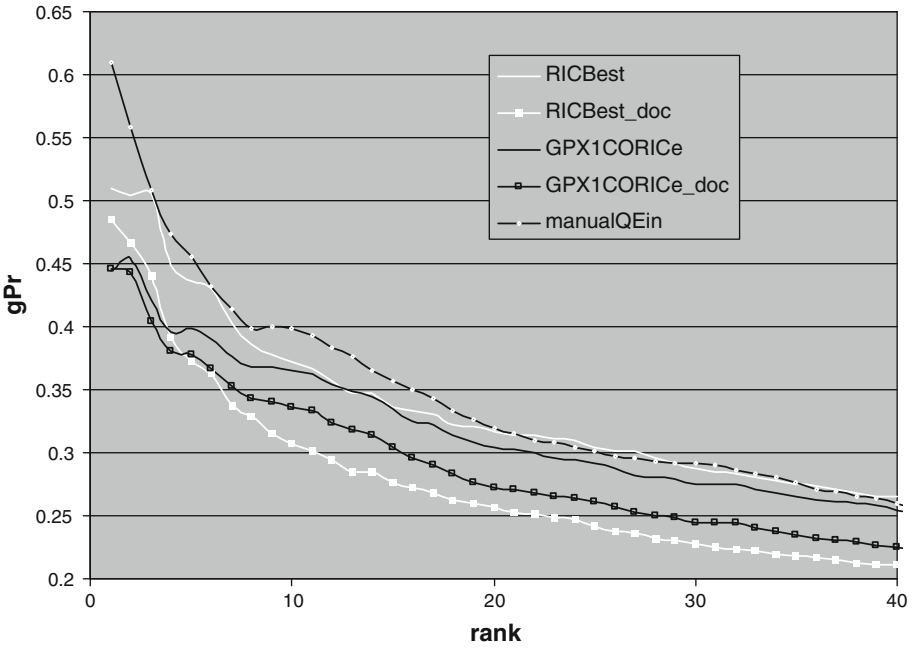

Fig. 6 Generalized precision at cut-off points ($gPr[i]/AveChP$). The document score is measured with the average character precision with the natural case reading order

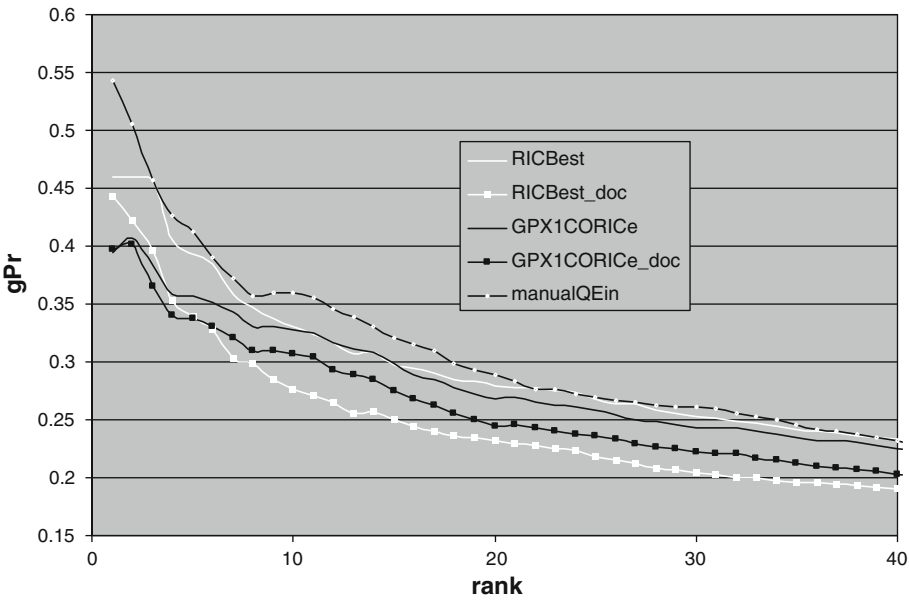

Fig. 7 Generalized precision at cut-off points ($gPr[i]/T2If_1(300)$). The document score is measured with the T2I *F-Score* 300 with the natural case reading order

localizing effort we bind the scoring to the screen size. As scoring for an individual document, we set:

$$LE(i) = \begin{cases} 1, & \text{if } i \leq sSize \\ 2, & \text{if } sSize < i \leq sSize \times 2 \\ 3, & \text{if } sSize \times 2 < i \leq sSize \times 3 \\ 4, & \text{otherwise} \end{cases}$$

$$NR = 5$$

where *sSize* denotes the screen size in characters. For the screen size, we experiment with two distinct values: 300 for a mobile screen and 2000 for a laptop screen. The results are labelled as *screen 300* shown in Fig. 9 and *screen 2000* shown in Fig. 10, respectively. The *MANCE\LE* score of each run is in "Appendix 2". The differences between RicBest and RicBest_doc, as well as GPXCORICe and GPXCORICe_doc are statistically significant ($p < 0.001$, *t*-test) measured with *MANCE\LE*.

The results verify that in comparison to full document retrieval, using a more focused strategy brings somewhat down the effort in localizing the relevant content. Not surprisingly this feature is stressed when using a smaller screen.

5.3 Comparative analysis of the metrics

In addition to comparing top runs, we calculated results for 38 INEX 2008 submissions. In Table 2 Kendall τ correlations of different measures are given. The correlations are based on the results of "Appendix 2". The $F_{0.25}$-*Score* and *ChP* results are calculated with *MAgP* and others with the *MANCE\LE* measure at list cut-off 600. For simplicity the correlations between *MANCE* and *MAgP* are reported as their opposite values, because the score

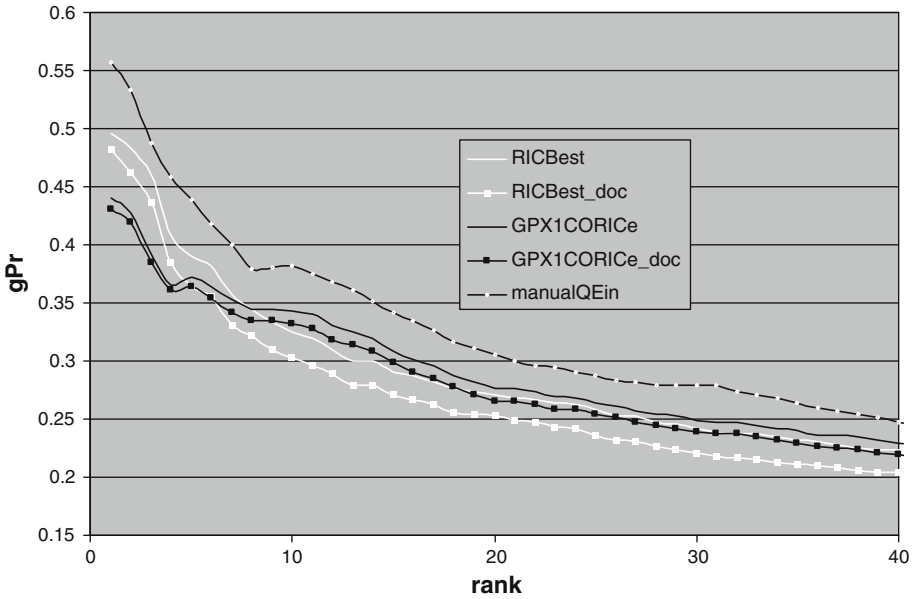

Fig. 8 Generalized precision at cut-off points ($gPr[i]/T2If_1(2000)$). The document score is measured with the T2I *F-Score* 2000 with the natural case reading order

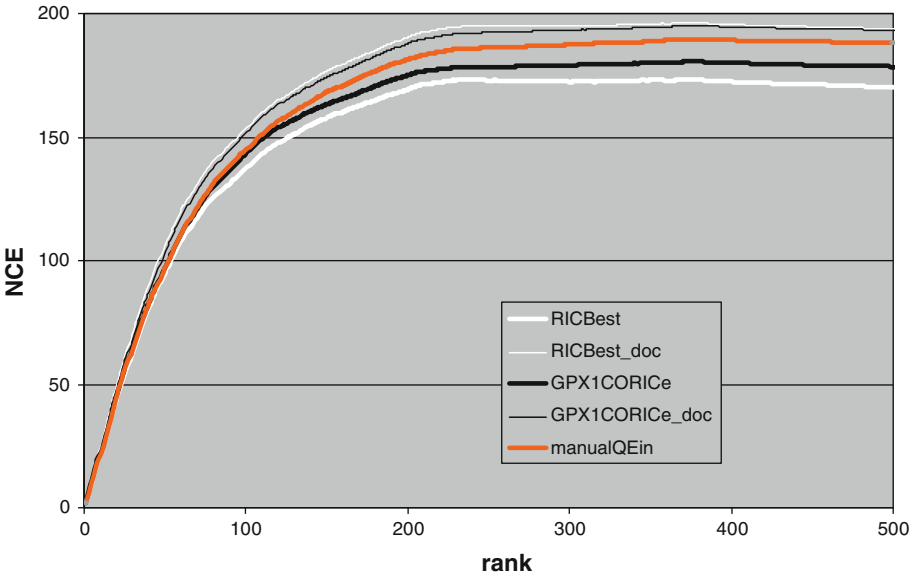

Fig. 9 Normalized cumulated effort with small screen interpretation (screen 300). NB. The lower the curve the less effort spent

interpretations are inverse. In the tables the *doc* ending refers to the document retrieval baseline. For example $F_{0.25}$-*Score doc* means that the runs are handled as if they were full document runs instead of element/passage runs. The correlation between a measure and its counterpart to full document evaluation is in bold.

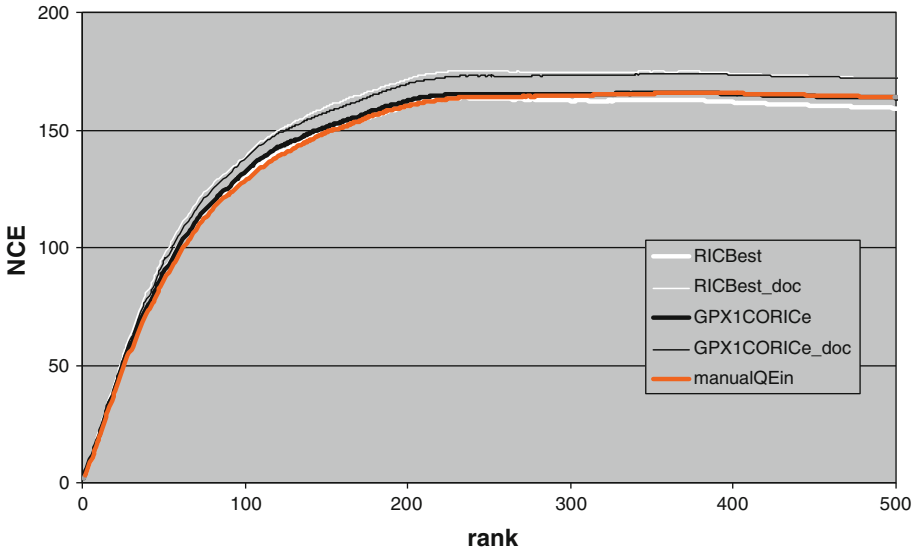

Fig. 10 Normalized cumulated effort with large screen interpretation (screen 2000) NB. The lower the curve the less effort spent

When comparing element/passage runs with their full document baseline, 19 out of 25 runs gain some improvement measured with the $AgP\backslash F_{0.25}$. With Cumulated Effort (for both screen sizes) all runs benefit from the more focused fetch and browse strategy. With *AgP\aveChP* and the reported *AgP\T2I* measures the numbers of benefiting runs are 21 and 15, respectively. The competitiveness of a full document run varies from measure to measure. For instance, the best performing such run, manualQEIndri, is third measured with $MAgP\backslash F_{0.25}$, ninth with the screen size 300 (*MANCE\LE*) and second with the screen size 2000 (*MANCE\LE*). With the *ChP* metric the *MAgP\aveChP* measure delivers third place for the run and with $MAgP\backslash T2If_1(300)$ the ranking is as low as tenth. However, while $T2If_1(300)$ is a rather early cut-off measure (at document level) it might be less reliable, as are the early cut-off measures in general in traditional document retrieval.

6 Discussion

The fetch and browse approach highlights the best matching passages in their context. The aim of this kind of passage retrieval is to make document browsing more effective. In other words the reading order of the retrieved document changes so that the new order is more convenient for the user in comparison to full document retrieval and sequential reading. Thus, successful passage retrieval reduces user effort in finding the best matching parts of the document. In the presented framework the effort is measured with the amount of text the user is supposed to read.

Quite recently the character level of text has been taken into account in the evaluations in the INEX initiative. However, the related *F-Score* metric is system-oriented, and the performance figures are calculated based on the sets of character positions. The set-oriented mindset does not take the reading order into account, which was one of the initial motivations of the fetch and browse style retrieval.

Table 2 Kendall τ correlation of official INEX 2008 results of 38 runs

	MAgP\ $F_{0.25}$ Foc.	*MAgP*\ $F_{0.25}$ Doc.	*MANCE**LE* Screen = 300 Foc.	*MANCE**LE* Screen = 300 Doc.	*MANCE**LE* Screen = 2000 Foc.	*MANCE**LE* Screen = 2000 Doc.	*MAgP*\ *aveChP* Foc.	*MAgP*\ *aveChP* Doc.	*MAgP*\ *T2If*$_1$(300) Foc.	*MAgP*\ *T2If*$_1$(300) Doc.	*MAgP*\ *T2If*$_1$(2000) Foc.
*MAgP**F0.25* Doc	**0.826**										
*MANCE**LE* Screen = 300 Foc.	0.478	0.378									
*MANCE**LE* Screen = 300 Doc.	0.731	0.762	**0.537**								
*MANCE**LE* Screen = 2000 Foc.	0.740	0.629	0.709	0.737							
*MANCE**LE* Screen = 2000 Doc.	0.754	0.768	0.525	0.897	**0.799**						
*MAgP**aveChP* Foc.	0.894	0.789	0.566	0.739	0.817	0.779					
*MAgP**aveChP* Doc.	0.815	0.966	0.372	0.751	0.623	0.762	**0.789**				
*MAgP**T2If*$_1$(300) Foc.	0.695	0.623	0.720	0.697	0.789	0.702	0.795	0.623			
*MAgP**T2If*$_1$(300) Doc.	0.717	0.851	0.408	0.774	0.585	0.711	0.725	0.857	**0.648**		
*MAgP**T2If*$_1$(2000) Foc.	0.837	0.805	0.560	0.776	0.771	0.770	0.897	0.800	0.811	0.773	
*MAgP**T2If*$_1$(2000) Doc.	0.777	0.928	0.405	0.783	0.611	0.749	0.757	0.916	0.628	0.923	**0.805**

We introduced two metrics based on our framework: Character precision-recall (*ChPR*) is based on traditional precision-recall metric. It takes all read text into account also after the first relevant spot and even after the last retrieved passage, if necessary. Within the metric two measures are introduced. Average character precision (*AveChP*) is considered more system-oriented and rewards systems, which are able to present the whole relevant content early to the user. *T2I* based measures takes the user's *tolerance to irrelevance* into account. These measures are based on the total amount of non-relevant characters the user is willing to read per document. Unsurprisingly, the more tolerance to irrelevance the user has the less benefit XML/passage retrieval systems bring.

The cumulated effort metric (*CE*) is a general purpose list measure in a sense that the document scores can be calculated in different ways. In this study the document level measure is localizing effort (*LE*), which measures the effort the user has to take in order to localize the relevant content. In other words, it measures the effectiveness to assess the document to be relevant.

The fetch and browse retrieval is considered a special case of full document retrieval having a flavour of focused retrieval. Thus, good article ranking tends to deliver good results regardless of the metrics. However, the results with the novel metrics showed that the user effort is overall reduced when using passage or XML retrieval. This is illustrated with the pairwise comparisons of element/passage and the corresponding full document run. Thus, the present study gives a partial answer to the concern aroused within the INEX community that the full document retrieval is a competitive approach in fetch and browse style XML retrieval (Kamps et al. 2008a).

Since the experiments were carried out using the existing runs of INEX, any overfitting strategies for the metrics did not show up. As a remote example of returning only the query words within a document might lead to high early precisions at character level. Obviously, the *CE* metrics would deliver good results with that strategy. Clearly, reading a single word is not enough for a user to assess text passages relevance or even to understand it, but he or she has to read the surroundings as well. Therefore, one credible solution preventing this kind of overfitting to the metric would be to set a minimum effort score (penalty) for reading a retrieved passage in addition to the constant effort score reading a character.

Even though we focused on the evaluation of fetch and browse style retrieval, in future studies we will aim to extend this approach to concern other styles of XML and passage retrieval. For instance, instead of starting from the beginning, the browsing of a document may start from the best entry point provided by the IR system (Finesilver and Reid 2003; Reid et al. 2006). This applies to the Best in Context task of INEX. Further, elements can be retrieved as such, i.e. without context. Thus, a result list having elements or arbitrary passages only, like the focused or thorough tasks of INEX, can also be measured within the presented framework.

7 Conclusions

The study gives a framework for the evaluation of element/passage retrieval systems. Unlike the contemporary approaches, the framework is based on reading order, which depends on the co-operative action of a retrieval system and a guiding user interface. The study was motivated by a small screen scenario, where the text is presented as a single column and the default reading of a document is sequential representing the full document retrieval baseline. As the focused retrieval alternative we used a so called fetch and browse approach where effective access to the best matching passages was provided by hyperlinks, still maintaining the document order. Within the scenario we introduced two metrics:

character precision-recall and cumulated effort. In character precision-recall we made an assumption of the user's tolerance to irrelevance, i.e. the point in which the user moves onto the next document. The document score for cumulated effort is calculated with localizing effort function *LE*. In the evaluations we used *LE* functions based on window size motivated by a small screen scenario. However, *LE* can be replaced with other effort measures. We performed laboratory evaluations within the INEX test bed. The results showed that in comparison to traditional full document retrieval, with our measures, more focused element/passage retrieval shows increase in system performance. This gives a better motivation for a fetch and browse style focused retrieval in comparison to the official $F_{0.25}$-*Score* measure.

Acknowledgments The study was supported by Academy of Finland under grants #115480 and #130482.

Appendix 1

See Table 3.

Table 3 List of symbols used in the study

Symbols related to document scoring	
$F\alpha(d)$ *fα*	*F*-Score
$S(d)$	General document score (of document *d*)
$P(d)$	Document precision
$R(d)$	Document recall
$rel(d)$	The set of relevant character positions
$ret(d)$	The set of retrieved character positions
Contribution of this study	
ChPR	Character precision-recall metric
ChP@600	Character precision at cut-off 600
aveChP	Average character precision
$P(p)$	Character precision at position *p*
$RL(p)$	Binary relevance value function of character position *p*
NRC	Number of relevant characters
LE	Localizing effort
NR	Default value for a non-relevant document
$ES(d)$	Effort score (of document *d*)
minES	Absolute minimum of ES function
T2Isco(*300*)	Score (sco) when 300 non-relevant characters are read. (i.e. Tolerance to irrelevance)
Symbols related to list scoring	
gP	Generalized precision
gR	Generalized recall
AgP	Average generalized precision
Trel	Number of relevant documents
Contribution of this study	
CE	Cumulated effort metrics
NCE	Normalized cumulated effort
ANCE	Average normalized cumulated effort
MANCE	Mean average normalized cumulated effort

Appendix 2

See Table 4.

Table 4 Comparison of whole document and passage/element INEX 2008 runs

Run	*MAgP\F*$_{0.25}$			*MANCE\LE* Screen = 300			MANCE\LE Screen = 2000			*MAgP\aveChP*			*MAgP\T2If*$_1$(300)			*MAgP\T2If*$_1$(2000)		
	Doc.	Foc.	Change %	Doc.	Foc.	Change%	Doc.	Foc.	Change %	Doc.	Foc.	Change %	Doc.	Foc.	Change %	Doc.	Foc.	Change %
p78-RICBest	0.188	0.228	21.3	171	152	−12.8	154	143	−7.3	0.202	0.250	23.6	0.136	0.187	38.1	0.168	0.205	22.0
p78-RICArt	0.216	0.227	5.2	162	153	−5.7	148	144	−2.2	0.231	0.249	7.6	0.172	0.196	14.1	0.206	0.221	7.5
p5-GPX1CORICe	0.187	0.211	12.7	169	158	−7.5	152	145	−4.7	0.200	0.228	13.9	0.133	0.163	22.8	0.167	0.190	13.8
p5-GPX2CORICe	0.183	0.206	12.9	173	162	−7.2	156	149	−4.4	0.196	0.224	14.1	0.130	0.160	23.1	0.163	0.185	14.1
p10-TOPXCOallA	0.168	0.195	15.9	171	157	−8.8	153	145	−5.7	0.181	0.216	19.5	0.122	0.154	26.5	0.151	0.176	16.2
p5-GPX3COSRIC	0.168	0.189	12.1	175	164	−6.2	158	152	−3.9	0.180	0.206	14.4	0.117	0.145	23.4	0.150	0.171	13.6
p6-inex08artB	0.158	0.176	11.2	179	168	−7.1	162	156	−4.1	0.173	0.196	13.6	0.112	0.137	23.3	0.141	0.162	14.2
p6-inex08artB	0.165	0.175	6.5	178	169	−5.3	161	157	−2.9	0.180	0.196	8.9	0.118	0.138	17.1	0.149	0.165	10.9
p6-inex08artB	0.164	0.174	6.1	178	170	−5.0	161	157	−2.7	0.180	0.195	8.4	0.118	0.137	16.3	0.148	0.164	10.5
p72-UMDRic2	0.166	0.172	3.6	179	163	−10.3	165	156	−5.7	0.182	0.197	8.2	0.128	0.150	17.5	0.155	0.172	11.1
p6-inex08artB	0.162	0.170	5.1	177	163	−8.4	160	153	−4.3	0.177	0.195	9.9	0.115	0.142	23.6	0.145	0.164	12.7
p6-inex08artB	0.164	0.170	3.5	178	167	−6.6	161	157	−3.1	0.179	0.193	7.3	0.118	0.141	19.4	0.148	0.164	10.8
p6-inex08artB	0.158	0.167	5.9	178	164	−8.6	161	154	−4.6	0.172	0.190	10.3	0.112	0.139	24.1	0.141	0.160	13.0
p72-UMDRic	0.162	0.167	3.3	181	168	−7.5	166	159	−4.7	0.177	0.184	3.9	0.123	0.134	8.4	0.150	0.161	7.1
p12-p8u3exp51	0.135	0.158	17.4	183	166	−10.4	165	157	−5.2	0.148	0.182	23.3	0.098	0.133	35.7	0.122	0.141	15.2
p12-p8u3exp501	0.135	0.158	17.4	183	166	−10.4	165	157	−5.2	0.148	0.182	23.3	0.098	0.133	35.7	0.122	0.138	13.0
p12-p8u3exp311	0.130	0.152	17.0	184	166	−11.0	167	158	−5.3	0.143	0.178	24.2	0.096	0.132	37.8	0.119	0.138	15.7
p48-LIGMLRIC4O	0.154	0.150	−2.5	175	166	−5.4	163	160	−2.1	0.166	0.169	1.9	0.125	0.137	9.7	0.149	0.157	5.7

Table 4 continued

Run	*MAgP\F*$_{0.25}$			*MANCE\LE* Screen = 300			MANCE\LE Screen = 2000			*MAgP\aveChP*			*MAgP\T2If*$_1$(300)			*MAgP\T2If*$_1$(2000)		
	Doc.	Foc.	Change %	Doc.	Foc.	Change%	Doc.	Foc.	Change %	Doc.	Foc.	Change %	Doc.	Foc.	Change %	Doc.	Foc.	Change %
p4-BEIGBEDERI	0.157	0.149	−5.0	177	153	−15.4	158	149	−6.4	0.171	0.206	20.5	0.119	0.165	39.4	0.146	0.167	14.3
p78-RICSum1	0.127	0.141	10.9	184	168	−9.4	171	161	−6.4	0.138	0.159	15.0	0.095	0.119	25.5	0.113	0.134	18.4
p4-BEIGBEDERO	0.106	0.107	0.8	199	181	−9.7	185	176	−4.6	0.115	0.141	22.9	0.078	0.111	41.1	0.097	0.113	16.1
p48-LIGVSMRIC4	0.102	0.096	−5.7	184	181	−1.7	179	177	−0.8	0.110	0.111	0.6	0.093	0.097	4.1	0.102	0.104	2.8
p16-007RunofUn	0.006	0.005	−19.7	293	285	−2.6	288	281	−2.5	0.007	0.006	−13.3	0.006	0.004	−25.6	0.006	0.006	−5.2
p16-009RunofUn	0.005	0.003	−38.5	295	287	−2.8	292	284	−2.8	0.006	0.005	−15.5	0.004	0.003	−26.4	0.005	0.004	−23.4
p16-008RunofUn	0.004	0.003	−36.5	293	285	−2.8	289	281	−2.9	0.006	0.005	−19.3	0.004	0.003	−27.6	0.005	0.004	−19.0
Full Document Runs																		
p92-manualQEin	0.211			164			144			0.232			0.151			0.188		
p4-WHOLEDOC	0.193			170			154			0.207			0.151			0.187		
p4-WHOLEDOCPA	0.193			170			154			0.207			0.151			0.187		
p5-GPX1CORICp	0.190			169			152			0.204			0.135			0.170		
p5-GPX2CORICp	0.187			173			156			0.200			0.133			0.166		
p92-manualweig	0.184			179			163			0.205			0.133			0.165		
p92-autoindri0	0.171			177			158			0.188			0.120			0.153		
p92-manualindr	0.158			183			169			0.171			0.114			0.143		
p5-Terrier	0.152			187			167			0.165			0.103			0.133		
p56-VSMRIP05	0.150			182			164			0.164			0.109			0.133		
p92-manualweig	0.148			188			175			0.163			0.110			0.135		
p56-VSMRIP04	0.131			188			173			0.142			0.091			0.115		
p56-VSMRIP06	0.122			188			171			0.132			0.084			0.108		

References

Ali, M. S., Consens, M. P., Kazai, G., & Lalmas, M. (2008). Structural relevance: A common basis for the evaluation of structured document retrieval. In *Proceedings of CIKM '08* (pp. 1153–1162).

Allan, J. (2004). Hard track overview in TREC 2004: High accuracy retrieval from documents. In *Proceedings of the 13th text retrieval conference (TREC 2004)*. Nist Special Publication, SP 500-261, 11 pages.

Arvola, P., Junkkari, M., & Kekäläinen, J. (2006). Applying XML retrieval methods for result document navigation in small screen devices. *In Proceedings of MobileHCI workshop for ubiquitous information access* (pp. 6–10).

Buyukkokten, O., Garcia-Molina, H., Paepcke, A., & Winograd, T. (2000). Power browser: Efficient web browsing for PDAs. In *Proceedings of CHI '2000* (pp. 430–437).

Chiaramella, Y., Mulhem, P., & Fourel, F. (1996). A model for multimedia search information retrieval. *Technical report, basic research action FERMI 8134*.

Cooper, W. (1968). Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation, 19*(1), 30–41.

de Vries, A. P., Kazai, G., & Lalmas, M. (2004). Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *Proceedings of RIAO 2004* (pp. 463–473).

Denoyer, L., & Gallinari, P. (2006). The Wikipedia XML corpus. *SIGIR Forum, 40*(1), 64–69.

Dunlop, M. D. (1997). Time, relevance and interaction modelling for information retrieval. In *Proceedings of SIGIR '97* (pp. 206–212).

Finesilver K., & Reid J. (2003). User behaviour in the context of structured documents. In *Proceedings of ECIR 2003*, LNCS 2633 (pp. 104–119).

Hyönä, J., & Nurminen, A.-M. (2006). Do adult readers know how they read? Evidence from eye movement patterns and verbal reports. *British Journal of Psychology, 97*(1), 31–50.

Ibekwe-SanJuan, F., & SanJuan, E. (2009). Use of multiword terms and query expansion for interactive information retrieval. In *Advances in Focused Retrieval*, LNCS 5631 (pp. 54–64).

INEX (Initiative for the Evaluation of XML Retrieval) home pages. (2009). Retrieved January 23, 2009 from http://www.inex.otago.ac.nz.

Itakura, K., & Clarke, C. L. K. (2009). University of Waterloo at INEX 2008: Adhoc, book, and link-the-wiki tracks. In *Advances in Focused Retrieval*, LNCS 5631 (pp. 132–139).

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transaction on Information Systems, 20*(4), 422–446.

Jones, M., Buchanan, G., & Mohd-Nasir, N. (1999). Evaluation of WebTwig—a site outliner for handheld Web access. In *Proceedings of international symposium on handheld and ubiquitous computing*, LNCS 1707 (pp. 343–345).

Kamps, J., Geva, S., Trotman, A., Woodley, A., & Koolen, M. (2008c). Overview of the INEX 2008 ad hoc track. In *INEX 2008 workshop pre-proceedings* (pp. 1–28).

Kamps, J., Koolen, M., & Lalmas, M. (2008a). Locating relevant text within XML documents. In *Proceedings of SIGIR'08* (pp. 847–848).

Kamps, J., Lalmas, M., & Pehcevski, J. (2007). Evaluating relevant in context: Document retrieval with a twist. In *Proceedings SIGIR* '07 (pp. 749–750).

Kamps, J., Pehcevski, J., Kazai, G., Lalmas, M., & Robertson, S. (2008b). INEX 2007 evaluation measures. In *INEX 2007*, LNCS 4862 (pp. 24–33).

Kazai, G., & Lalmas, M. (2006). Extended cumulated gain measures for the evaluation of content-oriented XML retrieval. *ACM Transaction on Information Systems, 24*(4), 503–542.

Kekäläinen, J., & Järvelin, K. (2002). Using graded relevance assessments in IR evaluation. *Journal of the American Society for Information Science and Technology, 53*, 1120–1129.

Opera Software ASA, Opera Mini™ for Mobile. (2006). Retrieved January 21, 2009 from http://www.opera.com/mini/demo/.

Piwowarski, P. (2006). EPRUM metrics and INEX 2005. In *Proceedings of INEX 2005*, LNCS 3977 (pp. 30–42).

Piwowarski, B., & Dupret, G. (2006). Evaluation in (XML) information retrieval: Expected precision-recall with user modelling (EPRUM). In *Proceedings of SIGIR'06* (pp. 260–267).

Piwowarski, B., & Lalmas, M. (2004). Providing consistent and exhaustive relevance assessments for XML retrieval evaluation. In *Proceedings of CIKM '04* (pp. 361–370).

Reid, J., Lalmas, M., Finesilver, K., & Hertzum, M. (2006). Best entry points for structured document retrieval: Parts I & II. *Information Processing and Management, 42*, 74–105.

Robertson, S. (2008). A new interpretation of average precision. In *Proceedings of SIGIR '08* (pp. 689–690).

Saracevic, T. (1996). Relevance reconsidered '96. In *Proceedings of CoLIS* (pp. 201–218).
Tombros, A., Larsen, B., & Malik, S. (2005). Report on the INEX 2004 interactive track. *SIGIR Forum, 39*, 43–49.
Trotman, A., Pharo, N., & Lehtonen, M. (2007). XML IR users and use cases. In *Proceedings of INEX 2006*, LNCS 4518 (pp. 400–412).

Study VII

Paavo Arvola, Jaana Kekäläinen, Marko Junkkari (2010) Focused access to sparsely and densely relevant documents. In *Proceedings of the 33rd International ACM Conference on Research and Development in Information Retrieval, SIGIR 2010,* 781-782.

Focused Access to Sparsely and Densely Relevant Documents

Paavo Arvola
Dept. of Information Studies and Interactive Media
University of Tampere, Finland
paavo.arvola@uta.fi

Jaana Kekäläinen
Dept. of Information Studies and Interactive Media
University of Tampere, Finland
jaana.kekalainen@uta.fi

Marko Junkkari
Dept. Of Computer Science
University of Tampere, Finland
marko.junkkari@cs.uta.fi

ABSTRACT
XML retrieval provides a focused access to the relevant content of documents. However, in evaluation, full document retrieval has appeared competitive to focused XML retrieval. We analyze the density of relevance in documents, and show that in sparsely relevant documents focused retrieval performs better, whereas in densely relevant documents the performance of focused and document retrieval is equal.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Search process

General Terms
Measurement, Performance, Experimentation.

Keywords
XML retrieval, tolerance to irrelevance, focused retrieval.

1. FOCUSED ACCESS TO A DOCUMENT
Ideal information retrieval (IR) systems would return only relevant information to the user. In traditional document retrieval, returned documents typically include both relevant and non-relevant content. Approaches like passage and XML retrieval aim at returning the relevant content more accurately: the user should be guided directly to the relevant content inside the document instead of having to browse through the whole document. Surprisingly, in recent studies document retrieval has been found a competitive approach to focused XML retrieval according to retrieval effectiveness [e.g. 7]. However, some essential features in focused access to a document have been overlooked: the order of browsing, the user's reluctance to browse non-relevant information and the proportion of relevant text in documents. In the present study this proportion is referred to as the *density* of relevance of the document.

An XML retrieval system provides retrieved and assumedly relevant passages first to the user. If a returned passage turns out to be non-relevant, the user will not necessarily browse it through but rather continues with the next result. This user behavior is combined to effectiveness evaluation in the tolerance to irrelevance (T2I) metric [4], which models the user interrupting to browse after a given amount of non-relevant information is encountered. The sooner T2I is reached, the less the document benefits the effectiveness in evaluation.

In this study, we follow a browsing model with the given assumptions: A focused retrieval system guides a user to the relevant content, and the user starts browsing the document from the passages indicated by the system [1,2]. Returned passages are browsed first and the browsing continues until T2I is reached. With this model, effectiveness measures like precision and recall can be calculated for the document. These measures are calculated based on the proportion of relevant text browsed at the point where T2I is reached.

We compare focused XML retrieval with document retrieval by taking into account the access point to the document, browsing order and T2I. We analyze the effectiveness of retrieval at *the level of a retrieved relevant document.* More specifically, we examine the effectiveness by the density of relevance in documents. Our hypothesis is that focused retrieval provides a more effective access to the relevant content of a relevant document than full document retrieval, especially when it comes to sparsely relevant documents.

2. DENSITY AND DISTRIBUTION OF RELEVANCE
As the test collection we use the frozen Wikipedia collection [3] of more than 650,000 documents covering various subjects. The collection is used with topics and relevance assessments of the INEX 2008 initiative [6], where relevant passages (in 4,887 relevant documents) for each topic (totally 70) are assessed. All the relevant documents are sorted according to their ratio of relevant text to all text, i.e. how many percent of the document's text is relevant. Then the sorted list of documents is split into deciles, each covering 10% of the documents. The (rounded) lower boundaries of the density (relevance ratio) for the deciles are 0.005%, 2.4%, 6.6%, 12.1%, 24.4%, 58.4%, 94.9%, 99.3%, 99.7% and 99.9% (dec 1, dec 2,…, dec 10 respectively). That is, the last 4 deciles i.e. 40% of the relevant documents have a very high relevance density. Obviously, focused access to those documents does not bring any improvements.

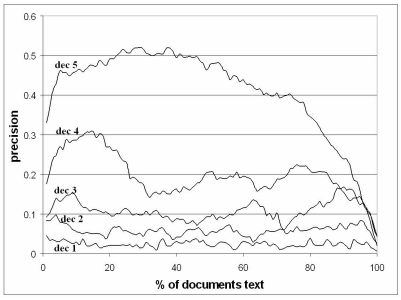

Figure 1: Average distribution of document's relevant text on five smallest deciles

SIGIR'10, July 19–23, 2010, Geneva, Switzerland.
ACM 978-1-60558-896-4/10/07

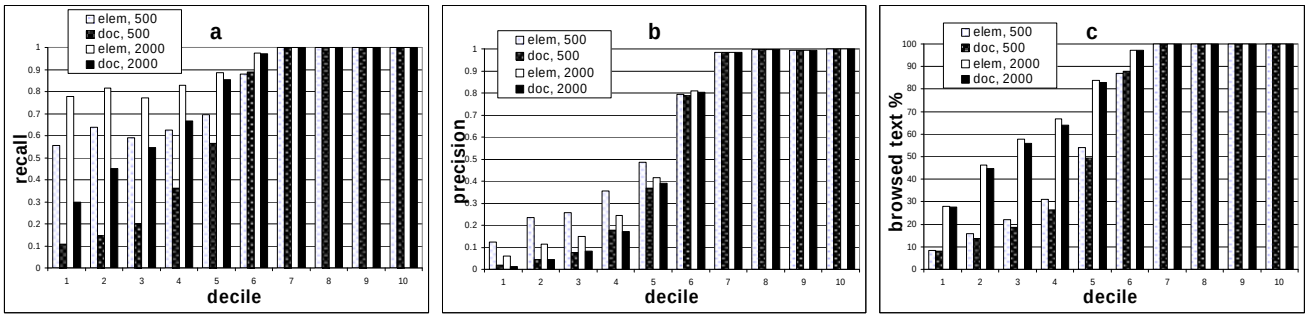

Figure 2: Average Recall (a), Precision (b), % browsed content (c) of relevant documents on each decile

Figure 1 shows the average precision of the five smallest deciles at percentages of the relevant documents' running text. The relevant content in the lowest deciles is somewhat steady across the average document, which means the relevant content may be at any location, whereas the fourth and fifth decile shows a slight bias towards the beginning of a document. The remaining deciles especially from 7 upwards draw a high, relatively straight line and are left out for the readability of the lowest curves.

3. PRECISION AND RECALL WITHIN A DOCUMENT

To study the benefit of focused retrieval strategy for the retrieval within a document on each decile, we selected the retrieved passages of each relevant document. These passages were provided by the best performing run at INEX 2008 (RiCBest, University of Waterloo [5]). Then we compared these focused results with a document retrieval baseline, where each relevant document is browsed sequentially. Figure 2 shows the average recall, precision and the percentage of browsed content in the relevant documents for each decile. The *elem* column refers to the focused retrieval strategy (i.e. RiCBest) while the *doc* column refers to the document retrieval baseline. We report figures on two T2I points: 500 and 2000 characters. In other words the browsing is expected to end when the user has bypassed 500 or 2000 non-relevant characters. The amount of 500 characters corresponds approximately to the next paragraph.

Figure 2c shows that the amount of browsed content is about the same for both of the strategies when assuming the same T2I. That is, the focused retrieval strategy does not reduce the amount of browsed content. However, with that amount the precision (Figure 2b) and especially the recall (Figure 2a) of the browsed content are notably higher with the focused strategy for half of the relevant deciles (dec1-5). The documents after sixth decile are uninteresting since they are densely relevant and neither browsing order matters nor T2I is reached.

4. DISCUSSION AND CONCLUSIONS

Focused retrieval is beneficial in locating the relevant content in between non-relevant material. Therefore documents with high relevance density are not interesting in the scope of focused retrieval. The results show that the less relevant content, the better the focused retrieval performs. In plain document retrieval, the user is responsible for finding the relevant content within a sparsely relevant document. This leads into poor performance with documents having only some relevant content, when T2I is assumed. Namely, in many cases the browsing ends before the relevant content is met. This leads to zero recall. On the other hand, if the relevant content is pointed out accurately, the recall for the document is typically 1 (100%). However, due to the nature of T2I, where the browsing goes on with the non-relevant material until the T2I is met the precision is always less than 1.

While we assume the T2I and browsing order in focused retrieval, our findings differ from the previous studies, where the full document retrieval has been a competitive approach [7]. This is due to the densely relevant documents, where the focused retrieval systems tend to retrieve only parts for why the recall per document remains low. This has led into overemphasizing precision, which is taken into account four times more than recall in the official metrics (i.e. the F-Measure).

5. ACKNOWLEDGEMENTS

The study was supported by the Academy of Finland under grants #115480 and #130482.

6. REFERENCES

[1] Arvola, P. 2008. Passage Retrieval Evaluation Based on Intended Reading Order. LWA 2008, 91-94.

[2] Arvola, P., Kekäläinen, J., Junkkari, M. 2010. Expected Reading Effort in Focused Retrieval Evaluation. To appear in Information Retrieval.

[3] Denoyer, L., and Gallinari, P. 2006. The Wikipedia XML Corpus. Sigir Forum, 40:64-69.

[4] de Vries, A.P., Kazai, G., and Lalmas, M. 2004. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In Proceedings of RIAO 2004, 463-473.

[5] Itakura, K.Y., and Clarke C.L.A. 2009. *University of* Waterloo at INEX 2008: Adhoc, Book, and Link-the-Wiki Tracks, In Advances in Focused Retrieval, 132-139.

[6] Kamps, J., Geva, S., Trotman, A., Woodley, A., and Koolen, M. 2009. Overview of the INEX 2008 ad hoc track. In Advances in Focused Retrieval, 1-28.

[7] Kamps, J., Koolen, M., and Lalmas, M. 2008. Locating relevant text within XML documents. In Proceedings SIGIR '08. ACM, New York, NY, 847-848.