Focused Information Access using XML Element Retrieval

Börkur Sigurbjörnsson

Promotor: Prof.dr. Maarten de Rijke

Co-promotor: Dr.ir. Jaap Kamps

Committee:
Prof.Dr.-Ing. Norbert Fuhr
Prof. Mounia Lalmas
Prof. John Mackenzie Owen
Dr. Maarten Marx
Dr.ir. Arjen de Vries

http://phd.borkur.net/

Printed and bound by Print Partners Ipskamp

ISBN-10: 90-9021317-1

ISBN-13: 978-90-9021317-0

Focused Information Access using XML Element Retrieval

Academisch Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof.mr. P.F. van der Heijden
ten overstaan van een door het college voor
promoties ingestelde commissie, in het openbaar
te verdedigen in de Aula der Universiteit
op donderdag 14 december 2006, te 11.00 uur

door

Börkur Sigurbjörnsson

geboren te Reykjavík, IJsland.

Promotor: Prof.dr. M. de Rijke

Co-promotor: Dr.ir. J. Kamps

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

SIKS Dissertation Series No. 2006-28

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Acknowledgments

In the autumn of 2002 I was discussing my future prospects with my master's thesis supervisor Maarten de Rijke. When I told him that my job-hunt had not delivered any results so-far, Maarten replied: "In a worst case scenario you should consider doing a PhD." After a short pause he added that a doing a PhD would probably not turn out to be a worst case scenario. A few months later I joined Maarten's group as a PhD student. Worst case scenario or not, the past four years have been a pretty good experience.

I owe gratitude to my PhD supervisors Maarten de Rijke and Jaap Kamps for their supervision over the past four years. I am very grateful for their rigorous feedback on various versions of this manuscript—resulting in a useful hint of verbosity in my otherwise succinct writing style. I also thank Maarten Marx for his occasional advise.

This thesis is based on extensive usage of the INEX XML retrieval test collection. Hence, I am grateful to the whole INEX community for creating this test collection. I also thank Andrew Trotman for co-organizing the INEX topic format specification.

I am grateful to the SIGIR 2005 Doctoral Consortium Program Committee for giving me the opportunity to present my research and get their feedback. In particular, I want to thank my doctoral consortium advisors Yoelle Maarek and Nick Belkin for the fruitful one-on-two discussions.

I also want to thank my students of the 2005 and 2006 editions of the Project Information Retrieval course for assisting me in my research. In particular, am I grateful to the 2005 students who created a user interface to my XML element retrieval system; key aspects of that interface are still in use today.

I am grateful to my thesis committee: Norbert Fuhr, Mounia Lalmas, John Mackenzie Owen, Maarten Marx, and Arjen de Vries. I thank you for accepting to be on my committee and for the feedback you gave on the manuscript.

I would like to thank the "ILPS lunchers" for enlightening lunch discussions about important issues facing our society today—such as the usefulness of drugs

tests for best-paper-award winners and the most appropriate way to wish someone a happy vacuum cleaning experience in Dutch. Without the lunch discussions the working days would have made way too much sense.

Last but not least, I want to thank my friends and family for their outstanding performance in the roles of supporting actors throughout the years. I thank my friends and family abroad for visiting me in Amsterdam and for their hospitality on my visits to Reykjavík, København, London, Saint-Prex, Åsgårdstrand, Saarbrücken, Liverpool, Toronto, etc. Finally, I thank my parents for regularly making sure that I have had enough to eat.

Amsterdam, October 23, 2006

Börkur Sigurbjörnsson

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Search engines play an important role in our daily lives. We use search engines to access a wide range of information sources: the web, our email, our desktop computers, library catalogs, etc. Most popular Internet search engines work via simple interfaces. Figure 1.1 shows a simplified view of the interaction between a user and a search engine. The user describes her information need by entering a few keywords into the so-called "search box." The search engine locates the documents that are likely to fulfill her information need, and then returns to the user a list of relevant documents, ranked by the likelihood that they satisfy her need. The search engine presents each relevant document by displaying its title and a short query based summary of the document's content—a text snippet. By clicking on the document title the user is brought to the corresponding document. This simple interface of search engines is very powerful since it can be applied to almost any document collection and can be used by almost any user.

When a user is presented with a ranked list of relevant documents her search task is usually not over. The next step for her is to dive into the documents themselves in search for the precise piece of information she was looking for. If the documents are long, this can be a tedious task. As an example, suppose a user is interested in hiking routes in northern Europe and the search engine locates a 50 page travel guide about Sweden. Then the user might have to do quite a bit of "scrolling" within the document before she has collected all information about hiking routes in Sweden. Can we give a better kind of support for the user in this scenario? Can we give her a more focused type of access to the relevant information?

Focused Information Access

The notion of "focused information access" can be used as a label for a wide range of applications. E.g., a medical search engine can be considered as "focused" since it focuses on a particular type of corpora [Tang et al., 2005]; a factoid

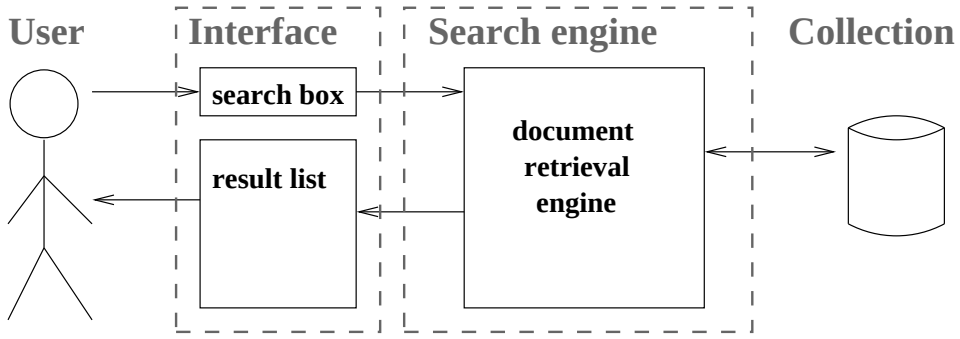


Figure 1.1: Simplified picture of the interaction taking place when a user posts a query to a search engine.

question answering engine can be considered as "focused" since it gives very focused answers [Green et al., 1963]; a passage retrieval system can be considered "focused" if it gives access to the most relevant passages of a document, rather than to the document as a whole [Salton et al., 1993].

In this thesis we explore the task of giving the user direct access to the relevant information, rather than merely the relevant documents. Our task can be seen as a special case of passage retrieval where the passages are defined in terms of document structure. More precisely, we study the XML element retrieval task [Kazai et al., 2004b]. Before we describe the XML element retrieval task, let us introduce semi-structured documents, and in particular XML documents.

Semi-structured Documents

All text has structure, and structure comes in different kinds such as linguistic structure, document structure, and layout structure. Some structure is implicit, such as a chain of arguments that the author uses to tell her story. In this thesis we focus on explicit text structure, such as paragraph segmentation, or assigned metadata. The form of explicit structure differs between documents and document collections. In its simplest form, flat-text documents have little explicitly marked-up structure, limited to sentence boundaries determined by a full-stop and perhaps paragraph boundaries determined by an empty line. Today, however, electronic documents are commonly marked-up with additional structure. Especially if documents are long and discuss multiple facets, it is necessary to make the text more accessible to the reader by adding structure such as section headings etc. This type of structure is usually added manually by the document author either directly using some sort of markup language, or by using advanced text editing tools.

The markup of text documents serves different purposes. Markup can be used to represent different levels of granularity of *text objects*. As an example, a text document may be divided into sections; and the sections into sub-sections

```
<travel_guide>
  <title>Guide to Sweden</title>
  <p>In this guide you will get all information you need for...</p>
  ...
  <section>
    <title>Smaland</title>
    <section>
      <title>Hotels in Smaland</title>
      <p>The Smaland area offers great variety in  accommodation ... </p>
      ...
    </section>
    <section>
      <title>Hiking in Smaland</title>
      <p>Lake-side strolls are a popular means to explore the ... </p>
      ...
    </secction>
  </section>
  <section>
    <title>Lapland</title>
    <section>
      <title>Hotels in Lapland</title>
      <p>Have you ever stayed in an ice hotel? The Ice hotel ... </p>
      ...
    </section>
    <section>
      <title>Hiking in Lapland</title>
      <p>Mount Kebnekaise is the highest mountain in Sweden ... </p>
      ...
    </section>
  </section>
</travel_guide>
```

Figure 1.2: Example of an XML document.

and paragraphs, etc. Markup can also be used to give special *"semantics"* to a certain piece of text. As an example, one might define section titles, document title, author names, etc. Finally, markup is often used to describe the *layout* of the text. As an example, the author can indicate that some words should be displayed in italics, other should be in boldface, etc.

Document structure can be marked-up using a number of different markup formats, such as, Microsoft-Word format [MS-Word], Portable Document Format [PDF], a scientific document preparation style [LaTeX], HyperText Markup Language (HTML) [Raggett et al., 1999], etc. In this thesis we will work with a general markup language, namely the eXtensible Markup Language (XML) [Bray et al., 1998]. XML is a flexible markup language which serves as a representative example of modern semi-structured markup languages. Figure 1.2 shows an example of an XML document. The example document can be seen as an XML

version of the travel guide mentioned earlier in this chapter.

Semi-structured text documents—and document-centric XML documents in particular—provide an ideal framework for the type of focused information access that we want to address in this thesis. The structural markup provides good handles on the text units to which we want to give focused access. The tags can be used as segment boundaries when giving access to the relevant sub-parts of document.

XML Element Retrieval

Using XML element retrieval to give focused information access has several advantages. First, as mentioned before, XML document collections are a representative example of modern semi-structured document collections; and the XML language is a "de facto" standard language for semi-structured documents. Second, in recent years, much attention has been given to the evaluation of XML element retrieval within the INitiative for the Evaluation of XML retrieval (INEX) [Kazai et al., 2004b].

In XML element retrieval, each element is considered as a retrievable unit. E.g., considering Figure 1.2 again, the root element (`<travel_guide>`), each section (`<section>`), each paragraph (`<p>`), each section title (`<title>`), etc. is a potential unit to retrieve. The INEX initiative has built an evaluation collection for evaluating the XML element retrieval task—which can be defined as follows:

XML element retrieval For each element in the collection estimate how relevant it is for the user's information need. This process is approximated by creating a ranked list of XML elements for each user query. The elements are ranked by decreasing likelihood of being relevant for the user's information need. More precisely, the XML retrieval engine should retrieve "the most specific relevant document components, which are exhaustive to the topic of request" [Gövert and Kazai, 2003, page 2].

Hence, the goal of the XML element retrieval task is to produce a ranked list of XML elements as a response to a query. If we return to our example document in Figure 1.2, we would expect the two sections—labeled "Hiking in Smaland" and "Hiking in Lapland"—to be ranked highly for the query *hiking northern europe*.

The main bulk of this thesis is about the XML element retrieval task. We show how this task can be modeled by adapting and extending existing retrieval models and we implement an XML element retrieval engine based on those models. We provide a rigorous evaluation of the retrieval engine using the INEX test collection.

Focused Information Access using XML Element Retrieval

Let us now briefly explain how our XML element retrieval engine can be used to give focused access to information. To this end, we return to our example scenario

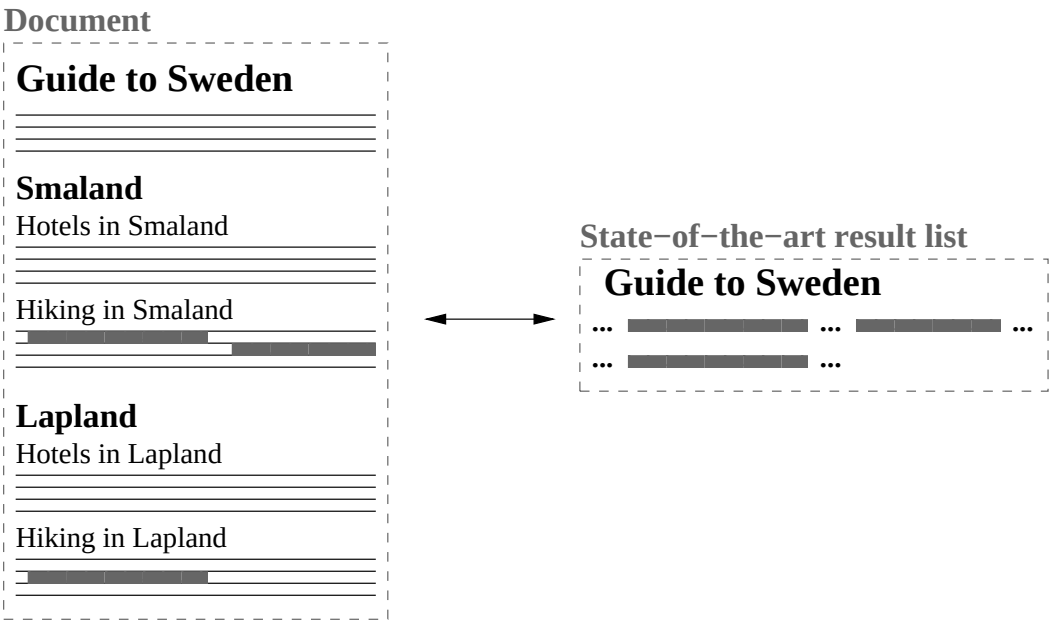


Figure 1.3: Example of how a single document is summarized in a state-of-the-art result-list. Suppose we have the query "hiking northern europe" and the grey shaded areas represent the text-snippets generated for the query. Left: the original document. Right: Document summary of state-of-the-art search engines.

where our user was looking for hiking routes in northern Europe. Figure 1.3 shows a simplified view of how state-of-the-art search engines would present the Swedish travel guide to the user as a search result for the query "hiking northern europe." Although the interface of state-of-the-art search engines is in general a powerful approach, it has several drawbacks in the case of our example scenario—i.e., the scenario where the information need is specific and can be answered by a relatively small portion of a longer document.

- First, the text snippet only gives evidence that the information exists somewhere in the document, but does not relate it to the overall discourse structure of the document. E.g., the result in Figure 1.3 does not state that the text snippet is composed of text from two distinct sections.

- Second, access is only given to the beginning of the relevant document and it is left to the user to search within the document for the desired information. In terms of the example in Figure 1.3, when clicking on the document title in the result-list, the user is given access to the beginning of the document and has to "scroll around" herself in order to find the two sections that are of interest.

The main proposal of this thesis is to use an XML element retrieval engine—that ranks individual parts of the document—to address these two shortcomings of

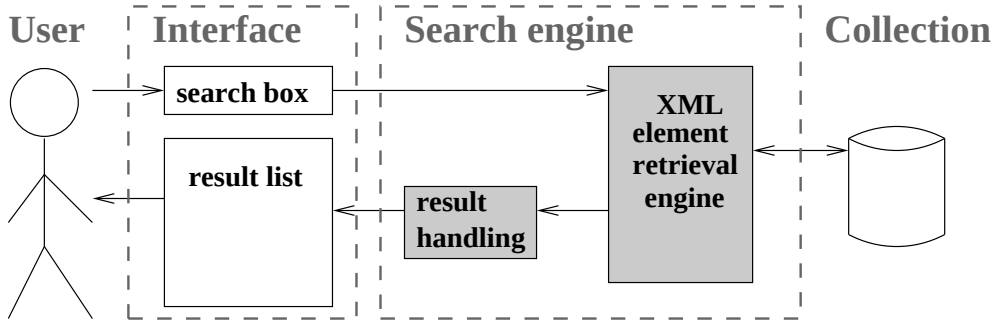


Figure 1.4: Simplified picture of the interaction taking place when a user posts a query to our focused search engine.

state-of-the-art search engines. Figure 1.4 shows a simplified picture of how we can put the XML element retrieval engine in to action. The figure is an extension of Figure 1.1 where we have replaced the document retrieval engine with an XML element retrieval engine, and added a result handler which implements focused information access by extending the state-of-the-art search engine approach with two new features:

Structured result lists Our result list uses XML element retrieval results to give a clear indication about the relation between the user's query and the "discourse structure" of the document. I.e., instead of showing only one text snippet for each document we show a text snippet for each relevant element, together with a partial "table of contents." E.g., in Figure 1.5 the result indicates that the relevant text can be found in two separate sections. The structured result list helps users to assess whether or not the retrieved documents are likely to answer their information need. Hence, they can make better judgments about which documents they should explore further.

Direct linking Direct access is given to relevant portions—relevant elements—of documents. Following these links, users get to the relevant information with less effort and less time. Additionally, if the users need to skim over a long document they are more likely to miss relevant information than if they are explicitly pointed to the part they should read. In terms of the example in Figure 1.5, by clicking on each of the section titles, "Hiking in ...", the user is brought directly to the corresponding section.

Focused information access is a good example of how XML element retrieval can be used in an operational setting. However, XML element retrieval may be used in a broader range of applications, e.g., document summarization for handheld devices [Buyukkokten et al., 2000], predictive annotations [Prager et al., 2000], the exploration of linguistically annotated corpora [Bird et al., 2005], and question answering using XML-based strategies [Ahn et al., 2006].

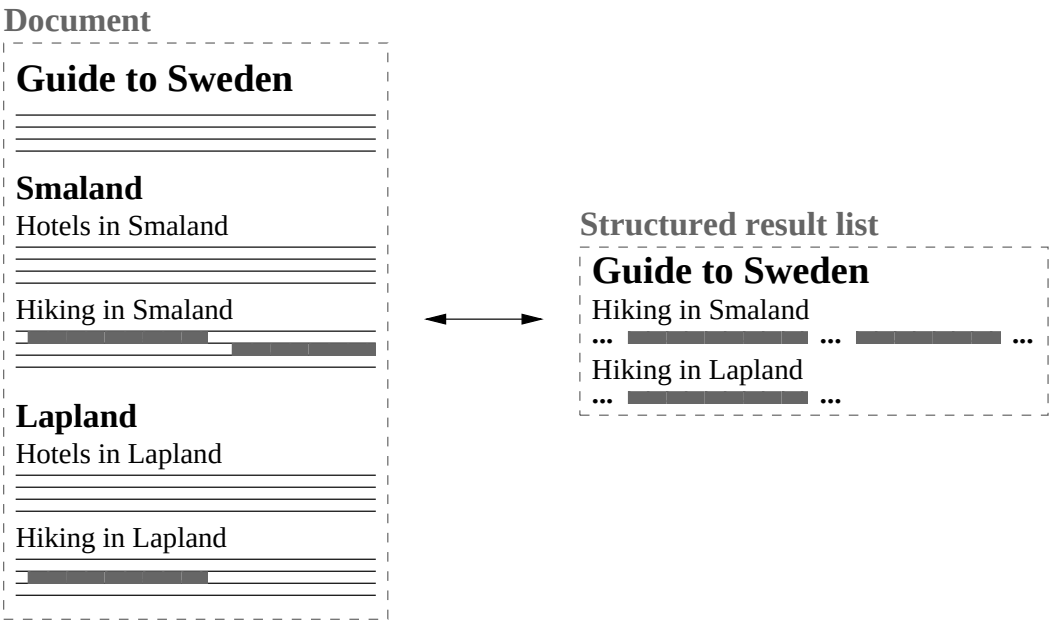


Figure 1.5: Example of how a single document is summarized in a structured result-list. Suppose we have the query "hiking northern europe" and the grey shaded areas represent the text-snippets generated for the query. Left: the original document. Right: Document summary as used in a structured result-list.

1.1 Research Questions and Contributions

The overall research question that we will address in this thesis is:

> How can we give focused information access to semi-structured documents using XML element retrieval techniques?

This question is composed of two sub-questions. A *system-oriented* question:

> How do we rank individual XML elements?

and a *user-oriented* question:

> How do we design an appropriate interface for providing focused information access?

Before we describe our more detailed research questions we give a short overview of the setup of this thesis. In Chapters 3–7 we address the *system-oriented* sub-question—we model, implement, and evaluate an XML element retrieval engine. In Chapter 8 we address the *user-oriented* sub-question—we implement and evaluate an interface for focused information access using our XML element retrieval engine as back-end. In Chapter 2 we situate the focused information access task and the XML element retrieval task in the broader context of information retrieval research; and in Chapter 9 we conclude. Now, let us look at the more detailed questions that we address in this thesis.

Elements vs. documents Given the long tradition in document retrieval research our first specific research question is about how the relative newcomer—XML element retrieval—compares to the well established document retrieval task:

> How is XML element retrieval different from document retrieval?

We apply a document retrieval model to the XML element retrieval task and observe differences in retrieval settings and performance. This analysis gives us insights into the difference between the two retrieval tasks.

Length normalization Our baseline experiments identify length normalization as being one of the core differences between element and document retrieval. We study this further and ask ourselves:

> What is the impact of length normalization for XML retrieval?

We show that one of the major differences between document retrieval and XML element retrieval is the *length distribution* of retrieval units. The length of documents usually follow a normal distribution, but the length of XML elements is a very skewed distribution—most elements are very short while a few ones are long. The distribution of the length of both relevant elements and relevant documents is, however, close to being a normal distribution. We show that *length normalization*—while useful for document retrieval—is essential for XML element retrieval.

Unit-of-retrieval The notion of *unit of retrieval* is a core notion in XML element retrieval. Our next research question is about whether all units are the same:

> Are all element types equally important retrieval units?

We analyze the INEX assessments and see that sections and paragraphs are the most common type of elements considered as relevant, followed by very long elements such as whole articles, but shorter element types appear infrequently in the assessments. We experiment with *selective indexing* where we remove either the shortest or the longest elements from our index. Our main finding is that leaving out the short elements does not degrade retrieval performance. The long elements—whole articles and article bodies—are, however, essential for achieving good retrieval performance.

Context In XML element retrieval, elements are considered to be atomic retrieval units. However, elements exist in the context of their surrounding XML document. Our next research question considers this context:

> Can we improve XML element ranking by incorporating element's context into the retrieval model?

We show that a *mixture model*, where we rank the elements using a mixture of element-, document-, and collection-models, leads to significant improvements in retrieval effectiveness.

Evaluation As mentioned above, one of the main differences between document retrieval and XML element retrieval is the unit of retrieval. This difference does not only have an effect on the way we model our retrieval process, but also on the way we evaluate retrieval performance. XML element retrieval evaluation methodology is still an active research area, but evaluation methodology in itself is not a research issue in this thesis. However, this thesis relies heavily on being able to evaluate XML element retrieval, and thus we cannot ignore the issue altogether. Hence, we have a "passive" research question about evaluation:

> How does our system's performance change when we use a different evaluation setup?

For the main evaluation in this thesis we choose a single evaluation framework—the "official" system-oriented framework as defined by the INEX initiative. We also look at some "unofficial" ways to evaluate our system. We look at system performance over different topic classes; we look at how the hierarchical nature of the documents—a.k.a. overlap—affects our evaluation; and we look at how element results can be used for ranking documents. The main outcome of this analysis provides insights into the factors that play a role in XML element retrieval evaluation.

Focused Information Access In our final research question we go back to the example scenario which we drew up in the beginning of this chapter—i.e., where the user needed focused access to information within the relevant documents. We ask ourselves how XML retrieval can be of use in this case:

> How can we put XML retrieval into action as a part of an operational system for giving focused information access?

We implement the two focused information access features—*structured result lists* and *direct linking*—using our XML element retrieval engine as a back-end. We discuss the main challenges that need to be faced when XML element retrieval is put into action. We evaluate the system in two interactive experiments. The main outcome of the evaluation is that users do appreciate the structured result list and the reduced search effort provided by direct linking.

1.2 Thesis outline

In this section we give a chapter-by-chapter outline of the remainder this thesis.

Chapter 2: Retrieval Tasks and Evaluation We provide a general overview picture of search tasks; and locate our focused information access task and our XML retrieval task within the overall picture. We give an overview of information retrieval evaluation; and introduce the INEX XML retrieval test collection which serves as the major evaluation framework for the methods described in this thesis.

Chapter 3 : A Baseline XML Element Retrieval System We introduce the main building blocks of our XML element retrieval system. We take an existing document retrieval system and adapt it to the element retrieval task. We explore the effect of changing the most basic parameter of our baseline element retrieval system, the smoothing parameter, and compare our findings to similar exploration for the document retrieval task. Our main finding is that, surprisingly, the smoothing parameter serves as a tool for controlling length of retrieved elements. The work in this chapter is partially based on work published in [Sigurbjörnsson et al., 2004a, Kamps et al., 2003a].

Chapter 4: Length Normalization We take an in-depth look at the role of the length normalization in XML element retrieval. We analyze the INEX relevance assessments and show that there is a considerable difference between the length distribution in the set of retrievable elements and the set of relevant elements. We show that effectiveness XML element retrieval can be improved significantly by using so-called *length priors* which bridge the length gap between the collection and assessments. The work in this chapter is based on work published in [Kamps et al., 2003b, 2004a, 2005a]

Chapter 5: Unit of Retrieval We take a closer look at the role of *unit of retrieval* in the XML element retrieval task. We analyze the tag-names of the relevant elements and explore if *selective indexing* strategies can improve retrieval performance. Our findings main findings are that we can safely remove the shortest elements from our index without harming retrieval effectiveness. However, removing the longest element from our index—and hence our retrieval runs—has a significant negative effect on the retrieval performance. The work in this chapter is partially based on work published in [Kamps et al., 2005a, Sigurbjörnsson and Kamps, 2006].

Chapter 6: Mixture Models In the previous chapters we have considered elements as atomic units. In this chapter we consider the context in which elements reside, and look at the elements in relation to their containing document. We look at how term statistics from the surrounding document can be incorporated into the calculation of element relevance. We implement this by using so-called *mixture models* which lead to a significant improvement of retrieval effectiveness.

The work in this chapter is based on work published in [Sigurbjörnsson et al., 2004a, 2005]

Chapter 7: Topic Classes, Overlap and Document Ranking We look at the evaluation of the XML element retrieval task from three different angles. First, we look at performance over a number of different classes of topics. Second, we evaluate our system without rewarding retrieval of overlapping text. Third, we evaluate how we can use our element retrieval results to rank documents. The main value of the work in this chapter is to gain insight into the factors that play a role in XML element retrieval. The work in this chapter is partially based on work published in [Sigurbjörnsson et al., 2005].

Chapter 8: Element Retrieval in Action We show how we can put our XML element retrieval system in action through a user interface which implements the two focused information access features described in the introduction chapter. We highlight the main issues that need to be taken into consideration in the interface design. Our evaluation results indicate that users do indeed appreciate to be given focused access to information. The work in this chapter is based on work published in [Kamps and Sigurbjörnsson, 2006, Sigurbjörnsson et al., 2006].

Chapter 9: Discussion and Conclusions We end the thesis by discussing the main conclusions that can be drawn from the research presented in this thesis. We also look ahead and discuss how this work might be extended in future research.

In this thesis we have decided to focus on XML retrieval using content-only queries. This means that we do not report here on our extensive work on content-and-structure queries [Sigurbjörnsson and Trotman, 2004, Trotman and Sigurbjörnsson, 2005a,b, Kamps et al., 2004b, Sigurbjörnsson et al., 2004b, Sigurbjörnsson et al., 2004, Kamps et al., 2005b, 2006].

Chapter 2

Retrieval Tasks and Evaluation

In this chapter we provide background material on information retrieval and introduce the evaluation framework used in this thesis. First we review the literature on search tasks and user-oriented models of search behavior. We explain how information retrieval systems are evaluated, both using laboratory experiments and interactive experiments, and we introduce the INEX evaluation framework that will serve as the main evaluation framework for our experiments. Before we go into details, we give a high level discussion of each section in this chapter.

Section 2.1, gives an overview of literature on information search behavior. We address the issue from a user perspective. We review three types of work: behavioral models of information seeking, interactive studies, and query-log analysis. The main contribution of this section is to locate our focused information access task relative to the 'big picture' of the interaction between users and information. In Section 2.2 we give an overview of information retrieval evaluation. We introduce both laboratory experiments—using re-usable test collections—and interactive evaluations. In Section 2.3 we give an overview of the INEX laboratory test collection. In Section 2.4 we give an overview of the INEX interactive track (iTrack). In Section 2.5 we describe the experimental setup of our evaluation in this thesis. Finally, in Section 2.6 we conclude the chapter.

In this chapter, we will not give an overview of related work on XML element retrieval systems. This we do in each of the more technical chapters 3–8 below.

2.1 Information Search Behavior

In this section we locate our work within the 'big picture' of *information behavior* [Wilson, 1999]. We start by looking at information behavior models which describe, on a broad level, users' interaction with information. We narrow our attention to *information seeking behavior* and look at both theoretical models and empirical studies. We then zoom in further on *information search behavior*,

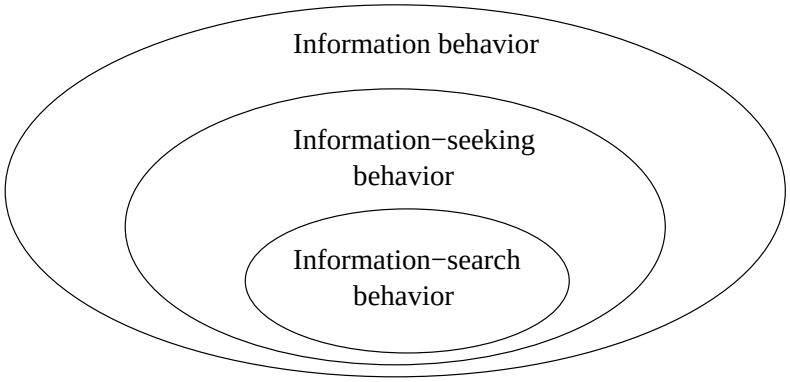


Figure 2.1: Wilson's nested model of the information seeking and information searching areas [Wilson, 1999, page 263].

and in particular on the role of our *focused retrieval approach* in the information search process.

2.1.1 Theoretical Models of Search Behavior

As outlined in Chapter 1 we motivate our application of focused information access by a specific scenario—a user and her rather specific information need. In this section we will first give a literature overview on the more general theme of "searchers and their context", then we will relate our focused information application to the key conceptualizations from the literature.

We discuss the theoretical models of search behavior in terms of Wilson's nested model of information seeking and information searching research areas (Figure 2.1).

Information Behavior Models of information behavior describe very generally a framework for the interplay between people and information.

Wilson and Walsh [1996] describe how a 'person-in-context' interacts with information. The model covers a wide range of research areas, including information science, decision making, psychology, and consumer research. The model attempts to describe aspects of user-information interaction, the origin of information needs, the choice of information source, active search and the use of search results. The work presented in this thesis is confined to the stage that Wilson refers to as 'active search,' i.e., the users' interaction with the search engine. The other stages—choice of information sources and use of search results—are beyond the scope of this thesis.

Dervin [1983] introduced the 'sense-making' framework which can be used to model how users make sense of reality. The model has four components: a *situation* which describes the context where an information problem arises; a *gap* which describes the difference between the contextual situation and the desired

situation; an *outcome* which describes the result of the sense-making process; and a *bridge* which is some sort of mechanism for closing the gap between the situation and the outcome. The set-up of this thesis can be described in the terminology of Dervin's model. We motivate our work by a situation where a user has a specific information need which is satisfied by a relatively small portion of a relevant document. The main content of this thesis is to build and evaluate tools that can bridge the gap to the desired outcome—namely, direct access to the relevant information.

Information Seeking Behavior Models of information seeking model the variety of methods users apply to discover and gain access to information resources.

Ellis et al. [Ellis et al., 1993, Ellis and Haugan, 1997] model the activities involved in information seeking. They identify six information seeking activities: starting, chaining, browsing, differentiating, monitoring and extracting. Several of these stages involve some sort of "focusing" where the user moves toward a more focused view of the whole information space. In this thesis we will mainly look at what Ellis referred to as extracting and verifying, i.e., identifying relevant material in an information source and checking the accuracy of the information.

Information Searching Models Information searching is a sub-set of information seeking, where the main focus is on the interaction between users and computer based information retrieval systems.

Ingwersen [1996] discusses a cognitive theory for information retrieval interaction. He stresses the importance of the work task when modeling search behavior. He argues that in the evaluation of information retrieval systems the notion of information need should not only consider topicality but also the cognitive state and work task of the searcher. I.e., the relevance of information depends on the user's cognitive state and work task. This may impact the focused information access task explored in this thesis since the appropriate focused access may depend on the work task or background knowledge of the searcher. E.g., a novice reader may require considerable context around a topically relevant text unit, while for a domain expert the topically relevant text unit might be sufficient—her background knowledge gives enough context. Similarly, a different type of focused access may be desired for a user searching for the query 'earthquakes in Turkey', depending on whether her task is to compile a list of earthquakes in Turkey or if she is writing a general overview paper on earthquakes in Turkey.

We will provide more background on related work on searching behavior when we discuss empirical work below (Section 2.1.2).

Focused Information Behavior

Let us sum up how our focused information access problem relates to the different conceptualizations in literature. The retrieval system is the central point of our

work. The bulk of our work does therefore fall with in the 'active search' part of Wilson's model. Wilson's notions of 'person-in-context' and 'context of information need'—together with Dervin's notion of 'situation' and Ingwersen's work tasks—are also important for focused information retrieval. In our introduction of focused information access—as we address it in this thesis—we motivated our approach by the context of the user and her information need. Our motivation for the usefulness of our focused retrieval system is based on the presence of the following context scenario:

Nested information The information that fulfills the user's need appears locally nested within a longer document in the collection. I.e., full documents are too long to be considered the appropriate units of retrieval.

Our focused retrieval approach depends on a number of "context variables," such as the *data* being searched, the *person* searching, and the *task* underlying the search.

Data What is being searched? Does the data support focused access? What sort of documents are being searched? Long? Short? Technical? etc.

Searcher Who is searching? Can the searcher make use of the focused access? Is the searcher an expert or novice? How well is the user acquainted with the collection? etc.

Task Which task is the user performing? Is focused access a suitable tool for the particular task? Is the user looking for a specific answer? Is the user looking for articles to be included in a survey of related work? Is the user looking for articles to learn about a new field? etc.

When we answer the question whether our system is suitable for satisfying a particular information need, we must look at a combination of the variables above. Some document collections are organized as a mixture of long and short documents where a specific information need might be answered either by a part of a longer document or as a full short document. As an example, suppose we need to find information on the 'National Convention era of the French Revolution.' In the World Book [World Book] the National Convention era is covered in one chapter nested inside a long article on the French Revolution. If we search the World Book, our information need is likely to be answered by a part of a longer document. In Wikipedia [Wikipedia], however, there is a whole page devoted to the National Convention era. Consequently, if we search Wikipedia, our information need is likely to be satisfied with a complete document. Hence, the usefulness of our focused approach is not solely dependent on some sort of "focused information needs," but the combination of the need and the collection.

2.1.2 Empirical Studies of Search Behavior

Let us now turn our attention to empirical studies of search behavior. We start by surveying several empirical studies and then we stop and relate the main observations to our own task of providing focused information access. The goal of the survey is to review related work on three aspects of our focused information access application: *target audience* of our application, *search tasks* for which the application is likely to be useful, and desired *functionality* of our application.

Choo et al. [2000] extend Ellis' model in a model of information seeking on the web. They define four modes of information seeking on the web and relate those to Ellis' information seeking activities. The four modes are: *undirected viewing*, where "the individual is exposed to information with no specific information need in mind;" *conditioned viewing*, where "the individual directs viewing to information about selected topics or to certain types of information;" *informal search*, where "the individual actively looks for information to deepen the knowledge and understanding of a specific issue;" and *formal search*, where "the individual makes a deliberate or planned effort to obtain specific information or types of information about a particular issue." The research in this thesis can be considered as addressing a subset of the informal and formal search modes—in particular we conjecture that our focused information access system is useful for the formal search tasks since "[t]he granularity of information is fine, as search is relatively focused to find detailed information." Choo et al. study the information seeking behavior of a group of "technically proficient Web users" via questionnaires, browser logs and personal interviews. Most information seeking episodes were informal searches (37.7%), followed by conditional viewing (29.5%), undirected viewing (19.7%), and formal search (13%).

Navarro-Prieto et al. [1999] investigate search strategies of web users in an interactive study. The test persons are computer science and psychology students. The researchers analyze three factors of how users interact with hyper-media: user experience, the type of search task, and information presentation. They explore two different types of tasks, *fact finding* and *general exploration*. The main observations are that experienced users apply different search strategies when solving the two different tasks, while there is not a clear distinction for the inexperienced users. For the fact finding task, the experienced users applied a bottom-up approach where they go from keywords to pages via search engines. In solving the task, the users did not browse within sites. For the general exploration task, the experienced users more often tried to narrow down their search by following links from pages the search engine brought them to.

Broder [2002] lays out the well-known taxonomy of web searches. He defined three classes of searches: *Navigational*: The user needs to locate a particular website. *Informational*: The user needs to acquire information which she assumes to exist on one or more web sites. *Transactional*: The user needs to locate a web site where she can perform a particular transaction. Through a user survey and query

log analysis Broder estimates that 40–50% of Internet searches are informational, 30–35% are transactional, and 20–25% are navigational.

Rose and Levinson [2004] try to understand the goals in Web search. Their classification of goals is an extension of Broder's taxonomy. They devise a hierarchical taxonomy where Broder's classes form the top level, except the transactional class has been replaced by a more general resource-finding class. The informational class was further divided into 5 subclasses: directed (specific), undirected (general), advice, locate (where to find a real world service/product), and list. An AltaVista query log (and following user interaction) was analyzed and queries were classified. About 61% of the queries were classified as informational. Of the informational queries, about 40% were undirected, about 40% were location, and only about 8% were directed.

Slone [2003] looks at the effect of age, search goals and experience on web search performed by visitors of a public library. It is difficult to generalize anything from this study due to its small population of users. The main observation to be read from this paper is that there is an "expected" correlation between age, experience and the search approach. The youngest group and the oldest group had, in general, the least Internet experience. The more experienced users applied more sophisticated search approaches.

Toms et al. [2003] look at the effect of task domain on search. Specifically, they look at four task domains: consumer health, general research, shopping and travel. They found significant difference between search approach used for solving tasks in different domains. When searching in shopping and travel, searchers spent more time browsing within a site. When searching in research and health, searchers spent more time exploring hit-lists. Importantly, they conclude that one-interface-fits-all is not a suitable approach. Based on their analysis they come up with design requirements of interfaces for each of the task domains. In the health domain, the information level (professional vs. lay), scope (brief vs. detailed), and source (academic vs. commercial) should be incorporated in the hit-list. In the research domain, the information level (overview, detailed, scientific) and format (journal article, newspaper, statistics) should be integrated into the hit-list. Users need "a quicker and more effective way to evaluate the content of a website from the hit-list" [Toms et al., 2003, page 8]. In the shopping domain, strong queries were required to "pre-determine functional versus informational needs" and "queries must be processed so that product type, brands, product names, and stores are distinguished" [Toms et al., 2003, page 8]. In the travel domain, the hit-list should differentiate between general information (e.g., culture and climate), and specific travel services (e.g., tour and ticket bookings). Interfaces should indicate the relative weight of information about the destination vs. activity.

People can have very specific information needs, which can be formulated as a question whose answer is merely a factoid [Voorhees, 2005]. Factoid question answering is widely supported in state of the art search engines. The most famous example is probably [Ask Jeeves]. Question answering is also provided by

the major search engines such as [MSN Search] which uses the [MSN Encarta] encyclopedia as a corpus for their question answering service.

When answering a factoid question the role of context is very important when it comes to displaying results to the user. Lin et al. [2003a,b] explore the role of context in question answering systems. In an interactive study they discovered that users prefer a chunk of text to merely the exact answer phrase or a sentence containing the answer. Regardless of the reliability of the information source, users generally prefer to have answers displayed in context. When users are researching several aspects of the same topic, increased amounts of context leads to a significant decrease in the number of questions the user asks the system. I.e., users seem to read the additional "context-text" provided by the system and use it for answering related questions.

Focused Information Access

We will now look at our focused information access task in the light of the above studies. We look at three aspects: the *target audience* for focused retrieval; the *tasks* for which focused retrieval could potentially be useful; and the potential *functionality* of our focused system.

Target audience The results of Navarro-Prieto et al. [1999] and Slone [2003] show that different user groups utilize search tools in different ways. Since our focused information access approach is a departure from the "traditional" document retrieval scenario we should—at least in the beginning—target it at experienced searchers. Experienced searchers are more likely to be able to understand and utilize the new—and presumably more powerful—search approach.

Tasks In terms of the web search taxonomies introduced by Broder [2002] and Rose and Levinson [2004] our focused information retrieval system is most likely to be useful for informational tasks. In particular, it is likely to be useful for directed searches, advice, and list compilation—where the information needs may require direct access to the most appropriate part of a relatively long document. I.e., our focused information access task is most closely related to the formal search task defined by Choo et al. [2000] and hence it targets only a relatively small portion of the whole range of search tasks.

Functionality Toms et al. [2003] conclude that in the research domain users need a quicker and more effective way to evaluate the content of retrieved documents. For our focused retrieval system this suggests that a structured overview of individual search results is potentially useful. To this end, a focused system could be useful by giving an overview of the content of different sub-parts of the documents. The results of Lin et al. [2003a,b] stress the importance of showing

relevant information in context. In our focused system we should be careful not to take the sub-document-level retrieval results out of their document context.

2.2 Information Retrieval Evaluation

In her guide to information retrieval experimentation Tague-Sutcliffe [1992] discussed two types of information retrieval evaluation frameworks, *laboratory tests* and *operational tests*. A laboratory test is one where many environmental variables are controlled, while an operational test is one where none is controlled. Laboratory tests are useful for comparing systems or individual aspects of a single system—and are thus often referred to as systems-oriented evaluation. The operational tests, on the other hand, tell us something about the usefulness of a system as a whole—and are often referred to as user-oriented evaluation.

Recall from our introductory chapter that our main research question has both a system-oriented aspect and a user oriented-aspect. Hence, in order to address our main research question we need to perform system-oriented and user-oriented evaluation. In the remainder of this section we give an introduction to each of the two types of evaluation framework.

2.2.1 Laboratory Evaluation

Information retrieval test collections provide a means to compare the effectiveness of different retrieval strategies in a laboratory setting. The most common test collections are based on the concept behind the Cranfield experiments [Cleverdon, 1967]. The Cranfield paradigm has been applied in many settings, such as, TREC [Voorhees and Harman, 2005], CLEF [Peters and Braschler, 2001], NTCIR [Kando et al., 1998], and INEX [Kazai et al., 2004b].

Ad-hoc information retrieval collections usually consist of three parts. *Documents*: A collection of documents, over which search is performed. *Topics*: Description of an information need. The information need is expressed in different formats—ranging from a short list of keywords to a verbose narrative. *Assessments*: A mapping between topics and documents indicating which documents satisfy the information need in the topic.

The Cranfield paradigm relies on a number of assumptions [Voorhees, 2002]. First, that relevance can be captured by topical similarity. Second, a single set of judgment is representative for the whole user population. Third, the relevance mapping between the topics and documents is complete.

A system is evaluated based on the ranked list of documents it produces. Effectiveness of a system can be measured in several terms of several criteria. The most basic criteria are *recall* and *precision*:

Recall measures the number of relevant documents retrieved as a portion of the total number of relevant documents.

Precision measures the number of relevant documents retrieved as a portion of the total number of documents retrieved.

Recall and precision are used to define frequently used measures:

Precision-Recall curve Precision is plotted at various recall points. The most common curve is the 11-point interpolated precision-recall average curve. The interpolated precision at recall level n is the maximum precision at any recall level $\geq n$.

Average Precision (AP) Precision is calculated for every relevant document retrieved and then averaged to get a single number for the query.

Precision@N Precision is measured at the point when N results have been retrieved. This measure is mostly used for reporting early precision, i.e., precision when 5, 10, or 20 results have been retrieved.

R-Precision Precision@R where R is the total number of relevant results for a particular query.

Bpref The number of judged non-relevant results found before a judged relevant result is found.

In order to get a stable measurement of retrieval performance the above measures are commonly averaged over a number of queries. As an example, the well known mean average precision (MAP) measure is—as the name indicates—the mean of the average precision (AP) for a number of topics.

The assumptions of the Cranfield paradigm are simplifications that are not true in practice. First, the assessments are not complete since for reasonably large collections it is practically impossible to judge each document against each query. Second, the notion of relevance is subjective and people are likely to disagree when judging relevance. It has, however, been shown that these two limitations do have a negligible impact when the test collections are used to measure the comparative performance of two systems using a single test collection [Voorhees, 2002]. The key to overcoming the limitations is to use many topics and to consider systems to be different only if the performance difference is reasonably large.

For XML element retrieval, relevance may not be captured completely by topical similarity alone. Specificity—or information granularity—is an important part of the relevance notion for XML retrieval and in the previous section we argued that the appropriate granularity of XML element retrieval results may not only rely on topical similarity, but also on the user's background knowledge or work task. It is still an open research question to determine how these issues affects the ad-hoc XML element retrieval evaluation framework (See e.g., [Kamps and Larsen, 2006, Kamps et al., 2006]).

2.2.2 Interactive Evaluation

The laboratory evaluation framework has been criticized for its failure to account for user interaction. Interactive information retrieval evaluation studies the interaction between searchers and retrieval systems and compliments the laboratory evaluation framework [Beaulieu et al., 1996, Borlund, 2003].

Interactive retrieval has been a part of TREC from its early days [Beaulieu et al., 1996, Dumais and Belkin, 2005]. Over [2001] gives an overview of interactive retrieval at TREC 1–8. The interactive track has developed four focal points (from [Over, 2001, page 369]):

- "the searcher in interaction with the system,
- behavioral details, the process, and interim results not just summary measures of final result,
- isolation of the effects of system, topic, searcher, and their interactions,
- evaluation of the evaluation methodology."

In the first TREC years, interactive systems were compared against automatic systems, but later the focus changed to comparing interactive systems among themselves. The track collects data both on user satisfaction and the search process, including video, think-aloud audio, and system interaction logs. In TREC 1–8 the data was collected by assigning subjects a description of an information need and asking them to find as many relevant documents as possible within a given time period. The interactive track did not address any central research questions, but served as an experimental framework where participants could address their own questions. The participants did however share tasks, topics, documents, and assessments.

The interactive track at TREC-9 explored fact-finding tasks [Hersh and Over, 2001]. Users were asked to answer a series of questions. Each question called for very short answers. The questions either called for a list of answers—e.g., name four films in which Orson Wells appeared—or to compare two facts—e.g., is Denmark larger or smaller in population than Norway? There was no centralized research agenda for the track and each participating group used their own system and followed their own research agenda.

One of the issues investigated by Belkin et al. [2001]—at the TREC-9 interactive track—was to investigate whether users preferred the document display to either begin at the beginning of a document or at the best passage. They did not find a significant difference between the two approaches. The comparison was, however, not direct due to a lack of resources. This result should therefore not discourage further sub-document-level approaches.

The interactive track at TREC 2001 and TREC 2002 explored interactive Web search in one two-year cycle [Hersh and Over, 2002, Hersh, 2003]. The paper by Toms et al. [2003] discussed in the previous section is based on this two-year cycle.

Table 2.1: Key figures of the INEX 2002–2005 collections

	Articles	Elements	Avg. depth	Size
INEX 2002	12,107	8,222,075	5.96	513 MB
INEX 2005	16,819	11,411,134	5.97	764 MB

The INEX initiative also includes an interactive track, which we will discuss in Section 2.4.

2.3 The INEX Ad-hoc Test Collection

XML element retrieval is the core task addressed as part of the INEX initiative [Kazai et al., 2004b]. The main aim of the exercise is to find focused pieces of text which satisfy users' information needs. The task is to find elements which are exhaustive in the sense that they fully discuss the user's information need, but at the same time they must be specific in the sense that they discuss little other than the user's information need. From the systems perspective the task is to provide a ranked list of XML elements. I.e.,

> "instead of retrieving whole documents, systems aim at retrieving document components (e.g., XML elements of varying granularity) that fulfill the user's query" [Kazai et al., 2004b]

In this section we introduce the building blocks of the INEX XML retrieval test collection. We discuss the INEX document collection (2.3.1), the tasks (2.3.2), the topics (2.3.3), the relevance assessments (2.3.4), and, finally, the metrics (2.3.5).

2.3.1 Document Collection

INEX 2002–2005 used a collection of full-text computer science articles donated by the IEEE Computer Society. The articles are marked up in XML format and originate from over 20 IEEE magazines and transactions.

The original document collection—used at INEX 2002–2004—contains 12,107 articles from the period 1995–2002. In 2005 the collection was extended with additional 4,712 IEEE Computer Society articles from the period 2002–2005. Table 2.1 shows some statistics of the IEEE collections. The experiments in this thesis are exclusively based on the INEX 2002 document collection. Consequently the assessments of the INEX 2005 test collection have been modified by removing the assessments of documents outside the INEX 2002 document collection. The results for the 2005 test collection—reported in this thesis—are thus not directly comparable with results that use the full 2005 document collection.

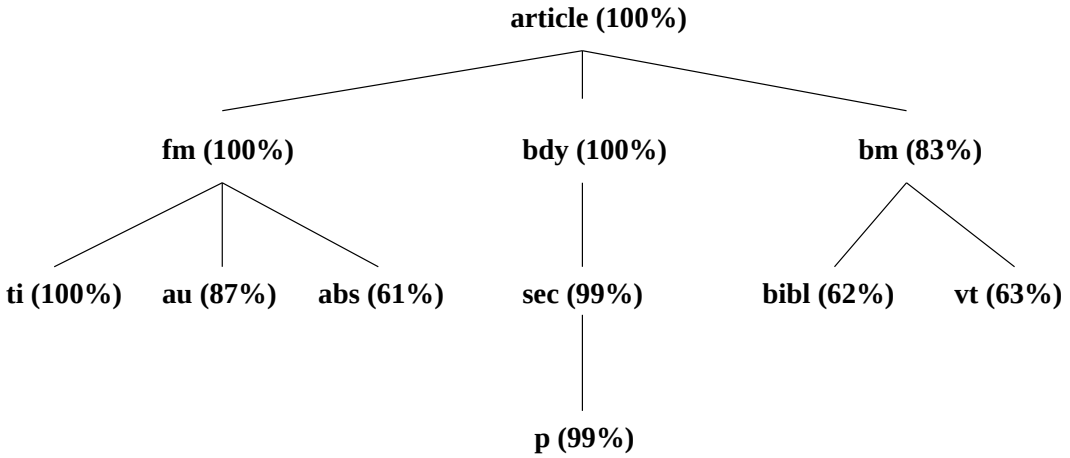


Figure 2.2: Simplified figure of the structure of a typical IEEE document. All documents have a root called *article*. At the next level, all documents are divided into *front matter (fm)* and *body (bdy)*. In addition, 83% of the documents have a *back matter (bm)* part. In the front matter part, all documents have a *title (ti)*, 87% have an *author (au)*, and 61% have an *abstract (abs)*. In the body part, almost all documents are divided into a number of *sections (sec)* and *paragraphs (p)*. In the back matter, 62% have a *bibliography (bibl)* and 63% have a *vitae (vt)*.

The collection is based on fairly complex markup using 176 different tag-names (in total the DTD contains 192 content types). Figure 2.2 shows a simplified structure of a part of a "typical" document.

2.3.2 Tasks

The main task of INEX from 2002 to 2005 is adhoc XML element retrieval:

XML element retrieval For each query, produce a ranked list of document components (a.k.a. elements) which satisfy the user's information need.

In 2002 to 2004 this task was divided into two sub-tasks based on the types of queries used. Each sub-task used a different set of queries.

Content-Only (CO) queries Plain keyword queries.

Content-And-Structure (CAS) queries Queries which are a mixture of content and structural constraints. The structure is used in two ways. First, it can be used to constrain the type of the target elements (target constraints). Second, it can be used to constrain the context in which the target element reside (context constraints).

The experiments in this thesis are exclusively based on the content-only queries.

In 2005 three types of element retrieval tasks were introduced. The tasks were defined as follows (from [Malik et al., 2006, pp. 8–9]).

Table 2.2: Statistics on the CO topics sets used in the experiments. The table shows the number of topics, average length of queries (no. of terms), standard deviation (in brackets), and median query length. Aggregation is done over three query formats: title-only, description-only and combination of title and description fields.

	Title			Description			Title+Description		
	Avg. len.		Median	Avg. len.		Median	Avg. len.		Median
2002	4.4	(2.2)	4.0	16.1	(10.4)	14.0	20.5	(10.8)	18.5
2003	4.8	(2.8)	4.0	13.3	(7.8)	10.5	18.0	(9.5)	16.0
2004	5.3	(2.4)	5.0	23.2	(17.9)	17.0	28.5	(19.1)	22.0
2005	4.6	(2.3)	4.0	17.5	(7.2)	16.5	22.1	(8.2)	21.0

Thorough "The aim here was for systems to find all relevant elements within the collection."

Focused "The aim was for systems to find the most exhaustive and specific element on a path ... and return to the user only this most appropriate unit of retrieval."

FetchBrowse "The aim of the fetch and browse retrieval strategy was to first identify relevant articles (the fetching phase), and then to identify the most exhaustive and specific elements within the fetched articles."

The thorough task can be seen as as the continuation of the main task of the previous years. In the experiments in this thesis we will mainly look at the thorough task. Our view of XML retrieval in this thesis is systems-oriented. I.e., we want to use an XML retrieval as a back-end retrieval engine. We believe the thorough task is best suited of evaluating such a system since it is not based on any specific end-usage assumptions. Our main evaluation in this thesis will thus be based on the focused task using "official" INEX metrics (see Section 2.3.5). However, in Chapter 7 we will evaluate the thorough task with respect to the two issues which the focused and FetchBrowse tasks address, namely, overlap and document ranking.

2.3.3 Topics

In the experiments in this thesis we will only use the so-called content-only (CO) topics of the INEX collection. The INEX topics contain three versions of the underlying information needs (a.k.a. fields)—using different amounts of verbosity. In our experiments we use two of those fields to create three different query-types. First, we use only the title field. The *title* field describes the information need of the topic as a list of content keywords. The query format is meant to resemble searches using state-of-the-art Internet search engines. Second, we use only the

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE inex_topic SYSTEM "topic.dtd">
<inex_topic topic_id="104" query_type="CO" ct_no="52">
  <title>Toy Story</title>
  <description>Find information on the making of the animated movie
    Toy Story, discussing the used techniques, software, or hardware
    platforms. </description>
  <narrative>To be relevant, a document/component must discuss some
    detail of the techniques or computer infrastructure used in the
    creation of the first entirely computer-animated feature-length
    movie, "Toy Story." </narrative>
  <keywords>Pixar, Disney, Buzz Lightyear, 3D-rendering, Sun,
    SGI</keywords>
</inex_topic>
```

(a) INEX topic 104

```
Toy Story
```

(b) Title query

```
Find information on the making of the animated movie Toy Story
discussing the used techniques software or hardware platforms
```

(c) Description query

```
Toy Story Find information on the making of the animated movie Toy
Story discussing the used techniques software or hardware platforms
```

(d) Title+description query

Figure 2.3: Example of an INEX topic (topic 104).

description field. The *description* field is a one or two sentence description of the information need, written in natural language. Third, we use a combination of the title and description fields. Figure 2.3 shows an example of an INEX topic (part (a)) and shows how we use it to create different query formats (parts (b)–(d)).

Table 2.2 shows some statistics about the length of the INEX content-only topic sets. The table shows the number of topics and the mean and median length of the queries. We report statistics for each of the query formats. We see that the length of the topic titles does not change much from one year to another. The length of the descriptions is, however, more varied.

2.3.4 Assessments

INEX relevance assessments have always been done using a two dimensional, graded scale. The scale has changed slightly from one year to another. More importantly, though, the assessment interface also changed between years.

INEX 2002

In 2002 the two dimensions were named *topical relevance* and *component coverage*. The dimensions are defined as follows (from [Gövert and Kazai, 2003, pp. 8–9]):

"Topical relevance, which reflects the extent to which the information contained in a document component satisfies the information need."

"Component coverage, which reflects the extent to which a document component is focused on the information need, while being an informative unit."

Topical relevance was assessed using a 4-point scale:

> "**Irrelevant (0):** The document component does not contain any information about the topic of request. **Marginally relevant (1):** The document component mentions the topic of request, but only in passing. **Fairly relevant (2):** The document component contains more information than the topic description, but this information is not exhaustive. In the case of multi-faceted topics, only some of the sub-themes or viewpoints are discussed. **Highly relevant (3):** The document component discusses the topic of request exhaustively. In the case of multi-faceted topics, all or most sub-themes or viewpoints are discussed."

Component coverage was assessed using a 4-point scale:

> "**No coverage (N):** The topic or an aspect of the topic is not a theme of the document component. **Too large (L):** The topic or an aspect of the topic is only a minor theme of the document component. **Too small (S):** The topic or an aspect of the topic is the main or only theme of the document component, but the component is too small to act as a meaningful unit of information. **Exact coverage (E):** The topic or an aspect of the topic is the main or only theme of the document component, and the component acts as a meaningful unit of information."

The assessment interface displayed each document in XML format and for each element there was a text-box where the assessor would assign relevance labels (e.g. 3E, 3L, 2S, etc.). We refer to [Gövert and Kazai, 2003] for more detailed information on the assessment system and screen-shots of the assessment interface.

INEX 2003 and 2004

In 2003 and 2004 the two relevance dimensions were renamed *exhaustivity* and *specificity*. The dimensions are defined as follows (from [Malik et al., 2005, pp. 8–9]).

> "**Exhaustivity (e),** which describes the extent to which the document component discusses the topic of request."

> "**Specificity (s),** which describes the extent to which the document component focuses on the topic of request."

Exhaustivity was assessed using a 4-point graded scale:

> "**Not exhaustive (e0):** the document component does not discuss the topic of request at all; **Marginally exhaustive (e1):** the document component discusses only few aspects of the topic of request; **Fairly exhaustive (e2):** the document component discusses many aspects of the topic of request; and **Highly exhaustive (e3):** the document component discusses most of all aspects of the topic of request."

Specificity was assessed using a 4-point graded scale:

> "**Not specific (s0):** the topic of request is not a theme of the document component; **Marginally specific (s1):** the topic of request is a minor theme of the document component [...]; **Fairly specific (s2):** the topic of request is a major theme of the document component [...]; **Highly specific (s3):** the topic of request is the only theme of the document component."

In 2003 the assessment interface was changed to make the assessment procedure easier. The text was not shown in raw XML format. The articles were made more "eye-friendly" using style-sheets. The tag-names were, however, shown for assessment purposes. Assessors assigned relevance labels to elements using a graphical interface. In 2004 the interface went through a minor modification. The most noteworthy difference in 2004 was that assessors could assess a group of elements simultaneously. In both years the interface used inference mechanisms to ensure the consistency of assessments [Piwowarski and Lalmas, 2004]. The inference was based on a set of rules that could not be violated—e.g., "An XML element cannot be more exhaustive than its parent element," "specificity cannot increase when going from (all) children to parent elements," etc.

Table 2.3: Number of assessed CO topics for each year of INEX. The table shows the total number of topics, total number of assessed topics, and the number of topics having a strictly assessed element.

Year	Topics	Assessed	Strict
2002	30	24	23
2003	36	32	27
2004	40	34	25
2005	40	29	26

INEX 2005

In 2005 the definition and naming of the two dimensions was unchanged. The assessment procedure did however undergo a fundamental change. Assessors no longer needed to assign to each element a two-dimensional relevance label. Instead, the assessor assessed each article in a two step process. First, she highlighted all text fragments which contained only relevant information. In the second step she assigned a exhaustiveness value to each element which contained some highlighted parts (see further in [Malik et al., 2006, page 11]). Exhaustiveness was assessed on a 3(+1)-point graded scale (from [Malik et al., 2006, page 10]):

> "highly exhaustive ($e = 2$), somewhat exhaustive ($e = 1$), not exhaustive ($e = 0$) and "too small" ($e = ?$)."

The specificity values of elements were derived from the highlighting ($s \in [0, 1]$). Fully highlighted elements are considered fully specific ($s = 1$). The specificity values for other elements are determined using a function based on the ratio between relevant content and all content.

2.3.5 Metrics

We now turn our attention to the INEX metrics. We will not review the plethora of metrics which have been proposed within the INEX community [Kazai et al., 2004a, de Vries et al., 2004, Piwowarski and Dupret, 2006, Kazai and Lalmas, 2006]. Instead, we limit ourselves to the metrics that will be used for evaluation in this thesis. For a more complete metrics discussion we refer to an overview of the INEX 2005 metrics by Kazai and Lalmas [2006].

Over the past few years the INEX initiative has made considerable progress toward the building of a reliable XML element retrieval evaluation collection. However, XML element retrieval evaluation is not yet considered to be a solved problem. This has some implications for this thesis. We cannot simply apply *the* evaluation framework to evaluate our system. Instead we choose a couple of

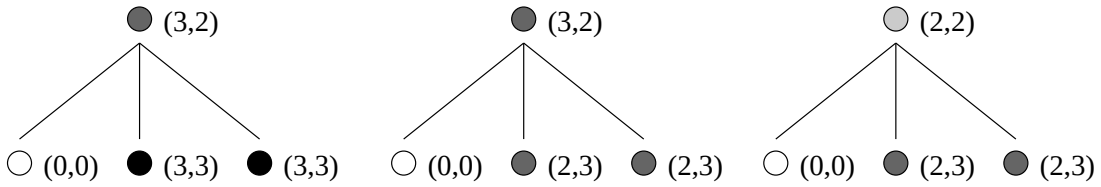


Figure 2.4: Example of three assessment scenarios. Each element is marked with an assessment value of the form (e,s) where e stands for exhaustiveness and s stands for specificity. The values are assigned according to the INEX 2003–2004 assessment scheme (See Section 2.3).

plausible frameworks and evaluate our system for each of them. In this section we motivate our choice.

We use a number of metrics to evaluate our experimental results. We use both "traditional" information retrieval metrics from the `trec_eval` package [trec_eval] and metrics from the EvalJ package [EvalJ] which are specially tailored at the INEX tasks. Before we describe the metrics we use, let us recall the task we want to evaluate.

XML element retrieval For each element in the collection estimate how relevant it is for the user's information need. This process is approximated by creating a ranked list of XML element for each user query. The elements are ranked by decreasing likelihood of being relevant for the user's information need.

In terms of the INEX assessments, XML element retrieval is defined as the task of creating a list of elements, ranked by decreasing likelihood of the element being *highly exhaustive* and *highly specific*. The straightforward evaluation of this task is to consider as relevant precisely those elements that are assessed both highly exhaustive and highly specific (this mapping is referred to as the *strict quantization* and is defined below). For each topic and each element we have a boolean judgment whether the element is highly exhaustive and highly specific. Using the boolean judgments we can use the `trec_eval` program to report scores using a number of metrics.

Using the strict quantization has some drawbacks. The main one being that it is too strict. I.e., it neither rewards near-misses, nor handles the case when no highly exhaustive and highly specific element exists. Consider the assessments in Figure 2.4. For the left-most example, one can argue that strict evaluation is sufficient, i.e., a system gets a perfect score if and only if it returns the two highly exhaustive and highly specific elements. Others may argue that a more lenient evaluation is desired, i.e., a system that returns the parent element should be rewarded with a partial score. For the middle example (in Figure 2.4), the strict evaluation would not reward the return of any of the elements, even if the information need is sufficiently answered by the parent element, i.e., that

element is highly exhaustive. One might thus argue that a partial score should be rewarded for the retrieval of the three partially relevant elements. For the rightmost example (in Figure 2.4), one may apply similar arguments for justifying the reward of partial scores. The reward may, however, be challenged by the fact that the information contained in the elements does not fully satisfy the user's information need.

The drawback of the strict assessments has been addressed within the INEX initiative by introducing so-called *generalized quantizations* (defined below). The generalized quantizations map the two-dimensional relevance scale to a one-dimensional graded graded relevance scale. There are two types of generalized quantizations used, exhaustiveness-oriented and specificity-oriented quantizations. The two types favor, respectively, the retrieval of exhaustive and specific elements.

Just as the strict quantization can be criticized for being too strict, the generalized quantizations can be criticized for being too lenient. The generalized quantizations, do reward partially relevant elements, independent of whether they contribute to a highly exhaustive ancestor element. E.g., in terms of Figure 2.4, the retrieval of the two $(2, 3)$ elements in the middle example would be rewarded equally as the retrieval of the two $(2, 3)$ elements in the rightmost example. However, the combination of the elements from the middle example are highly exhaustive, while the combination of the elements form the rightmost example is not.

Below we introduce the metrics we will use for evaluation in this thesis. First, we review the quantization functions—functions for mapping multi-dimensional relevance assessments to a single number.

Quantization

Quantization is a mapping from the relevance assessments to a single number:

$$f_{quant} : Assessment \rightarrow [0, 1] \tag{2.1}$$

In INEX 2002 two quantization functions were defined.

$$f_{strict} := \begin{cases} 1 & \text{if } (rel, cov) = 3E \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

$$f_{generalized} := \begin{cases} 1.00 & \text{if } (rel, cov) = 3E, \\ 0.75 & \text{if } (rel, cov) \in \{2E, 3L\}, \\ 0.50 & \text{if } (rel, cov) \in \{1E, 2L, 2S\}, \\ 0.25 & \text{if } (rel, cov) \in \{1S, 1L\}, \\ 0.00 & \text{if } (rel, cov) = 0N \end{cases} \tag{2.3}$$

In INEX 2003 the quantization functions were updated to reflect the changed assessment framework:

$$f_{strict} := \begin{cases} 1 & \text{if } (e,s) = (3,3) \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

$$f_{generalized} := \begin{cases} 1.00 & \text{if } (e,s) = (3,3), \\ 0.75 & \text{if } (e,s) \in \{(2,3),(3,\{2,1\})\}, \\ 0.50 & \text{if } (e,s) \in \{(1,3),(2,\{2,1\})\}, \\ 0.25 & \text{if } (e,s) \in \{(1,\{2,1\})\}, \\ 0.00 & \text{if } (e,s) = (0,0) \end{cases} \tag{2.5}$$

The generalized quantization function was questioned for preferring exhaustiveness over specificity [Kazai et al., 2004a]. In INEX 2004 a new generalized quantization function was introduced. This function preferred specificity over exhaustiveness:

$$f_{sog} := \begin{cases} 1.00 & \text{if } (e,s) = (3,3), \\ 0.90 & \text{if } (e,s) = (2,3), \\ 0.75 & \text{if } (e,s) \in \{(1,3),(3,2)\}, \\ 0.50 & \text{if } (e,s) = (2,2) \\ 0.25 & \text{if } (e,s) \in \{(1,2),(3,1)\}, \\ 0.10 & \text{if } (e,s) \in \{(2,1),(1,1)\}, \\ 0.00 & \text{if } (e,s) = (0,0) \end{cases} \tag{2.6}$$

In 2005 the quantization functions were again updated to reflect the changes in the assessment format:

$$f_{strict} := \begin{cases} 1 & \text{if } e = 2 \text{ and } s = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

$$f_{gen}(e,s) := e \cdot s. \tag{2.8}$$

Note that the number of topics used in the evaluation depends on the quantization being used. Not all topics have a strictly relevant element. These topics will thus be considered as having no relevant result when the strict quantization is used and will thus not contribute to the evaluation. Table 2.3 shows some statistics on the number of assessed topics for the different vintage of the INEX collection.

In this thesis we will devote special attention to the strict assessments. Table 2.4 shows the number of elements assessed strictly relevant for the different vintages of the INEX test collection. We see that the mean and the median number of articles having a strict assessment has decreased as time goes by. This is also true for the median number of elements assessed strictly relevant. The average number of elements assessed strictly relevant also decreases over the years, except in 2004 where we had a single outlier topic with 848 elements assessed strictly relevant.

Table 2.4: Statistics on the number of strict assessments in the INEX collections. The table shows aggregated results over different topics. (Left): min, max, mean and median number of strictly assessed elements per topic. (Right): min, max, mean, and median number of articles containing a strictly assessed element.

		Elements				Articles			
	topics	min	max	mean	median	min	max	mean	median
2002	23	2	210	60.7	36	1	108	27.4	17
2003	27	3	175	55.1	34	1	75	14.1	9
2004	25	2	848	103.6	18	1	51	11.2	6
2005	26	1	163	34.0	9	1	70	10.0	3.5
Total	101	1	848	62.9	28	1	108	15.33	9

Metrics

The INEX metric we will use in this thesis is the mean average effort-precision (MAep) metric [Kazai and Lalmas, 2006]—a systems-oriented retrieval metric which belongs to the family of metrics called eXtended Cumulative Gain (XCG). The MAep measure is similar to the mean average precision (MAP) metric described in Section 2.2.1, the main difference being that the MAep can handle graded relevance—i.e., the non-boolean output of the quantization functions. For metric details we refer to [Kazai and Lalmas, 2006].

XML retrieval evaluation is subject to the so-called "overlap problem" [Kazai et al., 2004a]. I.e., since the elements assessments may be overlapping, the retrieval of same information is rewarded multiple times. Depending on how element results are presented to users overlap may indeed be a problem [Tombros et al., 2005b], or be considered a feature [Kamps and Sigurbjörnsson, 2006, Betsi et al., 2006]. It is thus debatable whether or not retrieval of overlapping information should be rewarded by the evaluation metric. The XCG metrics come in two flavors—rewarding or not-rewarding the retrieval of overlapping information. In this thesis we are looking at the element retrieval task from a systems' perspective and are therefore not concerned with the "overlap problem." We will thus use—for the main evaluation in this thesis—the metric which does reward retrieval of overlapping relevant information. However, we will discuss the effect of overlap in Section 7.2 where we evaluate our system using the metric which does not reward overlap; and in Chapter 8 where we put element retrieval into action in an interface for giving focused information access.

2.4 INEX iTrack

Interactive XML retrieval has been studied at INEX since 2004 [Tombros et al., 2005a, Larsen et al., 2006a]. The main aim of the task has been to study the

behavior of searchers when interacting with components of XML documents. The main focus of the two years has been on observing users interacting with an XML element retrieval system.

In 2004, element retrieval results were presented to the user as a ranked list of elements. The main outcome of the evaluation was that users were frustrated by elements from the same document appearing at different rank in the result list [Tombros et al., 2005b].

In 2005, element retrieval results were presented to the user as a ranked list of documents, with additional links to relevant elements within the documents. Initial results show that users predominantly access the relevant documents at the beginning of the document [Larsen et al., 2006b].

A further discussion of the INEX interactive track results will be postponed until Chapter 8 where we discuss how element retrieval can be put into action as part of an operational system.

2.5 Experimental Setup of this Thesis

The evaluation in this thesis will be of two sorts:

Laboratory evaluation We evaluate different retrieval approaches for performing the INEX systems-oriented—thorough—XML element retrieval task (Chapters 3–7).

Interactive evaluation In two interactive experiments we evaluate a system which gives focused information access using XML element retrieval engine as a back-end (Chapter 8).

Below is a summary of important choices we made in our evaluation framework in this thesis.

Metrics We choose to evaluate our system using three different quantizations, strict, generalized, and specificity-oriented generalized. For each quantization we choose a mean-average-precision-like metric

- Mean Average Precision (MAP) for the strict quantization, and
- Mean Average effort-precision (MAep) for the generalized quantizations.

Document collection We choose to use the same document collection for all experiments in the thesis—i.e., the INEX 2002 document collection. This means that for the 2005 topics there are assessments which fall outside the document collection used in this thesis. Consequently, we have pruned the 2005 assessments by removing all assessments that fall outside the INEX 2002 document collection. This means that our results for the 2005 topics are not directly comparable with the official INEX results.

Document Retrieval Comparison A comparison between element retrieval and document retrieval is an important part of the experiments in Chapter 3. For this purpose we perform some document retrieval experiments on the INEX collection. For this evaluation we need a document-level relevance assessments. There are several ways in which such assessments can be derived from the INEX element-level assessments. For document retrieval, exhaustiveness is the most important criteria and specificity does not play an important role in the task. Hence we define document relevance as follows:

- A document is relevant if and only if it has an element which is assessed as highly exhaustive.

Note that we evaluate only over topics having at least one element assessed highly exhaustive. This means that the topic set over which article retrieval is evaluated may differ slightly from the topic set used for strict and generalized element evaluation. The relation is as follows:

$$T_{strict} \subseteq T_{exhaustive} \subseteq T_{assessed}$$

where T_{strict} denotes the set of topics having an highly exhaustive and highly specific element; $T_{exhaustive}$ denotes the set of topics having a highly exhaustive element; and $T_{assessed}$ denotes the set of assessed topics. For document retrieval we measures effectiveness using `trec_eval` metrics.

Significance Testing We test statistical significance of our results by using the two-tailed t-test for paired data. I.e., we calculate the probability that the actual mean difference between the pairs is zero. If this probability is low we can claim that the difference between the pairs is significant. We distinguish between three levels of significance: $P < .05$ is marked with *, $P < .01$ is marked with ** and $P < .001$ is marked with ***. Our choice of significance test is based on the work of Sanderson and Zobel [2005], who found the t-test to be highly reliable for document retrieval. They also found the t-test to be more reliable than the sign-test and the Wilcoxon test. In this thesis, we assume that this finding also holds for the element retrieval task.

2.6 Conclusions

In this chapter we have provided background material on information search behavior literature and we have located our focused information access task relative to the literature. We also gave some background on information retrieval evaluation methodology and introduced the INEX evaluation collection which will be the main means of evaluation for the retrieval methods studied in this thesis. Finally, we have outlined the experimental setup used in this thesis.

In the following five chapters—Chapters 3–7—we model, implement, and evaluate an XML element retrieval engine. In Chapter 8 we build an interface for giving focused information access, using our XML element retrieval engine as a back-end. Finally, in Chapter 9 we conclude.

Chapter 3

A Baseline Element Retrieval System

Before we dive into the development of an advanced element retrieval system we should take stock of the document retrieval tools we already have. We can use these tools to build a simple baseline system on which we can later base our more advanced element retrieval system. In this chapter we discuss such a baseline element retrieval system. We take a document retrieval system, modify it slightly and apply it to the task of retrieving elements. The main question we want to answer in this chapter is:

How is element retrieval different from document retrieval?

We apply our baseline system to both the element retrieval and document retrieval task and study the difference in retrieval performance when we alter the parameter settings. The differences give us "hints" about the differences between the two tasks and indicate which aspects of the document retrieval system we need to change to make it more applicable to the element retrieval task. Additionally, by applying a document retrieval system to the element retrieval task we get a milestone against which we can compare our more specialized element retrieval methods developed later in this thesis.

In addition to compare document retrieval and element retrieval we explain our baseline retrieval approach and discuss related work. We start by introducing related work on retrieval systems in Section 3.1. In particular we discuss sub-document-level retrieval system, both developed to solve passage retrieval tasks and XML element retrieval tasks.

In Section 3.2 we discuss the indexing of both text and XML structure. We discuss the indexing schemes that are relevant for our baseline system. This indexing scheme serves as a starting point for specialized indexing approaches introduced in Chapter 5.

Our retrieval model is discussed in Section 3.3. In this thesis we will focus on the application of language models to focused retrieval. We introduce our baseline language model framework. This baseline system will serve as a starting point for further extensions in Chapters 4 and 6.

In Section 3.4 we show the results of applying our baseline language model framework to the element retrieval task. We tune the main model parameter, the smoothing parameter, and compare the optimal settings for the element retrieval task to optimal settings for the task of retrieving documents. The results of the parameter tuning gives us valuable information on crucial differences between the two tasks. We discuss our evaluation results further in Section 3.5 and conclude in Section 3.6.

3.1 Related Work on XML Retrieval Systems

In this section we review related work on XML retrieval systems. We do not give a detailed background on retrieval systems; for a general background we refer to textbooks by Grossman and Frieder [2004] and Witten et al. [1999]. Instead, our discussion of focused retrieval is divided up into three parts. First, we look at earlier work on passage retrieval systems. Second, we look at focused retrieval in the context of semi-structured documents. Third, we review recent developments in XML element retrieval systems. Our XML retrieval discussion will be centered mainly around systems developed by the INEX community.

3.1.1 Passage Retrieval

Salton et al. [1993] investigate approaches to sentence similarity and passage retrieval. The approaches considered are motivated by two types of benefits: first, efficiency, "users are no longer faced with large masses of retrieved materials;" and second, effectiveness, "relevant short texts are generally more easily retrievable than longer ones." They provide experiments on an encyclopedia which show that section and paragraph retrieval can improve retrieval performance. Their conclusion is that the local match (sentence similarity) acts as a precision device, and that the passage retrieval acts as a recall device.

Callan [1994] investigates two types of passage retrieval approaches. The approaches differ in the way passages are defined. One approach uses paragraphs as passage units, and the other uses variable-length text windows. Passages are ranked using either passages alone, or in combination with document-level evidence. Callan evaluates the passage retrieval approaches using a document retrieval test collection. I.e., passage retrieval results are used to rank documents. The main outcome of the evaluation is that the variable-length text windows outperformed the paragraph-level approach. Callan suggests that the poor performance of the paragraph-level approach may be due to inconsistent paragraph division of different authors, or due to the passage size—i.e., paragraphs being too small units. Another outcome of the evaluation is that the combination of passage- and document-level evidence proved to be more effective than using passage-level evidence alone.

Kaszkiel and Zobel [1997] experiment with several passage retrieval approaches: using fixed-length passages, variable-length passages, discourse structures (e.g., sections and paragraphs), and text tiles [Hearst and Plaunt, 1993]. The passage retrieval approaches were evaluated using a document retrieval collection. I.e., the task is to use passage-level evidence to retrieve documents. The main conclusion is that fixed-length passages proved the most robust approach. Using discourse structures to determine passage boundaries did not prove to be effective.

Liu and Croft [2002] apply several language models on the problem of using passage retrieval to improve document retrieval. They compare generative language models and relevance models. The main conclusion is that passage retrieval can significantly outperform document retrieval on collections where documents are long and can span several topics, such as the Federal Register (TREC discs 1 and 2).

In sum, passage retrieval has proven a useful method for ranking documents—in particular when the documents are long. Fixed-length passages seem to be more effective than discourse structures.

3.1.2 Semi-structured Retrieval

Wilkinson [1994] explores ways to retrieve from structured documents. He uses the structured features of a subset of the TREC 1994 collection. He does both document retrieval and "XML" retrieval proper, where relevance was judged at the element level (in-house assessment process). Wilkinson raises four research questions

1. Can element retrieval help document retrieval?
2. Can we combine document and element retrieval to boost document retrieval?
3. Can document retrieval serves as a basis for element retrieval?
4. Can we combine document and element retrieval to boost element retrieval?

He shows that element retrieval can indeed help document retrieval. He also shows that the other way around is not effective, that is, scoring documents and then picking out elements is not a good strategy. The most important lesson learned is that the mixture of local and global evidence is the most effective strategy for both document and element retrieval.

Myaeng et al. [1998] introduce a flexible model for the retrieval of SGML documents. Their model goes below the document level and retrieves parts of SGML documents. The choice of appropriate unit of retrieval is left to the user as a structural requirement. Furthermore, the model caters for the use of additional structural constraints. Their retrieval model is based on inference networks. In

their experiments they try to answer three questions. First of all, they test whether the system can handle a variety of structural queries. Second, they test if element-based passage retrieval can help document retrieval. Third, they test whether appropriate element bias can help document retrieval. The reason they chose to evaluate document retrieval is the fact that there was no element retrieval test-collection available at the time. They use the TREC patent collection. Their results are positive: both element-based passage retrieval and careful assignment of element-type bias (weight) can help document retrieval

3.1.3 XML Retrieval

Prior to the INEX initiative there were two SIGIR workshops on XML and information retrieval [Carmel et al., 2000, Baeza-Yates et al., 2002]. Both workshops focused mostly on processing structured queries and are thus beyond the main scope of this thesis. We will however review two noteworthy contributions.

Fuhr et al. [Fuhr and Großjohann, 2001, Fuhr et al., 2003b] introduced a query language for information retrieval from semi-structured documents. They took the exact-match query language XQL and extended it with features from information retrieval such as term weighting, relevance ordered search, and vague predicates. The language allowed both for vague interpretation of content and structural constraints.

Carmel et al. [2002, 2003] suggested querying XML documents with XML fragments. The intuition behind XML fragments is to introduce a query language where users can easily express their information needs both in terms of content and structure constraints. The main difference between XML fragments and other XML query languages is that XML fragments are end-user oriented while most other XML query languages are a kind of "programming languages" to be used by XML retrieval application developers.

Since the advent of the INEX initiative a range of different approaches have been applied to the element retrieval task. Mass and Mandelbrod [2004, 2005] extend the vector space model [Salton et al., 1975] so that it can be applied to retrieving XML elements; Clarke and Tilker [2005] and Lu et al. [2006] apply the Okapi BM25 algorithm; and Geva [2005] uses a simple tf-idf heuristics. In the remainder of this section we will concentrate on approaches that are most related to our own—i.e., the application of language models to the XML retrieval task.

Within the INEX initiative there are three main groups which apply language models to the XML element retrieval task [Sigurbjörnsson et al., 2004a, List et al., 2004, Ogilvie and Callan, 2004]. Although the groups obviously have a lot in common they complement each other by addressing the task from different angles, respectively, an empirical angle, a database angle, and a modeling angle.

The TIJAH system is an XML retrieval system built on top of a relational database system [List et al., 2004, Mihajlović et al., 2005, List et al., 2005]. The system adheres to the three-level database architecture consisting of a conceptual

level, logical level, and a physical level. Language models are implemented at the conceptual level, they are translated into probabilistic region algebra expressions in the logical level, and executed in the physical level using the MonetDB database kernel. This layered approach allows for the exploration of a wide range of research issues ranging from the modeling of relevance at the conceptual level to the optimized query execution plans at the physical level.

Ogilvie and Callan [2004, 2005, 2006] introduce hierarchical language models for ranking XML elements. In their basic approach, the language model of leaf elements are estimated using the leafs' text. The language models of internal elements is estimated using a linear combination of elements' own text and the language models of the elements' children [Ogilvie and Callan, 2004]. In their extended approach, they bring element context into the ranking by adding a language model for the elements' parent into the linear interpolation [Ogilvie and Callan, 2005]. Interpolation parameters are estimated using a generalized expectation maximization algorithm [Ogilvie and Callan, 2006].

Our own application of language models to XML element retrieval has been driven by empirical analysis [Sigurbjörnsson et al., 2004a, 2005, Sigurbjörnsson and Kamps, 2006]. We gradually build our approach up from a simple baseline system to a system which gradually takes more of the document structure into account. Our approach will be discussed rigorously in the remainder of this thesis.

3.2 Indexing

We start describing our baseline XML retrieval system by describing our indexing structures. Figure 3.1 shows an overview of our system architecture. We build two types of indices: *text indices* and a *structure index*. The text is indexed using an in-house extension [ILPS-Lucene] of the Lucene retrieval engine [Lucene] and the structure is indexed using a relational database. Separating text indexing from the structure index has two main benefits. First, it gives a flexible experimental framework for testing different text indexing strategies. We can plug in different text indices without having to rebuild our structural index. Second, we can use our existing document retrieval system on our various text indices.

The indexing is managed by the document pre-processor (bottom of Figure 3.1). The pre-processor parses each document using an XML parser and extracts both information of the structure and text. The document structure is stored in the structure index, and the text is passed to the Lucene indexer. Tokenization, stop-word removal, and stemming is left to Lucene. Although we separate text and structure in the preprocessor, we use the same element/document identifiers in both the text indices and the structure index. We can thus easily join the two indices in our retrieval module.

In our baseline system, we build two types of inverted text indices, one for

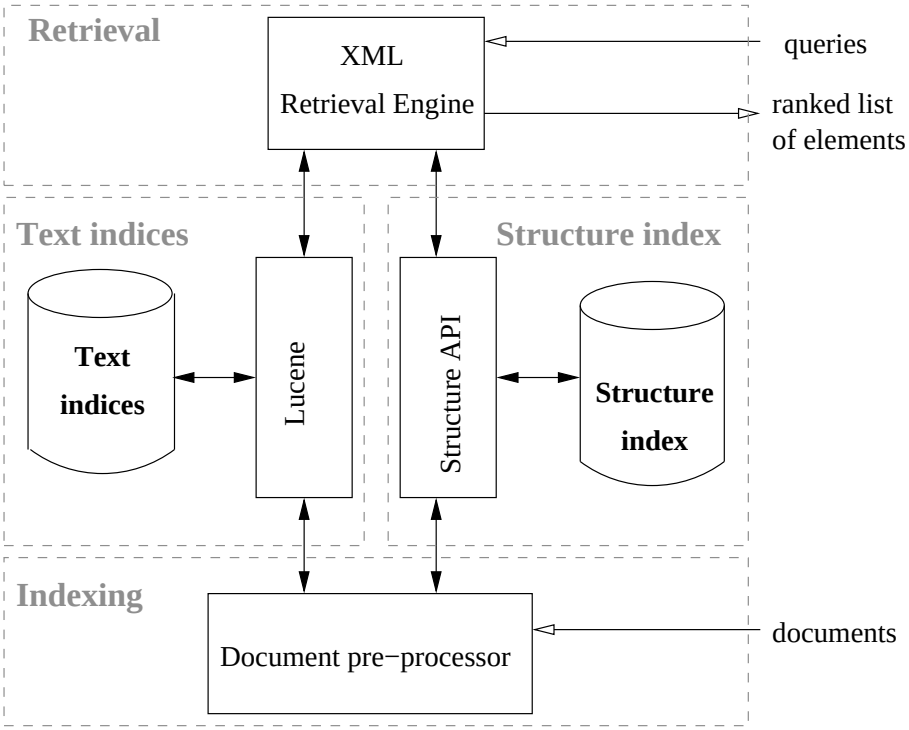


Figure 3.1: Overview of our XML retrieval system architecture. This figure can be seen as a zoom in on the "element retrieval engine" in Figure 1.4.

individual XML elements and another for complete XML documents—in our case each XML document corresponds to one journal article. In our baseline system in this chapter, the document index will be used to compare the performance of retrieving elements to retrieving articles. In later chapters, the document index will also function as a tool for adjusting for data sparseness in the element index (see Chapter 6). In the remainder of this thesis we will interchangeably use the terms "document index" and "article index," depending on which terminology better fits the context. Alongside our two text indices, we build a single index of the collection structure.

3.2.1 Indexing Structure

The structure of the collection is indexed using a relational database. To index the XML trees we use pre-order and post-order information of the nodes in the XML trees [Grust, 2002]. This indexing scheme gives us the opportunity to efficiently calculate relations between elements.

Table 3.1 describes the format of our structured index. We store the data in two database tables: one for XML documents (xmlfile) and one for XML elements (element). In the element table, the pre- and post-order provide efficient support for descendant and ancestor relation calculation. In addition, the level

```
<article>
  <sec>Animals eat food</sec>
  <sec>
    <st>Dogs</st>
    Dogs eat food
  </sec>
</article>
```

```
<article>
  <sec>
    Dogs <it>chase</it> cats
  </sec>
  <sec>Cats are animals</sec>
</article>
```

(a) Two example XML documents: doc1.xml (left) and doc2.xml (right)

elementID	fileID	tagName	preOrder	postOrder	xpos	level
1	1	article	1	4	1	1
2	1	sec	2	1	1	2
3	1	sec	3	3	2	2
4	1	st	4	2	1	3
5	2	article	1	4	1	1
6	2	sec	2	2	1	2
7	2	it	3	1	1	3
8	2	sec	4	3	2	2

(b) Structure index (element table)

term	(fileID,tf)+
animals	(1,1), (2,1)
cats	(2,2)
chase	(2,1)
dogs	(1,2), (2,1)
eat	(1,2)
food	(1,2)

(c) Document index (terms have been case-folded and stopwords removed)

term	(elementID,tf)+
animals	(1,1), (2,1), (5,1), (8,1)
cats	(5,2), (6,1), (8,1)
chase	(5,1), (6,1), (7,1)
dogs	(1,2), (3,2), (4,1), (5,1), (6,1)
eat	(1,2), (2,1), (3,1)
food	(1,2), (2,1), (3,1)

(d) Element index (terms have been case-folded and stopwords removed)

Figure 3.2: Example of a structure index (b), document index (c), and element index (d) for a collection of two documents (a).

Table 3.1: Descriptions of tables in our structural index.

(a) Table: xmlfile

Field	Type	Description
fileID	smallint(5) unsigned	Unique file identifier
name	varchar(15)	File name, relative to collection root

(b) Table: element

Field	Type	Description
elementID	int(10) unsigned	Unique element identifier
fileID	smallint(5) unsigned	Unique file identifier
tagName	varchar(15)	Tag-name of the element
preOrder	smallint(5) unsigned	Pre order of the element
postOrder	smallint(5) unsigned	Post order of the element
xpos	smallint(5) unsigned	Sibling order
level	smallint(5) unsigned	Level in the XML hierarchy

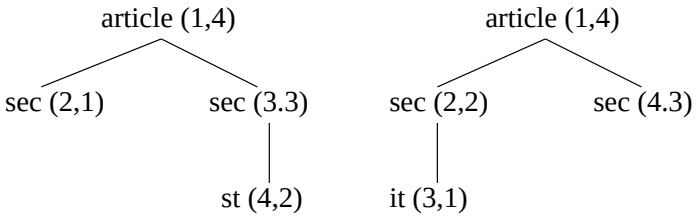


Figure 3.3: Tree representation of the two example documents from Figure 3.2 (a). Alongside each tag-name we show the (pre-order,post-order) tuple of the corresponding element.

information is useful for calculating child and parent relations. The xpos field records that the element is the xpos-th child of the type `tagName`. The xpos field is used mainly when writing out location XPaths.

We will further explain our structured index by means of an example. Figure 3.2 (b) shows the element table for two example documents shown in Figure 3.2 (a). The preOrder field of the table holds information about the order in which elements are "opened" (appearance of `<tag-name>`). The postOrder field holds information about the order in which the elements are "closed" (appearance of `</tag-name>`). As an example, the section title (`<st>`) of doc1.xml is the 4th element to be opened and the 2nd to be closed. Similarly for the italicized text (`<it>`) of doc2.xml, it is the 3rd element to opened and the 1st one to close. A tree-view of the example documents and the pre/post order information is shown in Figure 3.3.

The preOrder and postOrder plane has useful properties when calculating ancestor and descendent relationships. Figure 3.4 shows the two example docu-

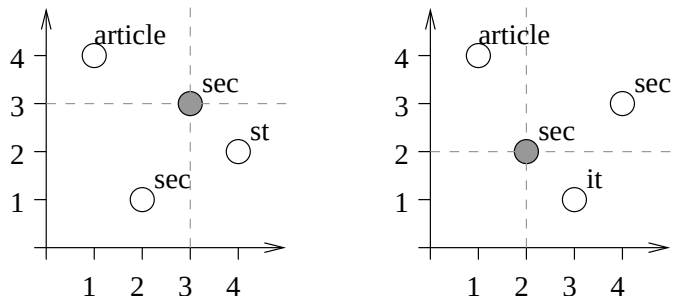


Figure 3.4: The two example XML documents from Figure 3.2 (a), when plotted in the preOrder-postOrder plane. Left: doc1.xml, with the 2nd section shaded with grey. Right: doc2.xml, with 1st section shaded with grey.

Table 3.2: Properties of the indices built using the INEX IEEE 2002 collection. *Unit* stands for the number of retrievable units. *Storage* stands for the size occupied in physical storage. *% of collection* stands for the size of the index, compared to the size of the collection. Note that we calculate collection size using the uncompressed collection, but the Lucene indices are compressed.

	Units	Storage	% of collection
Overlapping element index	6,157,742	1.5G	299%
Article index (document index)	12,107	147M	29%

ments from Figure 3.2 (a) plotted in the preOrder-postOrder plane. On the left, doc1.xml is plotted and the 2nd section is shaded with grey. We can use the plane to easily derive relationships between elements. For each element we can divide the plane into four parts by drawing horizontal and vertical lines through the point of the corresponding element. All the elements in the top-left part of the plane are ancestors, and all elements in the bottom-right part of the plane are descendants. The bottom-left part, consists of preceding elements, and the top-right part consists of following elements. In Figure 3.4 we can see that the 2nd section of doc1.xml has one ancestor (the article), one descendant (the section title) and one preceding element (the 1st section). Similarly the 1st section of doc2.xml has one ancestor (the article), one descendant (the italicized text), and one following element (the 2nd section).

3.2.2 Indexing Documents

For our document index we ignore the structure of the XML documents and index the text content of the documents. Hence, this is a standard inverted index as used for traditional document retrieval (see e.g., [Witten et al., 1999]). Figure 3.2 (c) shows the document index for our example documents. Each term is associated with a list of pairs of the form (fileID, tf) where fileID is the document identifier and the tf is the frequency of the term in the document. In the figure, all terms

have been case-folded and stopwords have been removed.

In the experiments in this thesis stopwords are removed using the English stopword list accompanying the Snowball text processing package [Snowball]. We do not apply any stemming algorithm to normalize the collection.[1] Table 3.2 (bottom) shows some statistics of our baseline document index. We see that the size of the document index is about 28% of the size of the original collection (See Table 2.1 for collection size).

3.2.3 Indexing Elements

Since individual XML elements are our units of retrieval, any XML element is a separate indexing unit. Hence the indexing unit can range from short elements such as words in italics (`<it>`) to full blown articles (`<article>`). For each element, all text nested inside it is indexed. Hence, the indexing units overlap. Text appearing in a particular nested XML element is not only indexed as part of that element, but also as part of all its ancestor elements. Figure 3.2 (d) shows the overlapping element index for our example documents in Figure 3.2 (a). As for the documents, each term is associated with a list of pairs (elementID, tf) where elementID is the element identifier and tf is the term frequency of the term in the element. For example, the term "dogs" appears twice in element 1 (doc1.xml:/article[1]), twice in element 3 (doc1.xml:/article[1]/sec[1]), once in element 4 (doc1.xml:/article[1]/sec[1]/st[1]), etc. In the figure, all terms have been case-folded and stopwords have been removed.

We use the same text processing for our element index as we used for the document index. I.e., we remove stopwords using the English stopword list that comes with Snowball, but do not apply any stemming algorithm. Table 3.2 shows some statistics of our element index. We see that the overlapping element index requires 3 times more storage than the original collection (See Table 2.1 for collection size). Although the blow-up in disk storage is a concern, we will not address it in our baseline system. We will discuss how we can reduce the size of the element indices in Chapter 5.

3.3 Retrieval Model

Our retrieval approach used in this thesis is based on the application of language models to information retrieval [Ponte and Croft, 1998, Berger and Lafferty, 1999, Miller et al., 1999, Song and Croft, 1999, Hiemstra, 2001]. We choose to apply language model for two reasons. First, language models provide an intuitive framework for combining evidence from multiple levels of document hierarchy

[1] Some preliminary experiments using the English stemmer that comes with Snowball did not give improvement in retrieval performance for the INEX collection.

(see further 6.1). Second, non-content features can be incorporated in the ranking formula in a principled manner (see further 4.2).

In the language model setting, documents are sorted by the probability $P(Q|D)$ of a query Q being randomly generated from a language model for a document D. Ponte and Croft [1998] model this probability as

$$P(Q|M_D) = \prod_{w \in Q} P(w|M_D) \cdot \prod_{w \notin Q} (1 - P(w|M_D)) \tag{3.1}$$

Hiemstra [2001], Miller et al. [1999], and Song and Croft [1999] model this probability as

$$P(Q|M_D) = \prod_{w} P(w|M_D)^{q_w} \tag{3.2}$$

where q_w is the number of occurrences of term w in query Q.

All the experiments in this thesis are performed using a multinomial language model with Jelinek-Mercer smoothing [Hiemstra, 2001]. For the implementation we use our in-house language model extension [ILPS-Lucene] of the Lucene search engine [Lucene]. We estimate a language model for each of the elements. The elements are then ranked according to their prior probability of being relevant and the likelihood of the query, given the estimated language model for the element:

$$P(e|q) \propto P(e) \cdot P(q|e). \tag{3.3}$$

We assume query terms to be independent, and rank elements according to

$$P(e|q) \propto P(e) \cdot \prod_{i=1}^{k} P(t_i|e), \tag{3.4}$$

where q is a query made out of the terms $t_1, \ldots, t_k$. To account for data sparseness we estimate the element language model by taking a linear interpolation of two models one for the element itself, and one for the collection. That is, $P(t_i|e)$ is calculated as

$$\lambda \cdot P_{mle}(t_i|e) + (1 - \lambda) \cdot P_{mle}(t_i), \tag{3.5}$$

where $P_{mle}(\cdot|e)$ is a language model for element e; and $P_{mle}(\cdot)$ is a language model of the collection. The parameter λ is called an *interpolation factor* (*smoothing parameter*). We estimate the language models $P_{mle}(\cdot|e)$ and $P_{mle}(\cdot)$ using maximum likelihood estimation. For the element model we use statistics from the element index. For the collection model there are several options we can use. The straightforward one would be to use statistics from the element index, either collection frequencies or element frequencies. However, one can argue that the overlapping element index as a whole does not represent a natural language—i.e., the terms are indexed in an overlapping manner and their frequency in the index does thus not reflect their frequency in the original text (written in natural language). To account for this one can use a document index to estimate the

collection model. The term frequencies in the article index do reflect the term frequencies in the original collection. As for the element index, we can either choose collection frequencies or document frequencies from the document index. Despite the un-naturalness of the language, in our baseline experiments we will use—as our collection model—the overlapping element index and element frequencies.

For ease of implementation purposes, the probability measure (equation 3.5) is rewritten into a different format [Hiemstra, 2001, pages 75–76]. I.e., we use a presence weighting scheme [Robertson and Spark Jones, 1976]; we divide the formula with the collection model; and use a sum of logarithm weights instead of a product of weights. The rewriting results in a scoring function $s(q, e)$, for an element e and a query q made of the terms $t_1, \ldots, t_k$,

$$s(e,q) = \sum_{i=1}^{k} \log \left(1 + \frac{\lambda \cdot \mathrm{tf}(t_i, e) \cdot (\sum_t \mathrm{df}(t))}{(1-\lambda) \cdot \mathrm{df}(t_i) \cdot (\sum_t \mathrm{tf}(t, e))} \right), \tag{3.6}$$

where $\mathrm{tf}(t, e)$ is the frequency of term t in element e, $\mathrm{df}(t)$ is the element frequency of term t, and λ is the smoothing parameter.

The language modeling framework allows us to easily model non-content features—using the prior probability $P(e)$. Prior probabilities have proven to be useful for various retrieval tasks [Kraaij et al., 2002]. In our baseline system, however, we will use a uniform prior, i.e., each element has the same prior probability. Later—in Section 4.2—we will look at how we can use the prior to enhance our retrieval.

3.4 The Effect of Smoothing

Most retrieval models have a set of parameters that need to be tuned to get optimal performance out of the retrieval system. The tuning is important to adjust the system to factors that are outside the model itself, such as the collection, the retrieval task, etc. [Greiff and Morgan, 2003]. In the language model framework it has been shown that the retrieval performance is generally sensitive to the value given to smoothing parameters [Zhai and Lafferty, 2004]. Smoothing is applied to account for data-sparseness and is therefore considered more useful for short text units than longer ones. The data-sparseness problem is particularly evident in a collection of very short texts, such as our collection of XML elements. In this section we will look at tuning the smoothing parameter of the language model. The smoothing parameter determines the ratio between the emphasis put on the model of the retrieved element and the model of the whole collection. We will look at how the tuning is affected by different query formats.

Query formats We tune the parameters for the three query formats introduced in Section 2.3.3. First, we look at the *title* field; then, we look at the *description* field; and finally, we look at the combination of the *title* and *description* fields.

Table 3.3: Element retrieval: Optimal value for the smoothing parameter λ. We report three quantizations: strict, generalized (gen), and specificity-oriented generalized (sog2). Improvement is calculated relative to our baseline "vanilla" system where $\lambda = 0.15$.

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
Title	.95	.0530	39%***	.95	.0707	60%***	.95	.0536	50%***
Desc.	.95	.0434	15%	.80	.0593	41%***	.75	.0457	32%*
T+D	.90	.0609	49%***	.90	.0801	61%***	.85	.0600	49%***

Quantizations The tuning of parameters requires that we have a known goal function. As mentioned in Section 2.3.5, the appropriate goal function in XML element retrieval is still an open research area. We will tune our system using the three different quantizations discussed in Section 2.3.5. For each quantization we will use an appropriate metric (see Section 2.5 for more information on the experimental setup). Our aim is not to evaluate the appropriateness of one quantization over another. Rather, we will look at the three quantizations independently and look at the effect of assuming that a particular quantization is the appropriate one.

Vintage As we pointed out in Section 2.3, the INEX evaluation framework has changed from one year to another. For instance, the definition of the relevance dimensions has changed slightly over the years. More importantly, the assessment tool has changed substantially from one year to another. These changes may have an impact on the effectiveness of our system. We will thus report separately results for each vintage of the INEX collection.

3.4.1 Evaluation

We evaluate the effect of the smoothing parameter by exploring values between 0.05 and 0.95—using increments of 0.05. We report the optimal values for different quantization methods, different query formats, and different vintages of the INEX collection. We compare the performance of the optimal settings to a "vanilla" baseline run, where λ is given the value 0.15. The value for the baseline is chosen based on previous experiments which suggest that a low value should be used for the smoothing parameter when we do ad-hoc retrieval using verbose queries [Hiemstra, 2001, Zhai and Lafferty, 2004]. In the experiments by Zhai and Lafferty [2004], the optimal value of the smoothing parameter was, however, higher when short query formats were used. In advance, we would thus expect our baseline to be close to the optimal settings for the verbose query formats, but not for the title-only query format.

Table 3.4: Element retrieval: Optimal value for the smoothing parameter λ, for each vintage of the INEX collection. We report three quantizations: strict, generalized (gen), and specificity-oriented generalized (sog2). Improvement is calculated relative to our baseline "vanilla" system where $\lambda = 0.15$.

(a) Using the *title* field

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
2002	.95	.0385	133%*	.95	.0591	108%***	.95	.0357	96%**
2003	.95	.0571	61%***	.95	.0595	72%***	.95	.0481	64%***
2004	.95	.0662	46%***	.95	.1068	57%***	.90	.0628	47%***
2005	.20	.0535	0.2%	.90	.0643	32%	.90	.0643	32%

(b) Using the *description* field

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
2002	.95	.0410	99%*	.95	.0655	76%***	.95	.0393	55%
2003	.80	.0567	36%**	.95	.0592	61%***	.90	.0457	49%*
2004	.95	.0739	90%*	.75	.0716	44%***	.60	.0513	34%**
2005	.20	.0473	0.4%	.50	.0490	15%	.50	.0490	15%

(c) Using both the *title* and *description* fields

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
2002	.95	.0528	108%***	.95	.0907	103%***	.95	.0545	83%***
2003	.90	.0775	78%***	.95	.0768	90%***	.95	.0607	79%***
2004	.95	.0749	72%	.80	.0983	56%***	.75	.0664	44%***
2005	.25	.0497	0.8%	.70	.0626	27%	.70	.0626	27%

Query formats Let us first look at the optimal smoothing settings for different query formats, calculated over all available topics. The optimal values for the smoothing parameter are shown in Table 3.3.[2] The results are somewhat unexpected. The optimal value for the smoothing parameter is high for all query formats and all quantization methods. I.e., little smoothing is required and thus a high weight is given to the element model. These results are not in-line with the results of Zhai and Lafferty [2004], which showed that verbose queries needed very aggressive smoothing settings. Figure 3.5 shows in more detail the effect of changing the smoothing parameter. For all query formats the mean average

[2]Note that the MAP and MAep scores in Table 3.3 are considerably lower than what people might be used to from document retrieval. However, evaluation scores are generally much lower for XML element retrieval than document retrieval, partly, due to the large recall-base—the large number of relevant elements (see Table 2.4).

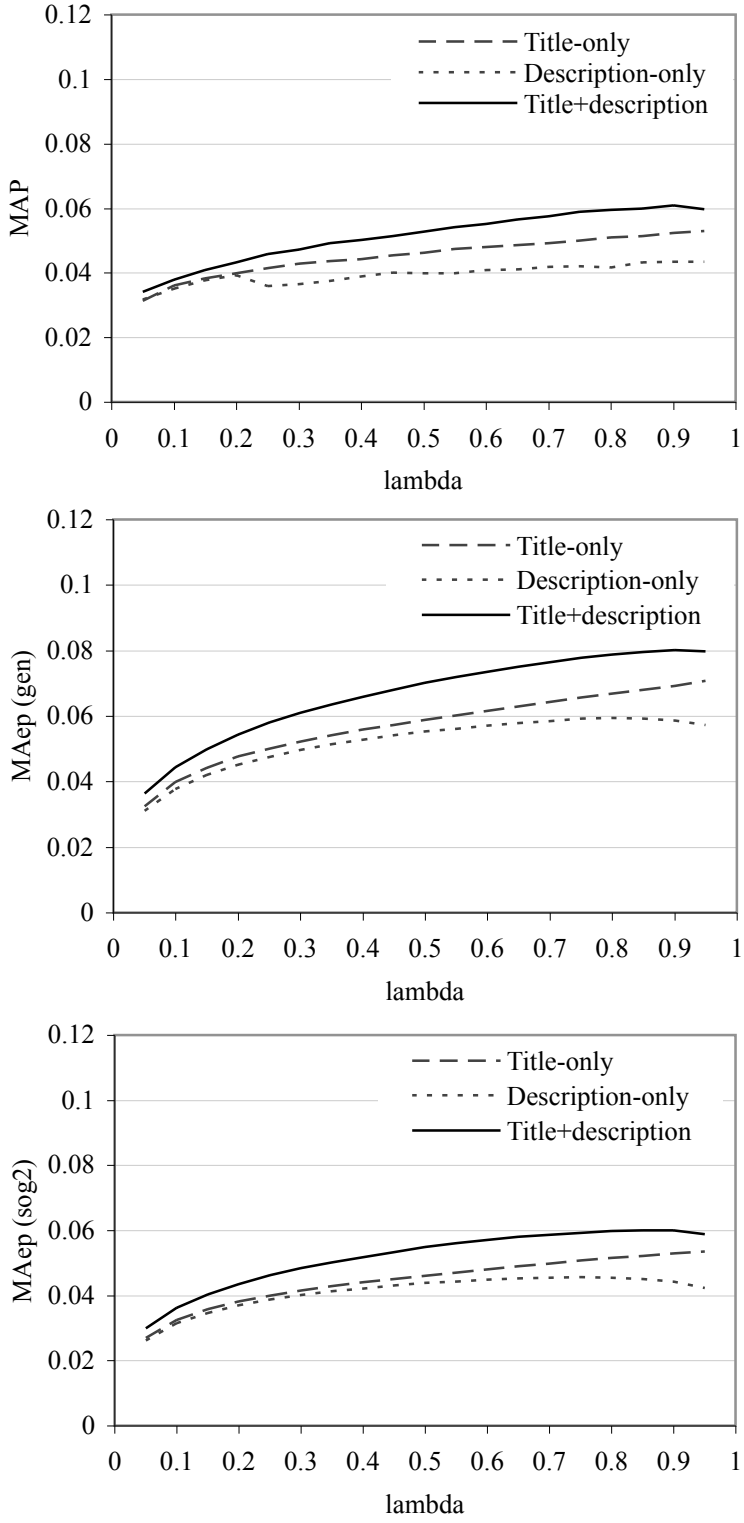


Figure 3.5: Element retrieval: Mean average (effort) precision of retrieval runs using different query format and different values of the smoothing parameter. Top: strict quantization. Middle: generalized quantization (gen). Bottom: specificity-oriented generalized quantization (sog2).

precision is low when a low value is used for λ. As λ increases, the mean average precision increases. For all query formats and all quantizations the optimal smoothing improves over our "vanilla" baseline. The improvement is statistically significant in all cases except one. This is indeed surprising since we expected our vanilla baseline to be close to optimal settings for the two verbose topic formats.

If we compare the performance of different topic formats, we see that the title-only format performs consistently better than the description-only format for all quantizations. Combined, the two topic fields give better performance than the title-only format for all quantizations.

Vintage Let us now look at the difference in performance between different vintages of the INEX test collection. Table 3.4 shows the optimal value for the smoothing parameter for the different vintages.

In terms of the strict quantization and mean average precision, the optimal value of the smoothing parameter λ is high for all query formats in the 2002 to 2004 collection. In the 2005 collection, however, the optimal value for the smoothing parameter λ is in the lower range for all the topic formats.

Let us now look at the retrieval performance in terms of generalized quantization and mean average effort precision. Table 3.4 shows that the optimal value for the smoothing parameter is high for all years. I.e., the optimal value for the 2005 topics is now in the high end, and hence in sync with the previous years.

For the 2002–2004 collections, the optimal smoothing parameter settings give significant improvements over the "vanilla" baseline settings. This holds generally for all query formats and all quantizations. For the 2005 collection, the optimal smoothing parameter settings do not give a significant improvement over the baseline settings.

Overfitting When parameters of statistical models are evaluated it is common practice to divide the evaluation collection into two parts: a training set and a testing set. The training set is used to tune the system parameters and the testing set is used to validate that the parameter settings were not "overfitted" to the training data [Kearns et al., 1997]. Overfitting refers to the case where a model performs well on the training data, but does not generalize to the process which generates the data and thus does not perform as well on the testing data.

In our evaluation so-far we have not addressed the issue of overfitting. We have found the optimal setting for the overall collection and for individual vintages of the collection. From Table 3.4 we see that although the optimal values for different vintages tend to be similar, they are not the same. We are thus running the risk of overfitting the smoothing parameter to each vintage. We address the overfitting issue by looking at the retrieval performance for each vintage using the overall optimal parameter settings. Hence, we test whether our overall parameter settings carry over to individual vintages.

Table 3.5: Element retrieval: Performance for each vintage of the INEX collection, using the overall optimal parameter settings. Improvement is calculated relative to our baseline "vanilla" system where $\lambda = 0.15$.

(a) Using the *title* field

	strict ($\lambda = 0.95$)		gen ($\lambda = 0.95$)		sog2 ($\lambda = 0.95$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0385	133%*	.0591	108%***	.0357	96%**
2003	.0571	61%***	.0595	72%***	.0481	64%***
2004	.0662	46%***	.1068	57%***	.0624	46%***
2005	.0491	-8.1%	.0640	31%	.0640	31%

(b) Using the *description* field

	strict ($\lambda = 0.95$)		gen ($\lambda = 0.80$)		sog2 ($\lambda = 0.75$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0410	99%*	.0605	62%***	.0368	45%*
2003	.0529	27%	.0574	56%***	.0448	46%**
2004	.0739	90%*	.0708	43%***	.0510	34%*
2005	.0062	-87%	.0469	10%	.0479	12%

(c) Using both the *title* and *description* fields

	strict ($\lambda = 0.90$)		gen ($\lambda = 0.90$)		sog2 ($\lambda = 0.85$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0511	101%**	.0871	95%***	.0516	73%***
2003	.0775	78%***	.0751	86%***	.0585	73%***
2004	.0649	55%	.0963	53%***	.0656	42%**
2005	.0460	-6.7%	.0610	24%	.0622	26%

Table 3.5 shows the results of using the overall optimal parameter settings. We see that using the overall optimal settings instead of per-vintage optimal settings we generally get a significant improvement over our "vanilla" baseline. This means that our overall optimal parameter settings are not particularly fitted to a single vintage, but give reasonable results for all vintages. The notable exception is the 2005 collection using the strict quantization. There is a decrease in performance, but not significant. The -87% decrease for the 2005 vintage using the description-only field can be to a large extent be explained by the performance of a single topic—topic 230—which has only 2 elements that are assessed highly exhaustive and highly specific (a single list-item and its enclosed paragraph). The MAP of the "vanilla" baseline run for that topic is 1.0, but the MAP of the optimized run is 0.0091 for the topic.

Table 3.6: Document retrieval: Optimal value for the smoothing parameter λ when evaluated in terms of MAP. Improvement is calculated relative to our baseline "vanilla" system where $\lambda = 0.15$.

	λ	MAP	Impr.
Title-only	.55	.3617	3.6%
Description-only	.10	.3028	0.6%
Title+description	.15	.3556	–

Summary Let us now summarize our observations on the effect of the smoothing parameter:

High λ: The optimal value for the smoothing parameter is very high, about 0.95. This holds generally for all query formats and quantizations.

2005: The 2005 topics are a striking exception. The optimal value for the λ is low when measured in terms of mean average precision using strict quantization.

We will analyze these results in more detail in Section 3.5. First, we compare our element retrieval settings to the optimal settings for document retrieval.

3.4.2 Document Retrieval

One of the aims of this chapter was to contrast two tasks: element retrieval and document retrieval. In this sub-section we will look at the document retrieval task (see Section 2.5 for the details on the setup of the document retrieval task). Table 3.6 shows the optimal value for the smoothing parameter for the different topic formats; and Figure 3.6 shows the mean average precision for the document retrieval task, over different values of the smoothing parameter λ. The optimal values for the smoothing parameter λ are in-line with the results of Zhai and Lafferty [2004]. The verbose topics need aggressive smoothing (low λ), but the title-only topics need less smoothing (a higher λ). Note that we are using the same topics as we did for the element retrieval experiments. These results do thus indicate that the unexpected values we saw for the element retrieval task are the result of the task rather than the topics.

Vintage Table 3.6 shows the document retrieval evaluation for individual vintages of the INEX collection. The results for the individual vintages are similar to the overall results, with a few notable exceptions. The optimal values for the 2004 topic set are lower than for the remaining topic sets. This is, in particular, visible for the query formats using the title field. This may be caused by the fact that the 2004 queries were relatively long and may thus require more smoothing (see Table 2.2 on page 25). The 2005 query formats which include the description field require less smoothing that their counterparts in the previous years.

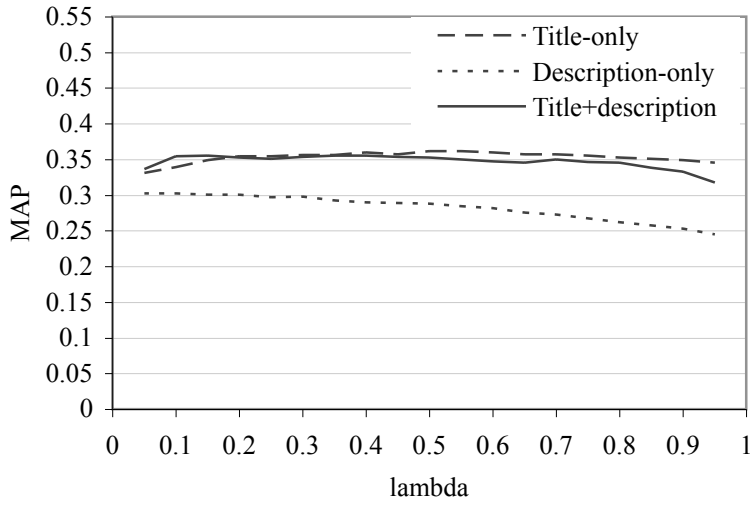


Figure 3.6: Document retrieval: Mean average precision for different values of the smoothing parameter λ.

Table 3.7: Document retrieval: Optimal parameter settings of each "vintage" of the INEX collection. Improvement is calculated relative to our baseline "vanilla" system where $\lambda = 0.15$.

(a) Title-only

	λ	MAP	Impr.
2002	.45	.2920	3.1%
2003	.55	.3380	5.8%
2004	.15	.4132	–
2005	.55	.4269	17%*

(b) Desc.-only

	λ	MAP	Impr.
2002	.10	.2437	0.4%
2003	.15	.3060	0.0%
2004	.05	.3846	4.5%
2005	.40	.2888	5.1%

(c) Title+desc.

	λ	MAP	Impr.
2002	.35	.3104	1.5%
2003	.45	.3516	1.9%
2004	.10	.4174	1.9%
2005	.70	.3791	8.0%

The reason for this is not clear, as many factors may play a role in causing this difference. We conjecture that the difference may be explained by a change in the role of the description field in 2005. In that year a structured version of the query was added to many of the topics and the text in the description field was changed—from being a more verbose description of the content of the desired results—to being a description of the syntactic form of the structured query.

3.5 Discussion

We have found that the optimal smoothing parameter settings differ considerably between element retrieval and document retrieval. While aggressive smoothing is useful for document retrieval, smoothing did not prove useful for element retrieval. In order to try to explain this difference we look at the fundamental difference between element retrieval and document retrieval—namely, the unit of retrieval.

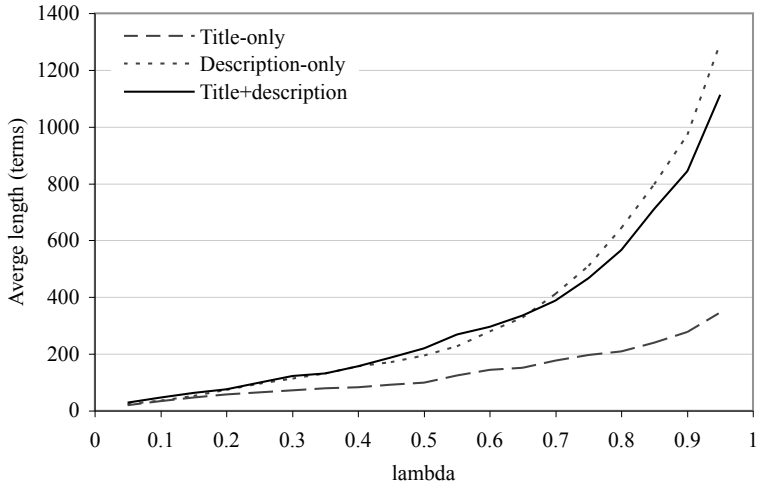


Figure 3.7: Element retrieval: Average length of the top-10 results of runs using different values for the smoothing parameter λ.

3.5.1 Length

Let us first look at the length of retrieved units. Figure 3.7 shows the average length of the top-10 elements retrieved using different values of the smoothing parameter λ. The figure shows results for the three query formats. From the figure we can see a clear length-bias effect. As we apply less smoothing—assign higher value to λ—the average length of the results increases. The effect is greater for the verbose query formats than for the title-only queries. This length effect can be explained by the connection between the smoothing parameter and coordination level ranking [Hiemstra, 2001, Appendix B]. Coordination level ranking is a partial ranking of results such that all results containing k query terms are ranked above results containing $k-1$ query terms. As the smoothing parameter approaches 1, the ranking approaches coordination level ranking. I.e., as we increase λ we put more emphasis on results containing all query terms. Obviously, short elements are less likely to contain all query terms than longer ones. Hence, the average length of elements increases when we increase λ. This effect is greater for verbose queries, simply because they contain more terms and hence the presence of more terms is required in the result elements.

Let us now do the same analysis for the document retrieval task. Figure 3.8 shows the average length of the top-10 retrieved documents for different values of the smoothing parameter. We see the same length-bias effect for documents as we saw for the elements. The effect is, however, not as dramatic as in the element retrieval case. In sum, decreasing the amount of smoothing—i.e., increasing the value for the smoothing parameter—has a length-bias effect for both the element retrieval and document retrieval tasks. The effect is, however, more dramatic for the element retrieval task.

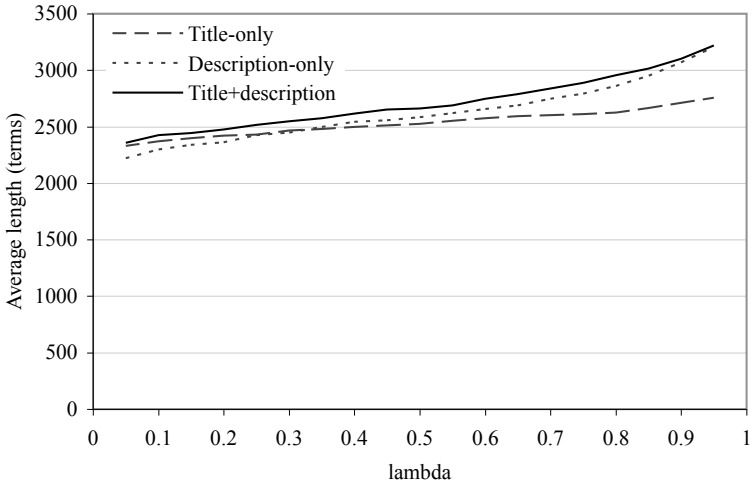


Figure 3.8: Document retrieval: Average length of the top-10 results of runs using different values for the smoothing parameter λ.

3.5.2 Unit of Retrieval

Let us now take a closer look at the element retrieval task and analyze the tag-names of the retrieved elements. Table 3.8 shows the frequency of different tag-names in the top-10 results for the INEX 2002–2005 topics. The table shows the average number or elements appearing in the top-10 results, averaged over all topics. The tables on the left hand side show the results of using "normal" smoothing settings ($\lambda = 0.15$); and for contrast the tables on the right hand side show the results of using the value 0.95 for the smoothing parameter. For the title-only query format, *paragraphs* (`<p>`) are the most frequent tag-name in the list of top-10 results for both runs. For the baseline run short element types such as *titles* (`<atl>`, `<ti>`, `<st>`), *emphasized text* (`<it>`, `<b>`), and headings (`<h>`) are frequent in the top-10. The short elements are less frequent in the run using $\lambda = 0.95$, but the frequency of *sections* (`<sec>`, `<ss1>`) has increased. We also see that the two types of very long elements—articles (`<article>`) and bodies (`<bdy>`)—appear in the list for the high λ, but are absent in the list for our baseline. For the verbose query formats, the baseline run is similar to that for the title-only query. If we use a high value for λ we see an increase in frequency of longer element types.

3.5.3 Why is the 2005 Vintage Different?

In this chapter we have seen that the 2005 vintage of the INEX test collection is considerably different from the 2002–2004 vintages. There are several factors that can play a role in the difference. First, characteristics of the topics can be different from one year to another—it is well-known that the topics can vary considerably between yearly cycles of evaluation initiatives [Harman, 2005]. Second,

Table 3.8: Most frequent tag-names appearing in the top-10 results. The Avg.freq. is the number of results in the top-10 that have the corresponding tag, averaged over all topics. Standard deviation is shown in brackets.

(a) Title-only queries

$\lambda = 0.15$			$\lambda = 0.95$		
Tag	Avg.	freq.	Tag	Avg.	freq.
p	2.39	(2.07)	p	2.27	(1.88)
atl	1.65	(2.51)	sec	1.03	(1.21)
it	1.19	(1.85)	atl	0.94	(2.04)
h	0.51	(1.22)	ip1	0.60	(0.77)
st	0.45	(0.88)	it	0.55	(1.42)
ip1	0.42	(0.77)	ss1	0.53	(0.80)
ti	0.41	(1.22)	article	0.50	(1.03)
b	0.38	(0.90)	bdy	0.44	(0.87)
sec	0.26	(0.62)	bb	0.27	(0.99)
bb	0.23	(0.70)	ti	0.26	(0.99)

(b) Description-only queries

$\lambda = 0.15$			$\lambda = 0.95$		
Tag	Avg.	freq.	Tag	Avg.	freq.
p	3.23	(2.07)	article	1.93	(1.68)
atl	1.24	(2.06)	sec	1.68	(1.36)
it	0.71	(1.39)	bdy	1.63	(1.54)
ip1	0.66	(0.83)	p	1.62	(1.88)
sec	0.44	(0.74)	ss1	0.49	(0.82)
ti	0.40	(1.10)	ip1	0.41	(0.75)
st	0.40	(0.81)	bm	0.32	(0.72)
b	0.30	(0.90)	atl	0.30	(1.28)
ss1	0.29	(0.60)	app	0.21	(0.48)
bb	0.26	(0.98)	list	0.13	(0.38)

(c) Title+description queries

$\lambda = 0.15$			$\lambda = 0.95$		
Tag	Avg.	freq.	Tag	Avg.	freq.
p	2.55	(2.04)	article	1.68	(1.59)
atl	1.55	(2.34)	p	1.55	(1.65)
it	0.94	(1.67)	sec	1.53	(1.25)
ip1	0.59	(0.83)	bdy	1.36	(1.39)
st	0.40	(0.83)	ss1	0.52	(0.72)
ti	0.40	(1.21)	atl	0.48	(1.51)
sec	0.38	(0.70)	ip1	0.42	(0.73)
h	0.34	(0.94)	bm	0.41	(0.81)
b	0.33	(0.93)	bb	0.21	(0.83)
bb	0.29	(0.96)	it	0.21	(0.77)

the assessors' understanding of the notions of relevance can change between years. This effect may be introduced by a change in the assessment guidelines, the increased experience of veteran assessors, and the ratio between veteran and novice assessors. Third, the changes in the assessment interface can affect the relevance assessments. In 2005, the interface for assessing specificity changed fundamentally. Instead of assigning an explicit specificity value for each element, specificity values were derived from "yellow-marking" of relevant text (see Section 2.3.4 for more details on the change in assessment interface).

Without further analysis we cannot explain the reason for the different behavior of the 2005 vintage. We will return to this issue when we analyze the element length of the INEX assessments in Chapter 4; and when we analyze the concept of *unit of retrieval* and the INEX assessments in the Chapter 5. The different behavior of the 2005 vintage will also feature in our evaluation of retrieval effectiveness in Chapters 4, 5, 6, and 7.

3.6 Conclusions

In this chapter we have applied our "off the shelf" document retrieval system to both the XML element retrieval task and XML document retrieval task. The research question we wanted to address in this chapter was:

> How is element retrieval different from document retrieval?

First, let us compare element and document retrieval in terms of the optimal value for the smoothing parameter.

- Element retrieval performance is more sensitive to the value of the smoothing parameter.
- For the element retrieval task, applying little smoothing gave the best retrieval performance.
- For the document retrieval task, aggressive smoothing gave the best retrieval performance.

The values for the document retrieval case are in line with results in the literature [Zhai and Lafferty, 2004, Hiemstra, 2001].

We analyzed the average length of retrieved elements and retrieved documents and made the following observation:

- For both tasks, decreased smoothing leads to the retrieval of longer retrieval units.
- The length-bias effect is, however, more dramatic for the element retrieval task.

Combining the observations for the optimal smoothing parameter values and the length-bias effect leads us to conjecture that an important difference between the element retrieval task and the document retrieval task lies in the effect of length normalization. In the next chapter (Chapter 4) we will look at length normalization in more detail.

We analyzed the tag-names of elements returned by the optimal smoothing parameter settings and compared it to the tag-names of elements returned by our "vanilla" baseline. There is a relation between the smoothing parameter settings and the units of retrieval. I.e., changing the smoothing parameter results in a difference in the type of elements retrieved. In Chapter 5 we take a closer look at the relation between retrieval performance and unit of retrieval.

Chapter 4

Length Normalization

In the previous chapter we studied a baseline system for XML element retrieval. Based on our study of the effects of the language model smoothing parameter we made two main observations:

- The length of retrieved elements is sensitive to the parameter settings of the retrieval model.
- Retrieval performance is also sensitive to the parameter settings.

In this chapter we look closely at the relation between length and retrieval performance. Our main question is:

> What is the impact of length normalization for XML retrieval?

We take a look at *length* of relevant elements, and we extend our baseline retrieval formula with so-called *length-priors*. Before going into details, we provide a high level overview of these three main themes of the chapter.

In Section 4.1 we look at the length of the INEX assessments. We have two main research questions:

- What are the characteristics of element assessments in terms of length?
- How is the length distribution of relevant elements different from the distribution of relevant documents?

Our main finding is that there is a clear length bias in the INEX element assessments. We see that this bias is much grater for element retrieval than for document retrieval.

Prior probabilities have proven useful technique for other retrieval tasks [Kraaij et al., 2002]. In Section 4.2 we investigate the following research question:

- Can we improve the effectiveness of element retrieval by using length priors?

We find that length priors give significant improvements in retrieval performance. We also show that their effect depends on the query format; they are more effective for verbose queries than for short queries.

After the two sections outlined above, we provide a brief conclusion.

The work in this chapter is a considerably extended version of previously published work [Kamps et al., 2003b, 2004a, 2005a].

4.1 Length in the Assessments

A major challenge of the XML element retrieval task is to identify the appropriate unit of retrieval. In principle, any element can be retrieved. However, it is not likely that all elements are equally suited as a retrieval unit. To better understand the element retrieval task we need to get some idea about the kind of elements we are dealing with. Thus, we look at the characteristics of the elements in the INEX document collection. In particular we investigate whether the characteristics of relevant elements are different from the overall element characteristics. More precisely, we compare two sets of elements. On the one hand, the set of all elements in the collection. On the other hand, the set of elements assessed strictly relevant in the INEX relevance assessment process—i.e., elements assessed as both highly exhaustive and highly specific. We compare these two sets in terms of two types of characteristics. First, in Section 4.1.2 we look at the distribution of element *lengths* in the two sets. Later, in Section 5.1 in the next chapter we look at the frequency of certain tag-names in the two sets.

4.1.1 Experimental Setup

Our analysis of element characteristics is based on our overlapping element index (see Section 3.2). We look at the characteristics of elements that are retrievable by our system. Hence, we only count elements that contain at least one term in our element index. Out of the 8,222,075 elements in the INEX collection 6,157,724 elements are in our element index. The remaining 2,064,351 elements do not contain any terms and are thus not indexed. There are two distinct types of "empty elements" that do not enter our element indices.

- Elements that are empty in the collection. There are 269,239 elements of this type. The most frequent tag-names of empty elements are pointers to external images (`<art>` 81,544), table entries (`<entry>` 61,580), table column specifications (`<colspec>` 32,527), and table span specifications (`<spanspec>` 29,402).

- Elements that contain only stopwords and are thus empty after the text has gone through a pre-processing stage. There are 1,795,112 elements of this

Table 4.1: Exponential-sized bins.

Bin	1	2	3	4	5	6	7	8	9	10
Log max length	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50
Max length	1	3	5	10	17	31	56	100	177	316
Bin	11	12	13	14	15	16	17	18	19	20
Log max length	2.75	3.00	3.25	3.50	3.75	4.00	4.25	4.50	4.75	5.00
Max length	562	1000	1778	3162	5623	10000	17782	31622	56234	100000

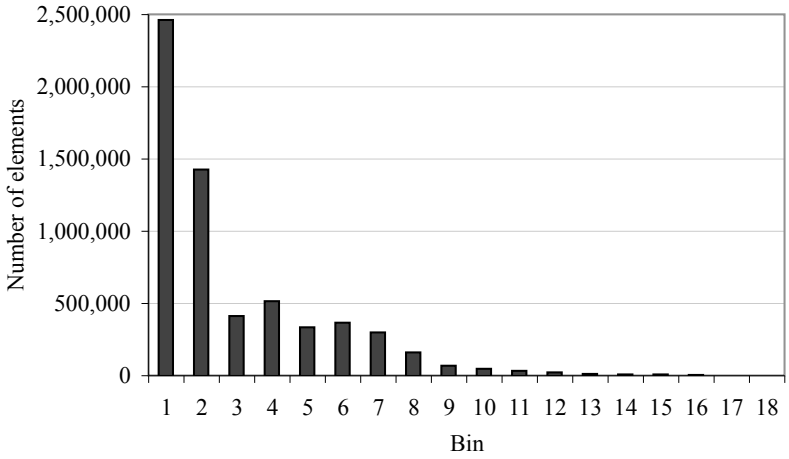
type. The most frequent tag-names of elements containing only stopwords are italicized text (`<it>` 904,352), first names (`<fnm>` 190,460), mathematics in text (`<tmath>` 170,549) and subscripts (`<sub>` 154,477).

The number of elements indexed is rather sensitive to the type of text pre-processing that is being used. In a previous publication [Kamps et al., 2005a] we report statistics using a different pre-processor than is used in this thesis. Our previous approach resulted in 6,779,686 elements being indexed. There are two main differences between the previous and current approach. First, math elements were not indexed in the previous approach. Second, single character words are considered as stopwords in our current approach, but were included in our index in the previous analysis. The large number of `<it>` elements that contain only stopwords is mainly due to the use of the `<it>` tag to present single character variable-names in mathematical texts, e.g., "`... let <it>N</it> denote a natural number ...`". The differences do result in noticeable difference in index statistics. However, the differences do not cause significant differences in retrieval performance, since the differences do not affect terms that are frequently used in queries. This can be verified empirically by comparing the results presented in this thesis and the results presented in the previous publication.

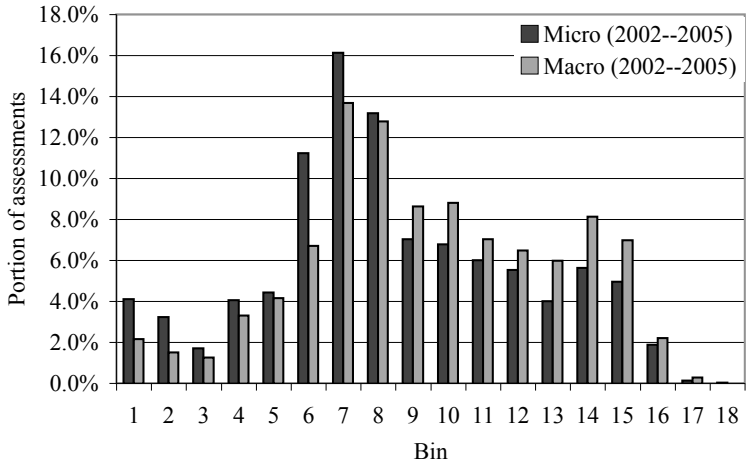
4.1.2 Length of Relevant Elements

To better understand the importance of element length for XML retrieval, we analyze the length of XML elements versus the length of relevant XML elements. We do this by ordering the elements in the INEX collection by length, and grouping them into several "bins" [Singhal et al., 1996b]. As before, we calculate length as the number of term occurrences in an element. Following Kraaij et al. [2002], we use exponential-sized bins. Specifically, we use 20 bins on an exponential scale ranging from 10^0 (=1) to 10^5 (=100,000). Table 4.1 gives the length of the longest element for each of the bins.

Figure 4.1 (a) shows the number of XML elements for each of the bins. The distribution of elements is heavily skewed toward short elements. The average XML element is short—with a length of 28 terms—while the median length of elements is only 2 terms.



(a) Length distribution of the INEX collection



(b) Length distribution of the INEX 2002-2005 strict assessments

Figure 4.1: Top: Length distribution of XML elements in the INEX IEEE Computer Society Collection. Bottom: Length distribution of the strict assessments in INEX 2002–2005. Micro: the fraction of elements in each bin. Macro: the fraction of elements in each bin, averaged over topics.

We also investigate the length of *relevant* XML elements, by using the strict assessments of INEX 2002–2005 CO topics. Figure 4.1 (b) shows the distribution of relevant XML elements over the 20 bins described above. We show both the absolute fraction of elements in each bin (micro), and the fraction normalized by the total number of strict relevant assessments for each topic (macro). In the micro distribution frequencies are not normalized for individual topics. This means that a topic with many relevant elements will have a greater impact on the distribution than a topic with few relevant elements. In the macro distribution the normalization means that all topics have the same weight in the overall

Table 4.2: Average and median length of elements in each assessment set and in the collection as whole. For the assessments we calculate mean average length (macro average) and mean median length (macro median) over different topics.

	Average		Median	
	micro	macro	micro	macro
Collection	28		2	
All assessments	730	921	79	661
2002 assessments	1294	1515	189	1119
2003 assessments	876	1101	190	830
2004 assessments	333	782	43	472
2005 assessments	110	220	35	181

distribution. This means that a topic with few relevant elements has the same impact as a topic with many relevant elements.

From Figure 4.1 we see that there is a radical difference between the length distributions of relevant XML elements and the distribution of all XML elements in the collection. Most of the elements in the collection are very short, having three or fewer terms (bins 1–2). The relevant elements do have a more even distribution, with a peak around elements having 32–100 terms (bins 7–8). There is another small peak around elements containing 1,778–5,623 terms (bins 14–15). Table 4.2 shows the average and median length of the assessments, compared to the collection. We see that the length of the "average relevant element" is considerably greater than the length of the "average element." I.e., the average length of the elements in the collection is 28 terms but the average length of relevant elements is 730 terms (921 terms if we use macro average).

Figure 4.2 shows the length distribution of strict assessments, separately for each year of the INEX test collection. If we compare the assessments from one year to another we see that there is a clear trend toward shorter elements each year. Table 4.2 reveals that the average length of relevant elements has changed from 1,294 terms in 2002 (1,515 terms if we use macro average) to 110 terms in 2005 (220 terms if we use macro average). However, even if the length of the "average relevant element" is getting smaller, it is still considerably longer than the length of the "average element" in the collection—which is only 28 terms.

The differences we see in the length of assessments from one year to another may be influenced by a number of factors.

- First, the topics are different from one year to another. The topic authors (and INEX organizers responsible for selecting topics) may have gone for more specific information needs—as opposed to general information needs—as years went by.

- Second, the assessors are not the same from one year to another. The

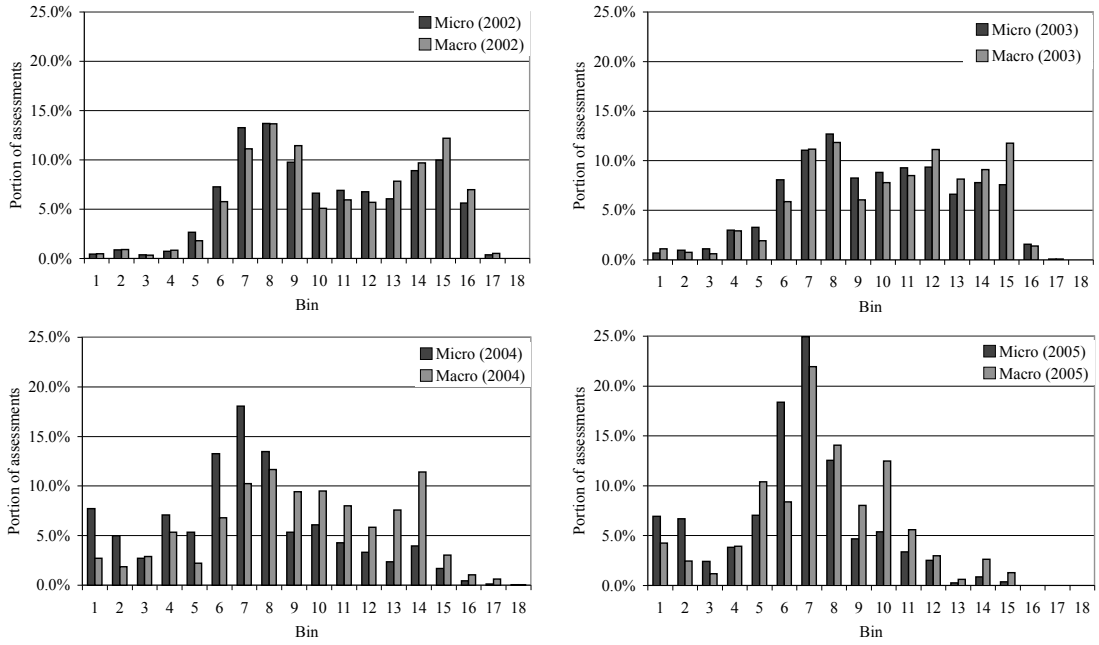


Figure 4.2: Length distribution of elements assessed strictly relevant in the INEX 2002–2005 test collections. Micro: the fraction of elements in each bin. Macro: the fraction of elements in each bin, averaged over topics.

understanding of the assessment process may differ between veteran and novice assessors.

- Third, the assessment interface has changed from one year to another.

From Figure 4.2 we see that the 2005 assessments are strikingly different from the remaining years in the sense that the longest elements have disappeared from the assessments. The bins 13 and above are populated in the 2002–2004 assessments, but hardly in the 2005 assessments. I.e., the longest elements in the collection are no longer considered as being both highly exhaustive and highly specific. The strikingly different assessments of 2005 coincide with a fundamental change in the assessment interface. The introduction of the "yellow marker" in 2005 is a clear example of how changes in the assessment interface may influence the assessments. Users seem not to be likely to highlight a long piece of text, whereas before they had no problem with assessing a long piece of text as highly specific.

The comparison of length distributions that we present here is intended to give insight into general trends in assessment characteristics between years. A more principled statistical analysis of the distributions is left as future work.

Let us now summarize our observations of our length analysis:

- The length distribution of the collection is very different from the length distribution of relevant elements.

Table 4.3: Average and median length of documents in each assessment set and in the collection as a whole. We calculate mean average length (macro average) and mean median length (macro median) over different topics.

	Average		Median	
	micro	macro	micro	macro
Collection	2,839		2,378	
All assessments	3,372	3,602	2,850	3,383
2002 assessments	3,503	3,744	2,875	3,550
2003 assessments	3,403	3,500	2,984	3,327
2004 assessments	3,413	3,503	2,852	3,266
2005 assessments	3,081	3,687	2,637	3,423

- The average element is very short.
- The average relevant element is substantially longer.
- The "very long" elements (in bins 13–18) that appeared in the 2002–2004 assessments are not present in the 2005 assessments.

4.1.3 Length of Relevant Documents

For comparison, we repeat the analysis that we have just conducted, but now with documents instead of elements. I.e., we look at the length distribution of the documents in the INEX IEEE document collection. The aim of this analysis is to stress further the fundamental difference between element retrieval and document retrieval.

Figure 4.3 (top) shows the number of documents for each of the bins. Compared to the element length distribution, the document length distribution is more even. There is no single bin that is completely dominating and there is less difference between the mean and the median. For the documents, the mean length is 2,839 terms and the median length is 2,378 terms. Let us now look at the distribution of relevant documents. As mentioned in Section 2.5, we define as relevant a document which contains a highly exhaustive element. The distribution of relevant documents over the length bins is shown in Figure 4.3 (bottom). (See Figure 4.4 for individual topic sets). The distribution of relevant documents is quite similar to the overall length distribution of documents. There is some length bias in the assessment set, but not as extreme as we saw in the element case (Figure 4.1).

Table 4.3 shows in more detail the statistics of the relevant documents. We see that the length of the "average relevant document" is close to the length of the "average document." This is quite different from the element retrieval case (Table 4.2).

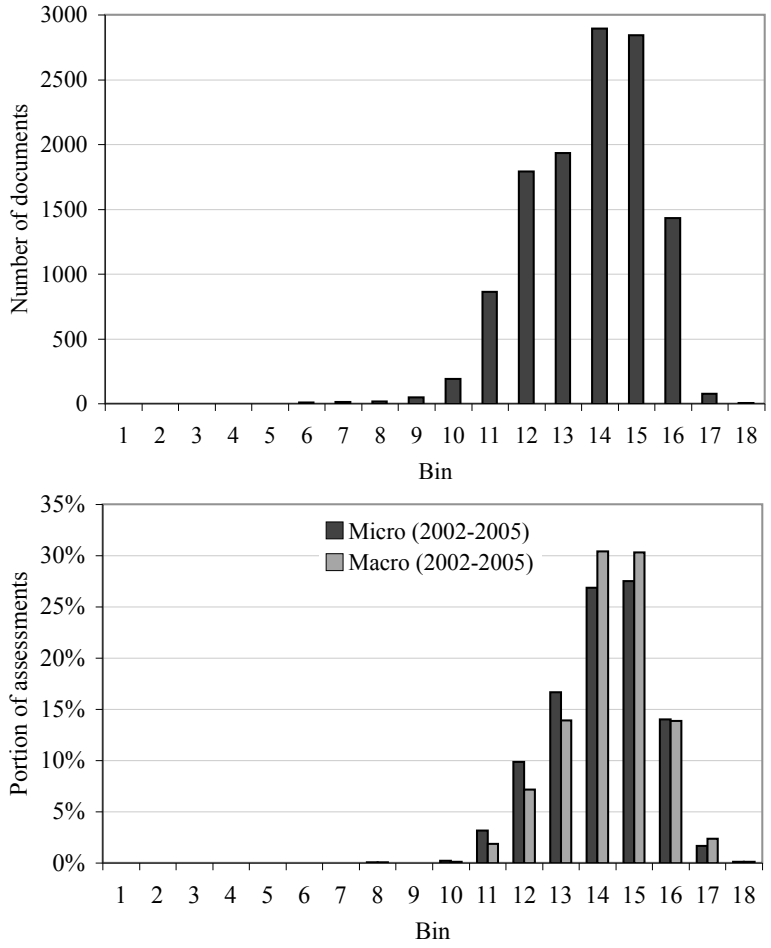


Figure 4.3: Top: Length distribution of documents in the INEX IEEE Computer Society Collection. Bottom: Length distribution of articles containing a highly exhaustive element in the INEX 2002–2005 assessments. Micro: the fraction of elements in each bin. Macro: the fraction of elements in each bin, averaged over topics.

The main lesson we can draw from this section is that one of the main differences between element retrieval and document retrieval is the relation between the length distribution of retrievable items and relevant items.

- For document retrieval the distributions are similar, with a small bias towards longer documents in the set of relevant items.
- For element retrieval the distributions are completely different, with a great length bias in the set of relevant items.

In the following section we will look at how we can incorporate this length bias into our element retrieval framework.

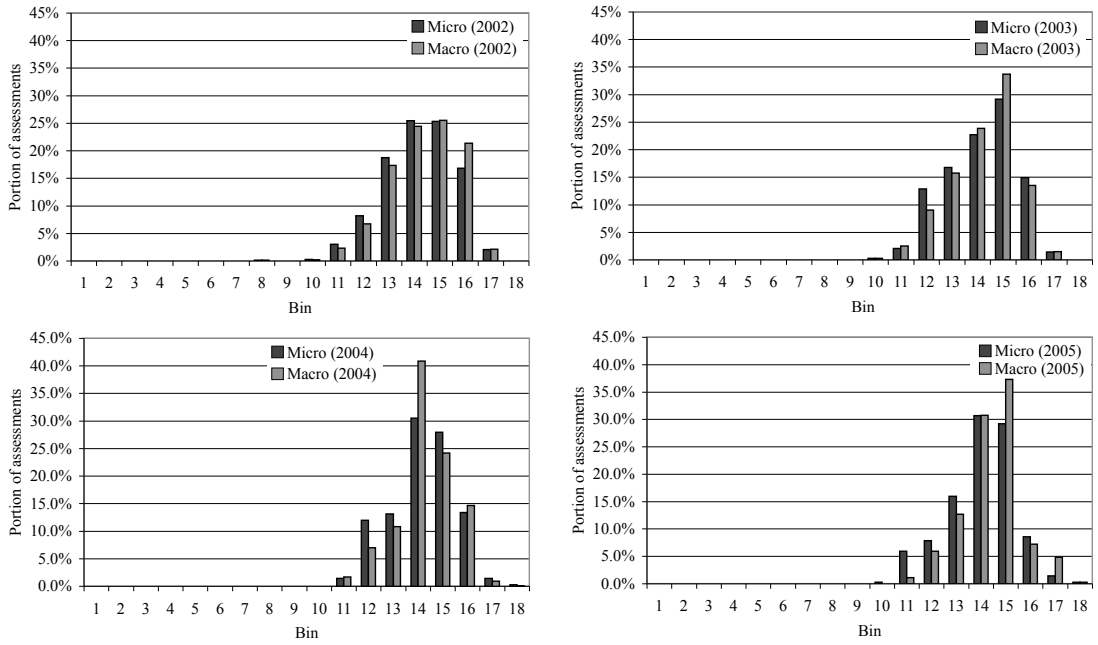


Figure 4.4: Length distribution of documents having an element assessed highly exhaustive in the INEX 2002–2005 collections. Micro: the fraction of elements in each bin. Macro: the fraction of elements in each bin, averaged over topics.

4.2 Length Priors

We have seen in Section 4.1.2 that there is a length bias in the assessments, in favor of longer elements. Furthermore, we have seen in Section 3.5 that the smoothing parameter can work as a length-bias generator. In this section we will see what happens if we introduce this length-bias explicitly using length priors. This sort of length normalization has proved to be effective for collections where there is a length bias in the assessments [Singhal et al., 1996a]. In the language-model framework, length priors—and other prior probabilities—have been used successfully in other retrieval tasks, such as entry page search [Kraaij et al., 2002].

In our implementation of length priors, we follow Kraaij et al. [2002] and assign a prior probability to an element e using a linear function of the element length:

$$P(e) = C \cdot |e|, \tag{4.1}$$

where $|e|$ is the number of terms in e and C is a constant that can be ignored in the ranking formula. We update our scoring formula (Equation 3.6) to incorporate the length prior,

$$s_{prior}(e, q) = \log(|e|) + s(e, q). \tag{4.2}$$

We refer to this length prior as the *baseline length prior*.

In addition to the baseline length prior, we introduce another prior which

Table 4.4: *Baseline length prior*: Optimal value for the smoothing parameter λ, based on both MAP and MAep. Improvement is calculated relative to the corresponding results for the uniform element prior (Table 3.3 on page 49).

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
Title	.15	.0882	66%***	.25	.0957	35%***	.20	.0729	36%***
Desc.	.15	.0819	89%**	.15	.0793	34%***	.15	.0623	36%***
T+D	.80	.0699	15%***	.70	.0890	11%***	.55	.0671	12%***

we refer to as the *flexible length prior*. The flexible length prior is a non-linear function of the element length:

$$P(e) = C \cdot |e|^{\beta}. \tag{4.3}$$

We implement this length prior by introducing a parameter β into our scoring formula (Equation 4.2). Our new scoring formula then becomes:

$$s_{prior}(e, q) = \beta \cdot \log(|e|) + s(e, q). \tag{4.4}$$

We refer to β as the *length prior parameter*. The goal of the introduction of the baseline length prior and the flexible length prior is to demonstrate the importance of length normalization for XML element retrieval. Experiments with other types of length priors—and more generally other types of priors—is left as future work.

4.2.1 Experiments

Baseline length prior

First we look at the effect of the baseline length prior where we set $\beta = 1.0$. We look at the effect of the length prior for different values of the smoothing parameter λ. Optimal values for the smoothing parameter can be seen in Table 4.4. We compare our baseline length prior settings to the optimal settings of the smoothing parameter when a uniform prior was used (Section 3.4). When a length prior is used, the optimal value of the smoothing parameter moves to the lower range for both title-only and description-only query formats. For the combined title and description format the optimal settings stay at the higher end. The optimal value for the smoothing parameter has moved to the expected range for the title-only and the description-only queries, but the combined title and description query format still gives unexpected results—the λ is still high.

As for the effectiveness of the baseline length prior, we see from the table that for all query formats and all quantizations the baseline length prior runs significantly outperform the runs where a uniform element prior was used. The improvement is far greater for the title-only and description-only query formats, than for the combined query format.

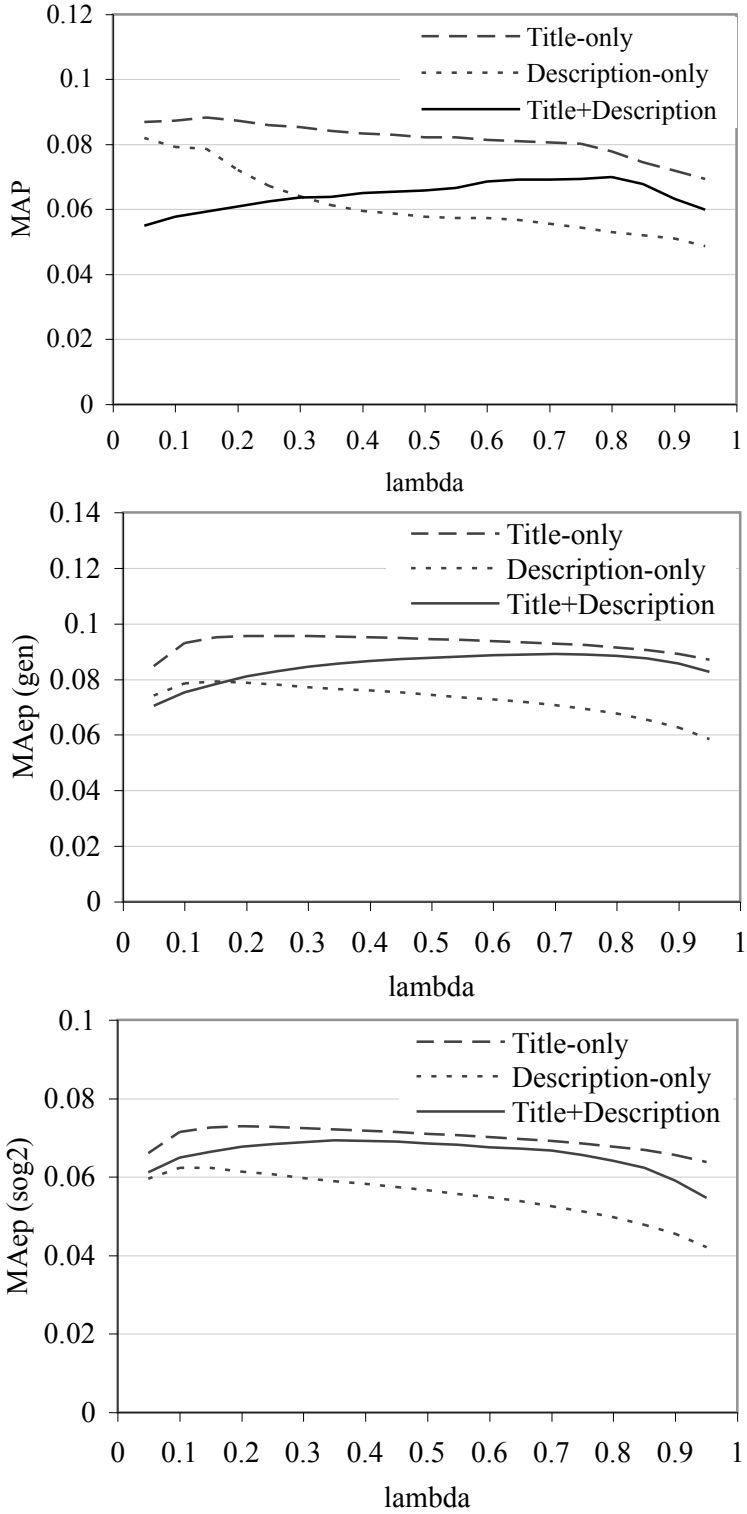


Figure 4.5: *Baseline length prior*: Mean average (effort) precision for different values of the smoothing parameter.

Table 4.5: *Baseline length prior*: Optimal value for the smoothing parameter λ, for each "vintage" of the INEX collection, based on both MAP and MAep. We report three quantizations: strict, generalized (gen), and specificity-oriented generalized (sog2). Improvements are calculated relative to the corresponding results for the uniform element prior (Table 3.4 on page 50).

(a) Using the *title* field

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
2002	.20	.0709	84%**	.95	.0932	58%***	.80	.0540	51%***
2003	.15	.1215	113%*	.30	.0834	40%**	.25	.0672	40%**
2004	.05	.1062	60%	.20	.1242	31%***	.15	.0851	36%***
2005	.05	.0539	0.7%	.15	.0817	27%*	.15	.0817	27%*

(b) Using the *description* field

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
2002	.05	.0573	40%	.25	.0834	21%	.05	.0505	28%
2003	.10	.1196	111%*	.25	.0834	41%***	.15	.0679	48%**
2004	.05	.0999	35%	.15	.0928	30%***	.15	.0680	33%***
2005	.15	.0483	2.1%	.10	.0608	24%*	.10	.0608	24%*

(c) Using both the *title* and *description* fields

	strict			gen			sog2		
	λ	MAP	Impr.	λ	MAep	Impr.	λ	MAep	Impr.
2002	.95	.0606	15%	.90	.0995	10%***	.85	.0598	10%***
2003	.80	.0909	17%**	.85	.0870	13%***	.85	.0691	14%***
2004	.95	.0859	15%*	.50	.1091	11%***	.35	.0756	14%***
2005	.10	.0503	1.1%	.35	.0693	11%***	.35	.0693	11%***

Figure 4.5 shows in more detail the effect of changing the smoothing parameter. From the figure we clearly see the difference between the effect on the title-only and description-only query formats—which peak with λ in the lower range—and the effect on the title+description query format—which peaks with λ in the upper range. The form of the curve for the title+description query format is similar to the curves we saw in the previous chapter when we studied the uniform element prior (Figure 3.5). This might indicate that the title+description query format requires more length bias than the shorter query formats. We will look at this when we study the flexible length prior experiments below.

If we compare the performance of different topic formats we see that the title-only query format outperforms the other two. This is different from the case where a uniform prior was used where the combined query format outperformed the others.

Vintage Table 4.5 shows the optimal settings for the smoothing parameter for our runs using the baseline length prior. We see that the baseline length prior generally improves performance compared to the case where an uniform prior was used. The 2005 vintage is again different from the other years. For the 2005 collection there is hardly any improvement of using the length prior if we measure performance in terms of strict quantization and mean average precision. For the generalized quantizations there is, however, a considerable improvement. For the strict quantization the improvements for individual topic sets are seldom statistically significant. For the generalized quantizations significant improvements are more frequent. There are at least two factors that may play a role here. First of all, there may be more variance in the strict quantization of the assessment than the generalized quantization. The strict quantization considers only a fraction of the assessment values while the generalized quantizations consider all assessment values. The strict quantization might thus be more sensitive to the variance in "assessment style" of different assessors. The other factor that may play a role here is the number of assessed topics. For the strict quantization there are on average 25 topics in each vintage while for the generalized quantizations there are on average 30 topics in each vintage. The difference between the quantizations lies in the topics which do not contain any element which is both highly exhaustive and highly specific (see Table 2.3 for more details on per-vintage topic counts).

Overfitting As for the experiments in the previous chapter we address the risk of overfitting by looking at the performance of using the overall optimal parameter settings for each vintage of the INEX collection. The evaluation results are shown in Table 4.6. We see that the improvements continue to be significant for a majority of the topic sets, even if we use the overall optimal parameter settings.

Summary We can summarize the effect of the baseline length prior in several observations:

- Using the baseline length prior gives a significant improvement in terms of all quantizations using mean average (effort) precision
- The optimal value for the smoothing parameter, λ, is usually in the lower range when the baseline length prior is used.
- The title-only and description-only query formats profited much more from the baseline length prior than the combined title and description query format
- Again, the 2005 topics are an exception, as there is only a negligible improvement of using the baseline length prior when measured in terms of the strict quantization and mean average precision

Table 4.6: *Baseline length prior*: Performance for each *vintage* of the INEX collection, using the *overall* optimal parameter settings. Improvement is calculated relative to the corresponding results for the uniform element prior (Table 3.5 on page 53).

(a) Using the *title* field

	strict ($\lambda = 0.15$)		gen ($\lambda = 0.25$)		sog2 ($\lambda = 0.20$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0699	82%**	.0895	51%***	.0525	47%***
2003	.1215	113%*	.0833	40%**	.0672	40%**
2004	.1056	60%*	.1237	30%***	.0851	36%***
2005	.0531	8.1%	.0814	27%**	.0817	27%**

(b) Using the *description* field

	strict ($\lambda = 0.15$)		gen ($\lambda = 0.15$)		sog2 ($\lambda = 0.15$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0488	19%	.0785	30%**	.0494	34%*
2003	.1191	125%*	.0828	44%***	.0679	52%**
2004	.0941	27%	.0928	31%***	.0680	33%***
2005	.0483	679%	.0601	28%*	.0601	26%*

(c) Using both the *title* and *description* fields

	strict ($\lambda = 0.80$)		gen ($\lambda = 0.70$)		sog2 ($\lambda = 0.55$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0596	17%**	.0960	10%***	.0564	9.3%**
2003	.0909	16%**	.0758	13%***	.0664	14%**
2004	.0807	20%	.1071	11%***	.0745	14%***
2005	.0465	1.1%	.0666	9.3%***	.0681	10%***

Table 4.7: *Flexible length prior*: Optimal values for the smoothing parameter λ and the length-prior parameter β. Improvement is calculated relative to the corresponding results for the baseline length prior ($\beta = 1.0$) (Table 4.4 on page 70)

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
Title	.20	1.5	.1000	13%	.45	1.5	.0965	0.9%	.20	1.0	.0729	–
Desc.	.25	2.0	.0827	1.0%	.15	1.0	.0793	–	.15	1.0	.0623	–
T+D	.15	3.0	.1031	47%**	.25	2.0	.1011	14%***	.15	2.0	.0775	15%***

Before we discuss the effect of length priors in more detail, let us look at the effect of the flexible length prior.

Flexible length prior

Let us now turn our attention to the effect of changing the length prior parameter β. Table 4.7 shows the optimal settings for both the smoothing parameter and our length-prior parameter. The optimal values for the smoothing parameter differ slightly for different topic formats, but are all quite low. I.e., considerable smoothing is required. The value is slightly higher for the title-only query format than for the verbose formats. These results are in sync with results from Zhai and Lafferty [2004]. The optimal values for the length-prior parameter also differ between the topic formats. Using the strict quantization the title-only queries require the lowest value and the combination of title and description require the highest value. That is, the longer query formats require a higher value for the length prior parameter than the shorter query formats. The performance improvement is, however, only significant in the case of the combined query format. Using the generalized quantization (gen) the flexible length-priors did not improve over the baseline length prior for the description-only query format. One can say that the same hold for the title-only query format as well, since the improvement of using the flexible length prior is very small. For the combined query format the improvement of using the flexible length prior is again significant. Using the specificity-oriented quantization (sog2) the flexible length prior does not give improvements for the title-only and description-only query formats. For the combined query format the flexible length-prior again gives a significant improvement when compared to the baseline length-prior.

Figure 4.6 shows in more detail the effect of changing the smoothing parameter, using the optimal length-prior settings. The main difference between the curves in Figure 4.6 and the corresponding curves for the baseline length prior (Figure 4.5) is that when using the flexible length prior the title and description runs "catch-up" with the title only runs and their optimal smoothing parameter settings move to the lower range—i.e., less smoothing is required.

Let us compare the optimal length prior parameter settings between different

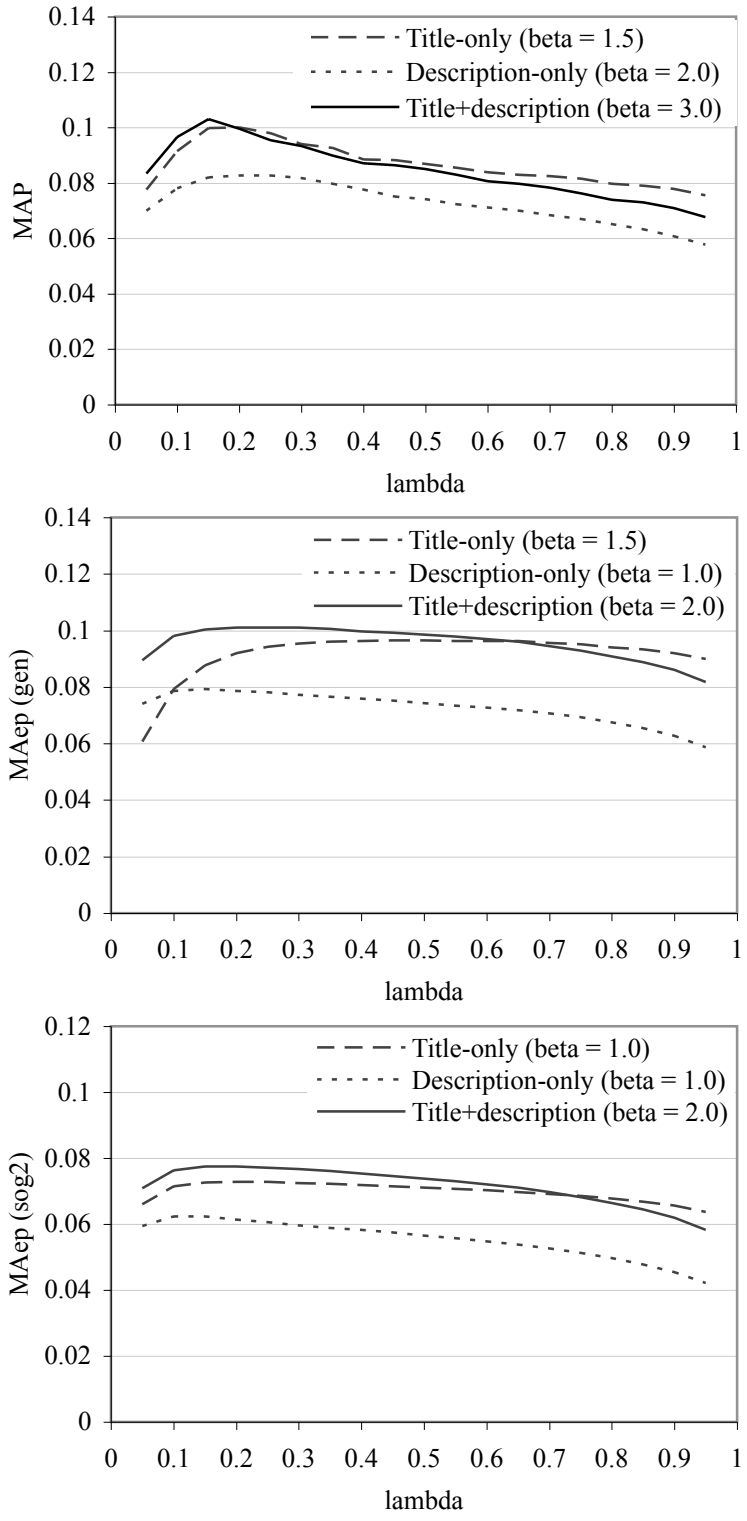


Figure 4.6: *Flexible length prior*: Mean average precision for different values of the smoothing parameter λ. Note that there are different values of β for different topic formats.

Table 4.8: *Flexible length prior*: Optimal values for the smoothing parameter λ and the length-prior parameter β. Improvements are calculated relative to the corresponding results for the baseline length prior ($\beta = 1.0$) (Table 4.5 on page 72)

(a) Using the *title* field

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.25	1.5	.0881	24%	.85	1.5	.1085	16%	.75	1.5	.0618	14%
2003	.30	1.5	.1277	5%	.65	1.5	.0934	12%	.35	1.5	.0736	9.5%
2004	.10	2.0	.1434	35%	.20	1.0	.1242	–	.15	1.0	.0851	–
2005	.05	0.5	.0556	3%	.15	1.0	.0817	–	.15	1.0	.0817	–

(b) Using the *description* field

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.15	2.0	.0697	22%	.25	1.5	.0816	3.0%	.05	1.0	.0505	–
2003	.20	1.5	.1201	0.4%	.25	1.5	.0867	3.9%	.25	1.5	.0700	3.0%
2004	.15	2.0	.1602	60%	.15	1.0	.0928	–	.15	1.0	.0680	–
2005	.15	1.0	.0583	–	.10	1.0	.0608	–	.10	1.0	.0608	–

(b) Using both the *title* and the *description* field

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.15	3.0	.0894	48%*	.40	2.5	.1098	10%	.20	2.5	.0658	10%
2003	.15	2.0	.1461	61%	.40	2.5	.1048	20%**	.35	2.5	.0838	21%*
2004	.10	3.5	.1642	91%*	.20	2.0	.1213	11%**	.15	2.0	.0850	12%*
2005	.10	2.5	.0542	8%	.10	2.0	.0776	12%	.10	2.0	.0776	12%

query formats. The verbose query format requires a higher value for the length prior parameter than the two less verbose query formats. We conjecture that this is an artifact of the simplification assumptions made in Sections 3.3 and 4.2 when we derived our retrieval formula (see e.g. [Cooper, 1995] for a discussion on inconsistencies and misidentified modeling assumptions in probabilistic information retrieval). In particular, the "score mass" that is accumulated in equation 3.6 increases as we add more terms to the query. Consequently, the "score mass" of the length-prior factor in equation 4.4 has more effect on the short query formats than on the verbose query format. Hence—in order to get the same length-bias effect—a higher length prior parameter value is needed for the verbose query format than for the shorter query formats.

Vintage Table 4.8 shows the optimal values for the smoothing parameter and the length-prior parameter for each vintage of the INEX collection.

Table 4.9: *Flexible length prior*: Performance for each *vintage* of the INEX collection, using the *overall* optimal parameter settings. Improvements are calculated relative to the corresponding results for the baseline length prior ($\beta = 1.0$) (Table 4.6 on page 74)

(a) Using the *title* field

	strict ($\lambda = 0.20$, $\beta = 1.5$)		gen ($\lambda = 0.45$, $\beta = 1.5$)		sog2 ($\lambda = 0.20$, $\beta = 1.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0873	25%	.1030	15%	–	–
2003	.1261	3.8%	.0915	9.8%	–	–
2004	.1311	24%	.1139	-7.9%**	–	–
2005	.0541	-1.5%	.0750	-7.9%*	–	–

(b) Using the *description* field

	strict ($\lambda = 0.25$ $\beta = 2.0$)		gen ($\lambda = 0.15$ $\beta = 1.0$)		sog2 ($\lambda = 0.15$ $\beta = 1.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0677	37%	–	–	–	–
2003	.1012	-15%	–	–	–	–
2004	.1556	65%	–	–	–	–
2005	.0066	-86%	–	–	–	–

(b) Using both the *title* and the *description* field

	strict ($\lambda = 0.15$ $\beta = 3.0$)		gen ($\lambda = 0.25$ $\beta = 2.0$)		sog2 ($\lambda = 0.15$ $\beta = 2.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0894	50%*	.1077	12%**	.0651	15%
2003	.1304	43%	.0983	30%**	.0789	19%*
2004	.1456	80%*	.1211	13%***	.0850	14%*
2005	.0460	-1.1%	.0754	13%***	.0773	14%**

For the title-only query format a moderate value of the length-prior parameter ($\beta = 1.5$) results in improvements for the earlier vintages (2002 and 2003). This observation holds for all quantizations. Using the generalized quantizations we see that the baseline length prior is optimal for the 2004 and 2005 topics.

For the description-only query format the per-vintage results are similar to the ones for the the topic-only query format.

For the combined query-format we see that the flexible length-prior improves performance compared to the baseline length-prior. This holds for all quantizations and all vintages. Our observation above—that the verbose topic format needs higher value for the length prior parameter—carries over to individual vintages and quantizations.

Overfitting Table 4.9 shows the effect of using the overall optimal parameter settings on the individual vintages of the INEX collection.

For the title-only and description-only query formats, the results of using the optimal parameter settings are mixed, ranging from significant decrease in performance to a large percentage-wise—yet non-significant—improvement. The optimal settings do not carry over to individual vintages. In particular, when we find the value of $\beta = 1.5$ to be optimal for the generalized quantization we are overfitting to the 2002 and 2003 vintages where we improve the performance. The cost of this overfitting is a significant decrease in performance for the 2004 and 2005 vintages.

For the combined topic format the results do, however, carry over to individual vintages. The flexible length prior give significant improvement for most of the vintage-quantization pairs. An exception—as so often before—is the 2005 vintage using the strict quantization. See Section 3.5 for a discussion of this phenomena.

Summary Let us now summarize our main observations from the experiments with the flexible length prior.

- Shorter query formats do not gain significantly from the use of a flexible length prior, compared to the baseline length prior.
- The most verbose query formats do gain significantly from the use of a flexible length prior, compared to the baseline length prior.

4.2.2 Discussion

Let us first summarize the results of using length priors.

- Baseline length priors give significant improvements, compared to the case where an uniform prior is used.
- Flexible length prior give significant improvements only for the most verbose query format.

Figure 4.7 shows the average length of the top-10 retrieved elements when we use the baseline length prior; different query formats and different values for the smoothing parameter λ. Additionally, the figure shows the average length for using the square length prior ($\beta = 2.0$) and the title+description query format. The figure shows that for small values of the smoothing parameter the baseline length prior has more length-bias effect on the title-only query format than on the more verbose query formats. The figure also shows that, for low values of the smoothing parameter, a value $\beta = 2.0$ for the length prior parameter in our title+description runs has a similar length bias effect as the baseline length prior has in our title-only runs. These observations support our conjecture from the previous section that the effect of the length prior parameter depends on the verboseness of the query format. We refer back to Section 3.5 for more discussion

Table 4.10: Most frequent tag-names appearing in the top 10 results. The Avg. freq. is the number of elements in the top-10 that have the corresponding tag, averaged over all topics. Standard deviation is shown in brackets. The table in boldface shows the results for the optimal length-prior parameter settings for the corresponding query format (using the sog2 quantization).

(a) Title-only queries

$\lambda = 0.15$, $\beta = 1.0$			$\lambda = 0.15$, $\beta = 2.0$		
Tag	**Avg.**	**freq.**	Tag	Avg.	freq.
article	**1.76**	**(1.67)**	article	5.32	(1.83)
p	**1.66**	**(1.69)**	bdy	2.72	(1.07)
bdy	**1.14**	**(1.13)**	sec	0.76	(0.98)
sec	**1.10**	**(0.98)**	bm	0.25	(0.53)
atl	**0.84**	**(1.90)**	ss1	0.18	(0.47)
ss1	**0.46**	**(0.73)**	p	0.16	(0.53)
ip1	**0.38**	**(0.62)**	atl	0.10	(0.79)
it	**0.38**	**(1.25)**	bib	0.08	(0.38)
bm	**0.24**	**(0.50)**	bibl	0.08	(0.35)
ti	**0.21**	**(0.82)**	app	0.07	(0.25)

(b) Description-only queries

$\lambda = 0.15$, $\beta = 1.0$			$\lambda = 0.15$, $\beta = 2.0$		
Tag	**Avg.**	**freq.**	Tag	Avg.	freq.
p	**2.38**	**(1.71)**	article	4.42	(1.94)
sec	**1.47**	**(1.25)**	bdy	2.64	(1.25)
article	**1.16**	**(1.33)**	sec	1.10	(1.16)
bdy	**0.85**	**(0.96)**	p	0.47	(0.87)
ss1	**0.68**	**(0.89)**	bm	0.32	(0.58)
ip1	**0.62**	**(0.78)**	ss1	0.31	(0.59)
atl	**0.51**	**(1.50)**	ip1	0.12	(0.34)
it	**0.20**	**(0.68)**	atl	0.08	(0.64)
bm	**0.18**	**(0.42)**	bib	0.08	(0.36)
ti	**0.17**	**(0.64)**	app	0.08	(0.29)

(c) Title+description queries

$\lambda = 0.15$, $\beta = 1.0$			$\lambda = 0.15$, $\beta = 2.0$		
Tag	Avg.	freq.	**Tag**	**Avg.**	**freq.**
p	2.57	(1.88)	**article**	**1.86**	**(1.50)**
atl	1.15	(2.04)	**p**	**1.63**	**(1.64)**
sec	0.92	(1.02)	**sec**	**1.44**	**(1.09)**
it	0.64	(1.37)	**bdy**	**1.32**	**(1.12)**
ip1	0.61	(0.81)	**ss1**	**0.60**	**(0.80)**
ss1	0.49	(0.74)	**atl**	**0.49**	**(1.50)**
bdy	0.32	(0.66)	**ip1**	**0.43**	**(0.63)**
bb	0.31	(0.96)	**bm**	**0.29**	**(0.58)**
article	0.29	(0.62)	**it**	**0.20**	**(0.68)**
st	0.29	(0.70)	**bb**	**0.15**	**(0.60)**

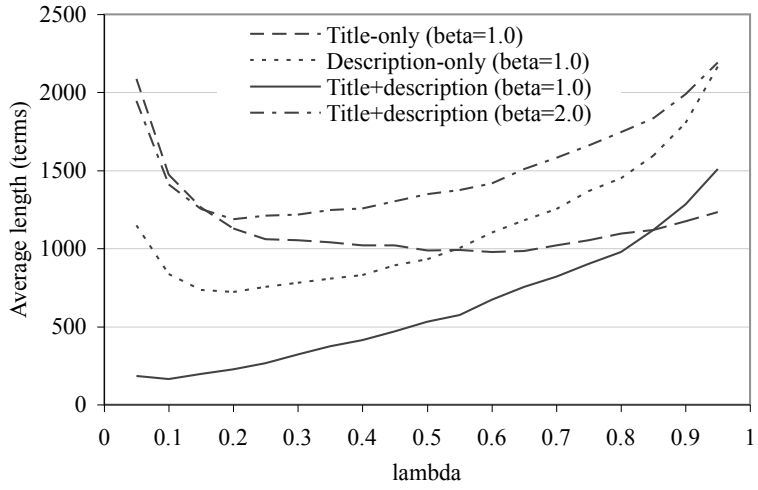


Figure 4.7: Average length of the top-10 results of runs using different value for the smoothing parameter λ

about the relation between verboseness of query format and length of retrieved elements.

Table 4.10 shows the most frequent tag-names among the top-10 retrieved elements of a selection of retrieval runs. We choose to show the result when using the value $\lambda = 0.15$ for the smoothing parameter and the values $\beta = 1.0$ and $\beta = 2.0$ for the length-prior parameter. The table gives a different view on our conjecture that the baseline length-prior parameter has a stronger length-bias effect on the title-only query format than on the more verbose query formats—articles and bodies are more prominent in the title-only run than for the other topic format. The table also shows that the square length prior ($\beta = 2.0$) has a similar length bias effect on the title+description query format as the baseline length prior has on the title-only query format.

Let us now compare our results to previously published results. The length prior experiments in this chapter are an extension of previously published work [Kamps et al., 2004a, 2005a]. Those publications were based on the 2002 and 2003 vintages of the INEX test collection. It used only the strict quantization and the combined query format of the title and description fields. Based on this setup we found the flexible length-prior (called "extreme length-prior" in the above publications) to be very effective. How do our results carry over to later vintages of the INEX test collection, to different quantizations, and to different topic formats?

Quantization First we look at the case where we keep the topic format and vintage fixed and look at the performance for different quantizations. We see that the flexible length prior continues to be effective when evaluated using different quantizations.

Vintages Now, let us see what happens if we keep the topic format and quantization fixed, but look at different vintages. We see that if we use the strict quantization our results carry over to the 2004 collection, but not to the 2005 collection. If we use the generalized quantizations (gen and sog2) our results carry over to both 2004 and 2005 collections.

Query-format Finally we look at the effect of changing the query format. As mentioned before, the flexible length prior does not prove to be effective for the title-only or the description-only query formats. More precisely, the length bias effect of the flexible length-prior is dependent on query verboseness—the more verbose the query is, the less is the length bias effect. Effectively, this means that the baseline length-prior provides sufficient length bias for the title-only query format, but more length bias is needed for the title+description query format.

In sum, our results from [Kamps et al., 2004a, 2005a] do thus—to a large degree—carry over to new topic/assessment sets and different quantizations. The results do, however, not carry over to different query formats.

4.3 Conclusions

In this chapter we have looked at length normalization for XML element retrieval. We have studied the length of relevant elements, and the effect of length-priors. We will now summarize our main conclusions and look at the road ahead.

In the introduction we asked ourselves:

> What are the characteristics of element assessments in terms of length?

We have seen that the length distribution of relevant elements is radically different from the length distribution of elements in the collection as whole—the relevant elements are on average longer than elements in general. We also asked ourselves about the difference between element retrieval and document retrieval.

> How is the length distribution of relevant elements different from the distribution of relevant documents?

We have seen that the overall distribution of document lengths is close to being a normal distribution, while the distribution of element lengths is a very skewed distribution—where most elements are very short. The distribution of both relevant elements and relevant documents resembles a normal distribution. The difference between elements in general and relevant elements is thus greater than the difference between general documents and relevant documents.

Additionally, we have shown that the 2005 strict assessments are very different from the assessments of the previous years. The main difference is that the longest elements in the collection are no-longer considered as being both highly exhaustive and highly specific.

In our introduction we also asked ourselves about the effect of length-priors:

> Can we improve the effectiveness of element retrieval by using length priors?

We have shown that our baseline length-prior gives significant improvements for all three query formats and three quantizations. For short queries, flexible length priors are not significantly better than baseline length priors.

Recall the main research question of this chapter:

> What is the impact of length normalization for XML retrieval?

In short, length normalization is important for XML retrieval. Compared to document retrieval—where length normalization is also important—the significance of length is of greater importance for XML retrieval.

Let us now take a peek at some of the things that await us in the following chapters. We have seen that length normalization is very important for achieving good retrieval performance. In the next chapter we take a closer look at the notion of "unit of retrieval" and ask ourselves:

> Are all element types equally important retrieval units?

In this chapter we have looked at elements as atomic units. We have not looked at element context.

> Can we make use of element context to improve retrieval effectiveness?

We study this question in Chapter 6 where we introduce a *mixture model* that estimates element relevance by combining term statistics from three sources, the element itself, its surrounding document, and the collection as a whole.

So-far we have evaluated element retrieval as an abstract task. We have not discussed how to put it into action to serve users. We will address this issue in Chapter 8, where our main research question is:

> How do we put element retrieval into action as part of an operational system?

Chapter 5

The Unit of Retrieval

In the previous chapter (Chapter 4) we have seen that there is a length bias in the assessments—the "average relevant" element is longer than the "average element" in the collection as a whole. Furthermore, we have seen in Section 4.2 that element length priors are useful for introducing a bias toward the "appropriate" unit of retrieval. The length priors are effective for moving the retrieval bias from shorter elements to longer ones. In this chapter we take a closer look at the relation between element characteristics and retrieval performance. Our main research question is:

> Are all element types equally important retrieval units?

We address this question both by analyzing the INEX assessments and by doing retrieval experiments.

In Section 5.1 we follow-up on our assessment analysis from the previous chapter and ask ourselves:

> What are the characteristics of the element assessments in terms of tag-names?

We find that mid-sized elements—such as sections and paragraphs—are the elements most frequently assessed as both highly exhaustive and highly specific.

In Section 5.2 we introduce selective indexing strategies for building element indices [Sigurbjörnsson and Kamps, 2006]. That is, we choose to index—and hence retrieve—only certain types of elements. In our index building we address two sub-questions of our main question:

> What is the effect of excluding *short* elements from our index?

and

> What is the effect of excluding *long* elements from our index?

Table 5.1: The 20 longest element types in the INEX collection. The statistics are based on the overlapping element index.

Tag	Description	Mean length	Collection frequency	Document frequency
article	article	2839.04	12,107	12,107
index	journal index	2572.29	117	53
bdy	document body	2331.57	12,107	12,107
bm	back-matter	517.86	10,058	10,058
sec	section	406.35	69,728	11,961
dialog	dialog	396.15	194	61
bib	bibliography	311.21	8,543	7,489
bibl	bibliography	310.99	8,551	7,492
ss1	(sub)section	224.75	61,454	7,647
app	appendix	220.44	5,856	3,792
ss3	(sub)section	153.05	127	27
ss2	(sub)section	151.89	16,276	2,702
proof	proof	105.10	3,765	587
numeric-list	list	79.89	54	25
fm	front-matter	77.83	12,107	12,107
dl	definition list	77.56	353	266
numeric-rbrace		66.68	114	37
tgroup	table	62.80	5,816	1,964
l4	list	62.49	115	56
tbody	table body	60.82	5,810	1,960

We also want to know how far we can go in our selective indexing strategies. We look at an index containing only "top-level sections"—i.e., sections at the top-level of the section hierarchy. In XML terminology, top-level sections are sections that do not have any section ancestors. We then ask ourselves the question:

What is the effect of retrieving only top-level sections?

In short, our experiments—in Section 5.3—reveal that we can safely exclude the short elements from our index, but excluding the longest ones significantly degrades our retrieval performance. Retrieving only top-level sections is not a successful retrieval strategy.

Finally, we close the chapter with discussion and conclusions. In Section 5.4 we provide an in-depth discussion of our results and in Section 5.5 we conclude.

Table 5.2: The 20 most frequent tags in the INEX collection. The statistics are based on the overlapping element index.

Tag	Description	Collection frequency	Document frequency	Mean length
p	paragraph	736,112	11,934	27.88
tmath	math	403,872	4,935	2.43
ref	reference	391,519	8,943	1.32
it	italicized text	388,511	11,077	2.65
au	author name	317,346	10,537	1.61
snm	surname	311,326	10,355	1.05
entry	table entry	233,775	1,961	1.48
ip1	paragraph	175,233	10,270	26.79
obi	other bib info	157,905	11,702	3.53
ti	title	155,681	12,105	4.29
pdt	publication date	154,978	12,107	1.46
yr	year	154,943	12,107	1.00
bb	bibliography item	149,167	7,494	17.77
sub	subscript	145,962	3,195	1.01
atl	article title	134,068	11,924	5.79
fnm	first name	125,493	10,491	1.03
b	boldface text	120,112	9,674	2.73
st	section title	113,382	10,733	2.92
pp	pages (in bibliography)	108,134	12,105	2.06
scp	smallcaps	107,429	4,433	1.03

5.1 Tag-names of Relevant Elements

In this section we take a look at the characteristics of the INEX element assessments in terms of tag-names being assessed as both highly exhaustive and highly specific. This section is a follow-up on Section 4.1 where we looked at the characteristics of the INEX element assessments in terms of length. For the analysis in this section we will use the same experimental setup as described in Section 4.1.

What sort of tag-names are used in the collection? Let us look at the frequency of different tag-names. In total, there are 176 tag-names used in the collection. Looking at all those tag-names in detail is too complex for our purpose. Instead we choose two smaller sets of tag-names which we will study in more detail. On the one hand, we study the tag-names of the longest elements in the collection. We consider this to be a representative sample of the "shallow" part of the document trees, i.e., the neighborhood of the root. On the other hand, we study the tag-names that appear most frequently in the collection. We consider this to be a representative sample of the "deeper" part of the collection, i.e., the neighborhood of the leaves. Table 5.1 shows the 20 longest element-types in the collection, and

Table 5.2 the 20 most frequently occurring element-types in the collection. The tables list the tag-name of the element-type, a description of the element content, the frequency of the element-type in the collection, the document frequency of the element-type in the collection (i.e., the number of different documents in which the tag-name appears), and the average length of elements bearing the tag-name. We calculate our statistics from the viewpoint of the retrieval system, as explained in Section 4.1. That is, we use the statistics as they appear in our indices. This means that we only look at elements that are non-empty after text-processing.

From Table 5.1 we see that the largest building blocks of the collections are articles (<article>), bodies (<bdy>), back matter (<bm>), sections (<sec>, <ss1>, etc.), and bibliography (<bib>, <bibl>). Additionally, there are some less frequent elements such as indices of journals (<index>) and dialogs (<dialog>).

From Table 5.2 we see that many of the most frequent element-types do not contain much text. Out of the top-20 most frequent elements, only three can be considered as text containers, i.e., the paragraphs (<p> and <ip1>), and bibliography items (<bb>). The other frequently occurring element-types are unlikely to contain enough text to be considered—on their own—as a satisfying fulfillment of an information need. But, what do the assessments say?

Table 5.3 shows the 20 most frequent tag-names of elements assessed both highly exhaustive and highly specific in INEX 2002–2005. The table shows statistics over all 101 topics that have a strict assessment. We calculate statistics over all assessed elements, independent of whether or not they can be retrieved (some of the relevant elements do not contain any text and can thus not be retrieved by our system). We report the fraction of topics having a relevant element with the corresponding tag-name (Topics), the frequency as a percentage of all strict assessments (Micro), and the frequency as a percentage normalized by the total number of strict relevant assessments for each topic (Macro). In the micro distribution frequencies are not normalized for individual topics. This means that a topic with many relevant elements will have greater impact on the distribution than a topic with few relevant elements. In the macro distribution the normalization means that all topics have the same weight in the overall distribution. This means that a topic with few relevant elements has the same impact as a topic with many relevant elements (See Table 2.4 for more information about the number of relevant elements per topic).

From Table 5.3 we see that the assessors seem to prefer the text-rich tag-types such as articles (<article>); bodies (<bdy>); sections (<sec>, <ss1>, <ss2>); and paragraphs (<p>, <ip1>). Together, those tag-names cover 73% of the assessments (macro: 85%). Some of the short elements were also judged relevant, but less frequently than the longer elements. These short elements include section titles (<st>); article titles (<atl>); italicized words (<it>); and boldface words (<b>).

Table 5.4 shows the 15 most frequent tag-names of elements assessed both highly exhaustive and highly specific for individual years of the INEX test collection. When we compare the assessments between years, we see that the weight of

Table 5.3: 20 most frequent elements in the INEX 2002–2005 strict assessments. *Topics*: is the fraction of topics for which the tag-name is assessed strictly relevant. *Micro*: is the fraction of the tag-name in the total assessment set. *Macro*: is the fraction of the tag-name in the total assessment set, normalized by the number of relevant element for each topic.

Tag	Description	Topics	Micro	Macro
p	paragraph	72.3%	27.0%	25.6%
sec	section	83.2%	15.0%	19.5%
article	article	57.4%	9.3%	13.1%
ss1	section	69.3%	8.6%	12.3%
ip1	paragraph	48.5%	5.8%	4.3%
it	italicized text	16.8%	5.8%	1.4%
bdy	body	55.4%	5.5%	8.7%
ref	reference	11.9%	1.6%	0.5%
ss2	section	29.7%	1.6%	1.5%
sub	subscript	2.0%	1.4%	0.2%
fig	figure	21.8%	1.3%	1.4%
bb	bibliography item	11.9%	1.3%	0.5%
atl	article title	14.9%	1.2%	0.6%
b	boldface text	7.9%	1.1%	0.2%
st	section title	25.7%	1.1%	1.1%
tmath	mathematics text	2.0%	1.1%	0.1%
art	external image	11.9%	1.1%	0.9%
abs	abstract	23.8%	1.0%	0.9%
item	list item	15.8%	0.9%	0.8%
li	list	19.8%	0.8%	0.5%

articles in the assessments set has decreased dramatically over the years. In 2002, 19 out of 21 assessed topics had a highly exhaustive and highly specific article. In 2005, there was only 1 out of 27 topics for which an article was considered both highly exhaustive and highly specific. The weight of the other text-rich elements (sections, and paragraphs) has increased over the years. The sudden jump of the italics tag (`<it>`) in the 2004 assessments looks interesting. However, 288 of the 297 strictly relevant italics tags were for the same topic.

As mentioned before (Section 4.1.2), the differences in the assessments from one year to another may be explained by a combination of several factors. Let us recall the factors:

- First, the topics are different from one year to another.
- Second, the assessors' understanding of the notions of exhaustiveness and specificity can change between years.

Table 5.4: Most frequent tag-names in the 2002–2005 content-only strict assessment sets. *Topics*: the fraction of topics for which the tag-name is assessed strictly relevant. *Micro*: the fraction of the tag-name in the total assessment set. *Macro*: the fraction of the tag-name in the total assessment set, averaged over topic averages.

	2002 assessments				2003 assessments		
	Topics	Micro	Macro		Topics	Micro	Macro
p	73.9%	27.4%	22.2%	sec	92.6%	20.9%	23.4%
article	82.6%	22.1%	22.7%	p	77.8%	20.7%	19.4%
sec	91.3%	20.8%	26.6%	article	81.5%	11.8%	15.8%
ss1	73.9%	8.2%	7.2%	bdy	85.2%	11.5%	14.1%
bdy	60.9%	6.4%	8.7%	ss1	77.8%	10.1%	8.2%
ip1	47.8%	4.4%	2.9%	ip1	48.1%	4.7%	3.9%
ss2	34.8%	1.8%	0.9%	ss2	37.0%	2.4%	1.7%
abs	30.4%	1.6%	1.0%	fig	25.9%	2.2%	2.1%
fm	26.1%	0.9%	1.2%	bb	14.8%	1.5%	1.0%
st	30.4%	0.8%	0.8%	app	25.9%	1.3%	0.8%
item	21.7%	0.6%	0.3%	atl	18.5%	1.3%	1.0%
app	26.1%	0.5%	1.5%	bm	22.2%	1.2%	0.7%
it	17.4%	0.4%	0.3%	art	3.7%	1.2%	0.4%
li	17.4%	0.4%	0.2%	fm	22.2%	0.9%	0.5%
b	13.0%	0.4%	0.4%	li	18.5%	0.9%	0.8%

	2004 assessments				2005 assessments		
	Topics	Micro	Macro		Topics	Micro	Macro
p	64.0%	26.7%	18.4%	p	73.1%	37.8%	42.1%
it	24.0%	11.5%	1.7%	ip1	46.2%	13.8%	6.7%
sec	92.0%	10.2%	16.8%	ss1	50.0%	9.8%	17.3%
ss1	76.0%	7.5%	16.3%	sec	57.7%	9.6%	11.9%
ip1	52.0%	4.6%	3.6%	it	19.2%	7.0%	3.2%
article	64.0%	3.8%	10.8%	item	19.2%	2.7%	2.6%
bdy	72.0%	3.4%	11.4%	ariel	3.8%	2.6%	0.8%
sub	4.0%	3.4%	0.4%	ref	15.4%	1.8%	1.0%
ref	24.0%	3.2%	0.8%	fig	23.1%	1.6%	1.2%
tmath	4.0%	2.6%	0.3%	art	15.4%	1.5%	0.7%
b	8.0%	2.1%	0.3%	b	11.5%	1.2%	0.3%
bb	20.0%	2.1%	0.7%	url	7.7%	1.2%	0.7%
atl	24.0%	2.0%	0.9%	abs	23.1%	1.0%	1.8%
st	36.0%	1.7%	2.7%	article	3.8%	1.0%	3.8%
art	28.0%	1.4%	2.3%	ss2	2.7%	0.9%	0.5%

- Third, and most importantly, the assessment interface has changed from one year to another.

In Section 4.1.2 we observed that the length distribution of the 2005 assessments is radically different from the assessments in previous years—the longest elements in the collection are no longer considered as both highly exhaustive and highly specific. In the present section we see the same phenomena from a different angle. In the 2005 assessments the fraction of paragraphs amongst the relevant elements has increased, the fraction of (sub)sections has stayed somewhat the same, and the fraction of articles and bodies has decreased dramatically. I.e., as opposed to earlier years articles and bodies are no longer considered as both highly exhaustive and highly specific. As before, we conjecture that the change in assessments characteristics is related to the introduction of the "yellow marker" interface in 2005. Assessing specificity with a yellow marker is a considerably different (and, no doubt, easier) cognitive process than assigning to each element a specificity value. In the case of articles (and bodies), it is indeed harder to imagine an assessor "marking a full article as yellow" than it is to imagine that an assessor would assign a specificity value 3 to an article. And indeed the numbers in Table 5.4 bear witness to this.

The number of topics for which a particular element-type is considered strictly relevant tells us something about the consensus of how suitable this element type is as a retrieval unit. Let us look at the set of tag-names which appear in the assessments for at least half of the topics. Three tag-names appear in the assessments for more than two-thirds of the topics. Sections (`<sec>`) have the greatest topic coverage, followed by paragraphs (`<p>`) and sub-sections (`<ss1>`). Additionally, two tag-names appear in the assessments for more than half of the topics: articles (`<article>`) and bodies (`<bdy>`). All the elements with the greatest consensus are text-rich elements.

There are three tag-names which appear in this consensus set over all four years: sections (`<sec>`), subsections (`<ss1>`), and paragraphs (`<p>`). It is interesting to note that two of the three tag-names with the largest consensus, sections and subsections, appeared in the table of top-20 longest element types (Table 5.1) and the third, paragraphs, was the element with the greatest mean length in the table of top-20 most frequent element-types (Table 5.2).

Let us now summarize the main observations of the analysis of tag-name distribution.

- On average, the most frequent tag-names in the collection contain quite short texts.
- On average, the most frequent tag-names in the assessments contain longer texts.
- Sections, sub-sections, and paragraphs appeared in the strict assessments of more topics than did any other element. One can thus say there is some

sort of consensus about those element types being "appropriate" units of retrieval.

- The strict assessments in 2005 were very different from the assessments of previous years. The most noticeable difference was that complete articles and bodies were no longer considered as being both highly exhaustive and highly specific.

5.2 Selective Indices

In Sections 4.1 and 5.1 we observed that there is a gap between the "average element" in the collection and the "average relevant element" in the collection. We have seen that some element types are more likely to be assessed relevant than others. We have also seen that the "average relevant element" is longer than the "average element" in the collection. These two observations lead quite naturally to the question whether we can use these facts to create more economic text indices where we only index elements that have a reasonable likelihood of being relevant. Indexing only only a subset of the elements has two potential advances:

- Improved effectiveness, since we reduce the "noise" caused by elements that have little change of being judged as both highly exhaustive and highly specific.
- Improved efficiency, since our posting-lists get shorter as we remove elements this can help to speed up our retrieval system, and we make more efficient use of disk storage.

In this section we will build several selective indices where we index only a subset of the total number of elements in the collection. We base our selection based on two criteria, length and tag-name.

- Length based selection: only elements whose length is above a certain threshold are indexed.
- Tag-name based selection: only elements bearing certain tag-names are indexed.

Various types of selective indexing schemes have been used in INEX participations. Index reduction based on eliminating the very many very short elements has been used by several teams at INEX (E.g., [Sigurbjörnsson et al., 2004a, Sigurbjörnsson and Kamps, 2006, Fujimoto et al., 2006]). Gövert et al. [2003] used a predefined list of tag-names which was compiled after careful analysis of tag name semantics. Mass and Mandelbrod [2004] and Clarke and Tilker [2005] used existing relevance assessments to define the appropriate units for their index.

We build one index where we apply a length based selection. I.e., we build an index where the shortest elements are excluded.

len An index of elements whose length is above a certain threshold. The length of an element is determined by the number of terms contained in the element. The term count is applied before stopwords were removed. For the experiments reported in this thesis we use a threshold of 20. We experimented with a few different values for the threshold and found 20 to be the best performing one.

We use this index to answer the question how the removal of short elements affects retrieval performance.

In Section 5.1 we have shown that elements with certain tag-names are more likely than others to be regarded as relevant. Based on this analysis we can decide to index only tag-names that are reasonably likely to be judged relevant. In Section 5.1 we argued that sections and paragraphs are element-types that are frequently assessed as highly relevant and highly specific. These element-types also appear in the strict assessment set of a large number of topics. With this information in mind, sections and paragraphs will be the main building blocks of our indices. We build three indices where tag-name is used as a selection criteria.

sec An index of top-level sections. I.e., we index only elements of the type `<sec>`. This is a non-overlapping index of the collection.

sp An index of sections and paragraphs. We use the DTD to choose the appropriate elements. I.e., we index the elements that are referred to in the DTD as sections (`sec`, `ss1`, `ss2`, `ss3`) and paragraphs (`ilrj`, `ip1`, `ip2`, `ip3`, `ip4`, `ip5`, `item-none`, `p`, `p1`, `p2`, `p3`).

absp Articles and bodies were also frequent in the strict assessments (at least for the first few years of INEX). We build an index containing the section and paragraph elements listed above, with the addition of article elements (`article`) and body elements (`bdy`).

These indices have several roles. First, the section index (sec) is an attempt to see how far we can go in the simplification of our indices. The section index is also interesting for the sake that our indexing units do not overlap. Second, the section and paragraph index (sp) shows the effect of indexing the popular elements (sections and paragraphs). One would expect to get reasonable performance by retrieving the most popular elements. Third, the index of articles, bodies, sections, and paragraphs (absp) shows the role of article and body elements in the evaluation. We have seen that the occurrence of those elements has varied considerably between different vintages of the INEX strict assessments (See further Table 5.4). Comparing the results of the absp index and the sp index we can

Table 5.5: Properties of the the different indices. This table is based on the INEX 2002 IEEE collection. Stopwords were removed, but no stemming was applied. *Unit* stands for the number of retrievable units. *Storage* stands for the size occupied in physical storage.

	Units	Storage	% of coll.
Overlapping elements	6,157,724	1.5G	299%
Length based (len)	1,086,549	971M	189%
Articles etc. (absp)	1,146,639	798M	156%
Sections & paragraphs (sp)	1,122,425	538M	105%
Sections (sec)	69,728	163M	32%

answer the question how the retrieval performance is affected by excluding the longest elements from the index.

Note that we use the relevance assessments to motivate the choices of elements to include in our indices. Since we later evaluate the selective indices based on the same assessments, we are to some extent running the risk of overfitting to the assessments. The experiments in this chapter should be read with this issue in mind. The experiments show the general relation between retrieval units and performance, but do not provide a description of "how to choose elements".

Table 5.5 shows statistics of our selective indices. We report the number of retrievable units (Units), the amount of storage space needed for the index (Storage), and the size of the index as a percentage of the size of the original collection (% of coll.). The table shows that each of our indexing strategies considerably reduced the total number of indexed elements. Our full overlapping element index contained roughly six million elements; our overlapping selective indices contain roughly one million elements; and our non-overlapping section index contains within a hundred thousand elements. In terms of storage space the full overlapping index requires triple the size of the collection; the selective indices containing the long elements require space roughly equal to double the size of the collection; the section and paragraph index requires space which is roughly equal to the collection size; and the section index space requirements are only one third of the space needed to store the collection.

We do not explore the efficiency issues of selective indexing, but refer to our INEX 2005 proceedings paper for some preliminary efficiency results [Sigurbjörnsson and Kamps, 2006]. Efficiency of selective indices has also been studied by Fujimoto et al. [2006].

5.3 Experiments

In this section we will report results of using the selective indexing strategies We present our results in terms of the research questions presented in the introduction

Table 5.6: *Selective indexing*: Optimal values for the smoothing parameter λ and the length-prior parameter β for our selective indexing strategies. Results are for title-only topics. Improvements are calculated relative to corresponding results on the full index (Table 4.7 on page 75).

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
len	.20	1.5	.1013	1.3%	.20	1.0	.1041	7.9%***	.20	1.0	.0805	10%***
absp	.20	1.5	.0996	-0.4%	.20	1.0	.1022	5.9%*	.20	1.0	.0802	10%**
sp	.30	1.5	.0582	-42%***	.15	1.0	.0659	-32%***	.15	1.0	.0548	-25%***
sec	.20	1.5	.0349	-65%***	.15	1.0	.0292	-70%***	.15	1.0	.0233	-68%***

to this chapter. First we consider:

What is the effect of excluding *short* elements from our index?

In order to answer this question we take a look at the retrieval performance on the length-based selective index (len).

Table 5.6 (top line) shows the optimal parameter settings and retrieval performance for our length-based selective index. For the generalized quantizations we get a significant improvement in retrieval performance, compared to the optimal length-prior settings for the full overlapping element index (see Table 4.7 for comparison). For the strict quantization we do, however, not get a significant improvement. It is quite remarkable that we can remove five million elements—82% of the total number of elements—from our index without reducing our retrieval performance. Better yet, we even get a significant improvement in retrieval performance when using the generalized quantizations. Note, however, that the average length of a plain English sentence is about 20 words. Hence, the elements we excluded from our index were all shorter than an average English sentence. It is thus not surprising that those elements can be safely excluded from our ranked lists since they are unlikely to give a highly exhaustive answer to fulfill the user's information need.

Note that the optimal parameter settings for the length-prior parameter do not change even if we exclude the shortest elements from our index. The baseline length prior is needed for the generalized quantizations and a value of $\beta = 1.5$ is the optimal when we use the strict quantization. These results indicate that even if we have excluded the shortest elements there is still a length gap between the index and assessments that needs to be bridged.

In our experiments we have only shown that short elements can be safely ignored in a final ranked list of elements. However, we have not addressed the question whether the short elements can be useful in intermediate stages of the retrieval. Short elements have been shown to improve the retrieval of web documents [Cutler et al., 1997, Robertson et al., 2004]. A recent study has shown

Table 5.7: *Selective indexing*: Optimal values for the smoothing parameter λ and the length-prior parameter β for our selective indexing strategies. Results are for title-only topics. Improvements are calculated relative to corresponding results on the full index (Table 4.8 on page 77).

(a) Length-based indexing (len)

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.25	1.5	.0908	3.1%	.85	1.5	.1105	1.8%*	.75	1.5	.0630	1.9%
2003	.15	1.0	.1404	9.9%	.65	1.5	.1020	9.2%	.20	1.0	.0808	9.8%
2004	.10	2.0	.1438	0.3%	.20	1.0	.1284	3.4%	.20	1.0	.0891	4.7%*
2005	.10	0.0	.0597	7.4%	.15	1.0	.0895	9.5%**	.15	1.0	.0895	9.5%**

(b) Tag-name index: Articles, bodies, sections and paragraphs (absp)

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.30	1.5	.0939	6.6%	.85	1.5	.1111	2.4%	.85	1.5	.0643	4.0%
2003	.45	1.5	.1293	1.3%	.65	1.5	.1009	8.0%	.55	1.5	.0801	8.8%
2004	.05	1.5	.1589	11%	.20	1.0	.1208	-2.7%	.20	1.0	.0912	7.2%
2005	.10	0.0	.0431	-22%	.20	1.0	.0912	12%	.20	1.0	.0912	12%

(c) Tag-name index: Sections and paragraphs (sp)

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.10	1.0	.0557	-37%	.80	1.5	.0452	-58%***	.70	1.5	.0291	-53%***
2003	.25	1.5	.0743	-42%*	.20	1.0	.0612	-34%**	.20	1.0	.0517	-30%*
2004	.25	2.0	.0687	-52%	.15	1.0	.0832	-33%***	.15	1.0	.0634	-25%***
2005	.15	1.5	.0445	-20%	.10	1.0	.0701	-14%*	.10	1.0	.0701	-14%*

(d) Tag-name index: Top-level sections (sec)

	strict				gen				sog2			
	λ	β	MAP	Impr.	λ	β	MAep	Impr.	λ	β	MAep	Impr.
2002	.10	0.5	.0423	-52%*	.45	1.0	.0295	-73%***	.45	1.0	.0180	-71%***
2003	.20	1.5	.0579	-55%**	.20	1.0	.0325	-65%***	.20	1.0	.0288	-61%***
2004	.20	2.0	.0374	-74%**	.10	0.5	.0333	-73%***	.05	0.5	.0246	-71%***
2005	.15	1.5	.0122	-78%	.05	0.5	.0225	-72%***	.05	0.5	.0225	-72%***

that the short elements can also be useful for improving the ranking of longer elements [Ramirez et al., 2006].

Vintage Let us now look at individual vintages. Table 5.7 (a) shows the performance of our length-based selective indexing strategy for individual vintage of the INEX collection. We see that excluding the short elements from our index—and retrieval runs—does not lead to reduced performance for any of the INEX

Table 5.8: *Selective indexing*: Performance for each *vintage* of the INEX collection, using the *overall* optimal parameter settings. Results are for title-only topics. Improvements are calculated relative to corresponding results on the full index (Table 4.9 on page 78).

(a) Length-based indexing (len)

	strict ($\lambda = 0.20$ $\beta = 1.5$)		gen ($\lambda = 0.20$ $\beta = 1.0$)		sog2 ($\lambda = 0.20$ $\beta = 1.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0895	2.5%	.0968	-6.0%	.0574	9.3%*
2003	.1291	2.4%	.0991	8.3%	.0808	20%*
2004	.1316	0.4%	.1284	13%***	.0891	4.7%*
2005	.0540	-0.2%	.0894	19%**	.0894	9.4%**

(b) Tag-name index: Articles, bodies, sections and paragraphs (absp)

	strict ($\lambda = 0.20$ $\beta = 1.5$)		gen ($\lambda = 0.20$ $\beta = 1.0$)		sog2 ($\lambda = 0.20$ $\beta = 1.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0928	6.3%	.0961	-6.7%	.0578	10%
2003	.1245	-1.3%	.0971	6.1%	.0795	18%*
2004	.1445	10%	.1208	6.1%	.0912	7.2%
2005	.0366	-32%	.0912	22%**	.0912	12%

(c) Tag-name index: Sections and paragraphs (sp)

	strict ($\lambda = 0.30$ $\beta = 1.5$)		gen ($\lambda = 0.15$ $\beta = 1.0$)		sog2 ($\lambda = 0.15$ $\beta = 1.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0545	-38%	.0432	-58%***	.0285	-46%***
2003	.0740	-41%*	.0611	-33%**	.0517	-23%*
2004	.0595	-55%*	.0832	-27%***	.0634	-25%***
2005	.0440	-19%	.0699	-6.8%	.0699	-14%*

(d) Tag-name index: Top-level sections (sec)

	strict ($\lambda = 0.20$ $\beta = 1.5$)		gen ($\lambda = 0.15$ $\beta = 1.0$)		sog2 ($\lambda = 0.15$ $\beta = 1.0$)	
	MAP	Impr.	MAep	Impr.	MAep	Impr.
2002	.0366	-58%*	.0284	-72%***	.0170	-68%***
2003	.0579	-54%**	.0324	-65%***	.0288	-57%**
2004	.0325	-75%**	.0330	-71%***	.0239	-72%***
2005	.0116	-79%	.0218	-71%***	.0218	-73%***

vintages. This holds for all three quantizations. For the generalized quantizations (gen and sog2) we see that the exclusion of the short elements even leads to significant improvements in performance for some vintages.

Overfitting As before, we look at whether we are running the risk of overfitting to a single vintage. Table 5.8 (a) shows the performance of using the overall optimal settings for individual vintages. We see that the overall settings carry

over reasonably well to individual vintages. The only notable exception is for the 2002 vintage using the generalized quantization. We conjecture that the reason for this is the decrease in the optimal value for the length prior parameter. But as we saw in Section 4.1.2, the elements assessed as relevant in 2002 were on average longer than for the remaining years.

We now turn our attention to the second subquestion we raised in the introduction of this chapter.

What is the effect of excluding *long* elements from our index?

We explore this question by comparing the retrieval performance of two of our tag-name-based selective indices: the index of articles, bodies, sections and paragraphs (absp); and the index of sections and paragraph (sp). The only difference between the two indices is the inclusion of whole articles and of article bodies in the first index.

Lines 2 and 3 in Table 5.6 show the optimal parameter settings and retrieval performance for the two indices. We see that the performance of the absp index is similar to the length-based index described above. For the strict quantization there is hardly any difference compared to the full element index. For the generalized quantizations there is, however, a small but significant improvement. The performance of the index of sections and paragraphs only is quite different. For all quantizations there is a big and significant decrease in performance. The performance decrease is largest for the strict quantization, but smallest for the specificity-oriented quantization.

Based on the above results, including articles and bodies in our retrieval runs seems essential for good retrieval performance. This seems quite counter-intuitive since our main motivation for XML retrieval was to retrieve elements nested below the article level. Before we discuss this finding further let us look at the performance of our two indices for different vintages of the INEX test collection.

Table 5.7 (b) and (c) shows the optimal settings and retrieval performance for individual vintages. We see that the absp index generally gives improvement—yet non significant—when compared to the full element index. The large decrease in performance for the 2005 vintage and strict quantization can be explained by the performance of topic 230, whose performance drops from 1.0 to 0.5. For the section and paragraph index (sp) there is a significant decrease in performance for each vintage when evaluated using either of the generalized quantizations. The decrease is the largest for the 2002 vintage, but the least for the 2005 vintage. Using the strict quantization, the sp index also gives considerably worse performance, but the decrease is rarely significant. The decrease is the least for the 2005 vintage.

It is interesting to note that the baseline length prior continues to be useful for the section and paragraph index. The flexible length prior is even useful if we evaluate using the strict quantization. This means that even if we exclude both

the shortest and the longest elements from our index, the length prior continues to be useful to bridge the length gap between the index and the assessments.

In sum, excluding the long elements from our index leads to a significant decrease in retrieval performance. We discuss this finding further in Section 5.4.

In our analysis in Section 5.1 we saw that top-level sections (sec) was the element type that appeared in the strict recall base of the most number of topics—i.e., sections are frequently considered as both highly exhaustive and highly specific. Sections are also interesting retrieval units in the sense that they cover almost all the textual content of the collection and do not overlap each other. Our next research question is thus about section retrieval.

What is the effect of retrieving only top-level sections?

Table 5.6 (line 4) shows the optimal parameter settings and performance of our system when we exclusively retrieve sections. The performance is quite poor. The mean average (effort) precision is 65–70% lower than when we retrieve using the full overlapping element index. From Table 5.7 we see that the decrease in performance holds for each of the vintages of the INEX test collection. We will return to the top-level section retrieval task in Section 7.2 when we evaluate our system using a metric which does not reward overlap.

5.4 Discussion

We have seen that for all vintages of the INEX test collection whole articles and article bodies seem to be essential for achieving good retrieval performance. When using the generalized metrics, this observation even holds for the 2005 vintage where there were hardly any articles or bodies assessed as both highly exhaustive and highly specific. These results are quite counter-intuitive since one of the main motivations for the INEX test collection was to look below the article level for relevant information. There are several factors that may play a role in this article-retrieval bias. First of all, there are factors related to the topics:

- The information need behind the *topics* can be very general and hence satisfied by full-articles.
- The documents in the *collection* may have too narrow scope such that they are—in their entirety—highly specific to many topics.

Second, there are factors related to "unjustified" bias in the evaluation framework, i.e., the evaluation framework may introduce a bias toward the article-level, even if the topic author/assessor did not intend for such bias:

- The *assessments* may have an "unjustified" bias toward complete articles—e.g., this may be due to assessors lack of understanding of the highly complex assessment scheme.

- The *evaluation* metrics may have "unjustified" bias toward retrieval of complete articles—e.g., the metrics may be biased toward rewarding exhaustiveness over specificity.

Third, there are factors that have to do with the system's ability to model relevance:

- The *variance* in the way assessors assess specificity may be high and thus difficult for systems to "learn" the right way estimate specificity. However, independent of how specificity is assessed, exhaustiveness always accumulates up the XML tree—i.e., all ancestors of a highly exhaustive element are also highly exhaustive. As a result, the systems may give up on trying to learn how to estimate specificity and "back-off" to learning a much easier task with lower variance—namely, to estimate exhaustiveness of whole articles.

- Due to the relatively large amount of text contained in whole articles—compared to e.g., sections—the language models for articles are based on richer statistics than the language models for smaller elements. The retrieval model may be having more problems with estimating relevance of sections than the relevance of articles. Hence, our system gives better performance when articles are among the retrieved elements retrieved.

We believe that the article-retrieval bias is a combination of the above factors. The test collection does include very general topics—e.g., topic 53 which has the description "*Information retrieval on XML repositories.*" One can easily imagine whole articles—or even PhD theses—being highly exhaustive and highly specific for that topic. The test collection also includes specific topics—e.g., topic 101 which has the description "Use of the t-test in information retrieval." One can imagine that this information need is implicitly answered by looking at the "experimental setup" sections of information retrieval articles. We have not done a thorough analysis of the overall distribution of general vs. specific topics in the INEX topic set. This is a very difficult task to perform since for most topics it is very difficult for a non-expert to judge whether an information need is likely to be answered by an entire article or a smaller portion of text. Instead, we refer to ongoing work at INEX 2006 where this issue is being addressed [Kamps and Larsen, 2006]. For further analysis on the role of assessments and evaluation framework we refer to Chapter 7 where we look at alternative ways to evaluate our retrieval. For a closer look at how data sparseness affects our retrieval performance we refer to Chapter 6 where we use an element's context to improve the relevance estimation.

5.5 Conclusions

In this chapter we have looked at the importance of the notion of "unit of retrieval" in XML retrieval. We have analyzed the characteristics of relevant elements and compared them to the statistics of the full collection. Our research question was:

> What are the characteristics of element assessments in terms of tag-names?

We have seen that sections and paragraphs are the most frequent tag-names that are assessed both highly exhaustive and highly specific. Very long elements, such as articles and bodies, are also quite frequent.

We have looked at the use of selective indexing for XML element retrieval. First, we explored the effect of excluding the very many short elements from our index.

> What is the effect of excluding *short* elements from our index?

Excluding the shortest elements from our index did not lead to a decrease in performance. On the contrary it leads to significant improvement in performance when using the generalized quantizations.

We looked at the effect of excluding the long elements (articles and bodies) from our index.

> What is the effect of excluding *long* elements from our index?

Excluding the longest elements from our index did lead to a significant decrease in performance. It turns out that whole articles and article bodies are essential for achieving good retrieval performance. We have conjectured that the reason for this somewhat surprising finding may lie in a combination of factors—including, the test collection topics; the evaluation framework; and the system's modeling of relevance. The topics issue is being addressed within the INEX initiative [Kamps and Larsen, 2006]; we will take a closer look at the evaluation issue in Chapter 7; and we explore the system's modeling issue in Chapter 6.

We also looked at indexing only the top-level sections.

> What is the effect of retrieving only top-level sections?

Indexing and retrieving only the top-level sections leads to very poor performance. We will look at the section retrieval approach again in Section 7.2 when we evaluate our approaches using a metric which does not reward overlap.

Let us now return to the main research question in this chapter:

> Are there some element types that are more important than other for achieving good retrieval performance?

In our assessment analysis we found out that sections and paragraphs are the element types which are the most often considered to be both highly exhaustive and highly specific. In our evaluation we found whole articles and article bodies to be essential to achieve good retrieval performance.

Let us now briefly look forward to see how and where we follow-up on the observations in the present chapter. In the next chapter (Chapter 6) we look at how we can take the context of element into account when estimating relevance. We look at estimating an element's relevance based both on its own text and on the text of the surrounding article. In Chapter 7 we take a closer look at the evaluation framework in order to try to better understand our evaluation results.

Chapter 6

Mixture Models

Compared to full-text documents, elements are in general small. When we base our relevance calculation on the element text alone, we are working with relatively sparse statistics. In the previous chapters we account for this data sparseness by smoothing our element model with a collection model. In this chapter we incorporate document-level evidence into our ranking model in order to provide a more localized type of smoothing. Our main research question is:

> Can we improve XML element ranking by incorporating element's context into the retrieval model?

To answer this question we introduce a *mixture model* where we rank elements based on a mixture of three language models: element model, document model, and a collection model. In Section 6.1 we describe our mixture model and its implementation. We review related work on mixture models—both simple models which combine document and element score; and more complex models that combine evidence from various levels of the hierarchy.

We divide the main research question into three sub-questions. First, we ask ourselves what the overall effectiveness is:

> Does the mixture model improve retrieval effectiveness?

In Section 6.2 we present the results of applying our mixture model to our two best performing indices, the full element index and the length-based selective index. Our main finding is that the mixture model improves retrieval performance significantly.

In our second research question we ask ourselves further about the effect of our model:

> How does the mixture model affect the type of retrieved elements?

Our initial intuition is that the mixture model would be particularly helpful for medium-size elements such as sections and paragraphs. In Section 6.3 we take

a look at the distribution of elements retrieved by our mixture model and find that it leads to further accumulation of whole articles and bodies among our top retrieved elements.

In our third research question we take a look at the mixture model's robustness:

> How robust is our mixture model w.r.t. different indices?

In Section 6.4 we look at the effect of using the mixture model to retrieve sections and paragraphs only—using our section paragraph index (sp). Our main finding is that for the task of retrieving sections and paragraphs the mixture model improves performance significantly, compared to the baseline model with optimized length-priors. This result suggests that our mixture model is a robust technique that is effective for different indexing strategies.

Finally, in Section 6.5 we conclude the mixture model experiments.

6.1 Mixture Models

Individual elements—and even individual documents—contain too short text to be used on their own to estimate relevance. Hence, in the previous chapters we have accounted for this data sparseness by smoothing the element statistics using a model of the whole collection (see Section 3.3). In this section we introduce a localized smoothing method which takes element context into account—more precisely, we look at elements in the context of their surrounding document. Combining evidence from the document level has proven useful for passage retrieval [Callan, 1994] and semi-structured retrieval [Wilkinson, 1994]. The approach has also been applied successfully to the XML element retrieval task [Sigurbjörnsson et al., 2004a, 2005, Mass and Mandelbrod, 2005].

We implement our mixture model as an extension of our baseline retrieval system, introduced in Section 3.3. In our new model we estimate the element language model as a linear interpolation of three language models: one for the element itself, one for the document that contains the element, and a third one for the collection. That is, $P(t_i|e)$ is calculated as

$$\lambda_e \cdot P_{mle}(t_i|e) + \lambda_d \cdot P_{mle}(t_i|d) + (1 - \lambda_e - \lambda_d) \cdot P_{mle}(t_i), \tag{6.1}$$

where $P_{mle}(\cdot|e)$ is a language model for element e; $P_{mle}(\cdot|d)$ is a language model for the document d in which e is contained; and $P_{mle}(\cdot)$ is a language model of the collection. The parameters λ_e and λ_d are interpolation factors (smoothing parameters). We estimate the language models, $P_{mle}(\cdot|\cdot)$ and $P_{mle}(\cdot)$, using maximum likelihood estimation. For the element model we use statistics from the element index; for the document model we use statistics from the article index; and for the collection model we use document frequencies from the article index.

Note that in our mixture model approach we do not use the same collection model as we did in our baseline system. I.e., in our mixture model we use document frequencies from the article index, but in the baseline system we used element frequencies from the overlapping element index.

As we did in the baseline retrieval model we rewrite the mixture model into a scoring function that is easier to implement. Our rewriting steps are the same as used by [Hiemstra, 2001, pages 75–76]. I.e., we use a presence weighting scheme [Robertson and Spark Jones, 1976], dividing the formula with the collection model, and using a sum of logarithm weights instead of product of weights. This rewriting steps result in the following mixture-model scoring formula.

$$s_{mm}(e,q) = \sum_{i=1}^{k} \log\left(1 + \frac{\lambda_d \cdot \mathrm{tf}(t_i,d) \cdot (\sum_t \mathrm{df}(t))}{(1-\lambda_e-\lambda_d) \cdot \mathrm{df}(t_i) \cdot |d|} + \frac{\lambda_e \cdot \mathrm{tf}(t_i,e) \cdot (\sum_t \mathrm{df}(t))}{(1-\lambda_e-\lambda_d) \cdot \mathrm{df}(t_i) \cdot |e|}\right) \quad (6.2)$$

where $|d|$ is the length of document d ($|d| = \sum_t \mathrm{tf}(t,d)$); and $|e|$ is the length of element e ($|e| = \sum_t \mathrm{tf}(t,e)$). We add a length prior to our mixture model scoring formula in the same manner as discussed in Section 4.2.

Our mixture model is fairly simple—we add only one additional local smoothing layer, namely the surrounding document. The INEX document collection—and XML documents in general—allows for more advanced smoothing layers, both between the element and document layer and beyond the document layer. Our simple mixture model is sufficient for answering the main research question of this chapter: *Can we improve XML element ranking by incorporating element's context into the retrieval model?* We leave it as future work to implement more elaborate mixture models. In the remainder of this section we give an overview of related work where more—or at least other—layers of the document hierarchy are used.

Ogilvie and Callan [2005] use a hierarchical language model for XML element retrieval. Relevance of an element is estimated using its own content together with the relevance scores of both children and parent elements. Arvola et al. [2005] look at incorporating various "amounts of context" into their element retrieval ranking. They look at parent-context, root-context, and tower-context—where the last approach uses all context levels from parent to root. All contextualization approaches significantly outperformed their baseline, but using root-context gave the overall best performance. Ramirez et al. [2005] go beyond the document layer and investigate the effectiveness of integrating journal-level evidence into the retrieval process. They show that journal-level evidence can significantly improve performance. Ramirez et al. [2006] use relevance scores of small elements (section titles and italicized terms) to adjust relevance scores of the their parent/ancestor elements. Finally, one can consider the content-and-structure queries to be descriptions of query dependent mixture of relevance scores [Lalmas and Rölleke, 2004, Sigurbjörnsson et al., 2004b].

Table 6.1: *Mixture model*: Optimal values for the mixture model parameters λ_e and λ_d. We report results for all three quantizations: strict, generalized, and specificity-oriented generalized. Improvement is calculated relative corresponding results for the element+collection model ((a) Table 4.7 on page 75; (b) Table 5.6 on page 95)

(a) Full element index

	λ_e	λ_d	β	MAP	Impr.
strict	.05	.10	1.0	.1125	13%*
gen	.05	.40	0.5	.1112	15%***
sog2	.05	.25	0.5	.0850	17%***

(b) Length based index (cutoff20)

	λ_e	λ_d	β	MAP	Impr.
strict	.05	.05	1.0	.1177	16%***
gen	.05	.20	0.5	.1196	15%***
sog2	.05	.20	0.5	.0932	16%***

6.2 Experiments

In this section we evaluate the effectiveness of our mixture model. Table 6.1 shows the optimal parameter settings for our mixture model when applied to two types of indices: the full element index (See Section 3.2), and a length based index with cutoff at 20 terms (See Section 5.2). We see that our mixture model approach gives significant improvements relative to the optimal settings presented in the previous chapters (Chapters 4 and 5). This holds true for both the full element index and the length-based selective index. The significance of the results also holds true for all three quantizations. It is interesting to note that the optimal value for the length prior parameter, β, is lower for the mixture model than for the model presented in the previous chapters. This holds for both index types and for all three quantization methods. We will return to this observation in the next section, after we have looked at the results for different vintage of the INEX text collection.

Vintage Table 6.2 shows the optimal parameter settings and performance of our mixture model for each vintage of the INEX collection. We see that the mixture model performs well for all vintages. The improvement is, however, only significant for the 2003 and 2004 topic sets. It is interesting to note that the decrease in the optimal value for the length prior parameter, β, carries over to individual vintages of the INEX test collection. I.e., the optimal value for the length prior parameter, β, is about 0.5 lower for the mixture model than for the model presented in the previous chapters. As mentioned before, we will return to this issue in the next section. First, we look at whether we are running the risk of overfitting.

Overfitting Table 6.3 shows the results of using the overall optimal parameter settings for each vintage of the INEX collection. We see that the overall settings carry well over to individual vintages. I.e., we get the similar significance results

Table 6.2: *Mixture model*: Optimal values for the mixture model parameters λ_e and λ_d for each vintage of the INEX collection. Improvement is calculated relative corresponding results for the element+collection model (Table 4.8 on page 77; and Table 5.7 on page 96)

(a) Full element index (strict)

	λ_e	λ_d	β	MAP	Impr.
2002	.05	.10	1.0	.0913	3.5%
2003	.05	.25	1.0	.1658	30%**
2004	.05	.05	1.5	.1574	9.8%*
2005	.10	.10	1.0	.0581	4.5%

(b) Full element index (gen)

	λ_e	λ_d	β	MAP	Impr.
2002	.15	.40	1.0	.1134	4.5%
2003	.05	.40	1.0	.1081	16%*
2004	.05	.25	0.5	.1492	20%***
2005	.05	.25	0.5	.0884	8.1%

(c) Full element index (sog2)

	λ_e	λ_d	β	MAP	Impr.
2002	.15	.40	1.0	.0656	6.1%
2003	.05	.40	1.0	.0874	19%*
2004	.05	.20	0.5	.1052	24%***
2005	.05	.25	0.5	.0884	8.1%

(d) Length based index (strict)

	λ_e	λ_d	β	MAP	Impr.
2002	.10	.20	1.0	.0945	4.1%
2003	.05	.25	0.5	.1752	25%**
2004	.05	.05	1.5	.1575	10%**
2005	.05	.05	0.0	.0611	2.3%*

(e) Length based index (gen)

	λ_e	λ_d	β	MAP	Impr.
2002	.20	.40	1.0	.1156	4.6%
2003	.05	.30	0.5	.1183	16%
2004	.05	.15	0.5	.1545	20%***
2005	.05	.10	0.5	.0991	11%*

(f) Length based index (sog2)

	λ_e	λ_d	β	MAP	Impr.
2002	.20	.40	1.0	.0670	6.4%
2003	.05	.30	0.5	.0965	19%**
2004	.05	.15	0.5	.1107	24%***
2005	.05	.10	0.5	.0991	11%*

if we use the overall optimal settings instead of the optimal settings for individual vintage. The usual exception of the 2005 vintage using the strict quantization. As discussed several times in the course of this thesis the 2005 vintage seems to be very unstable. A single topic can cause large shifts in average performance.

6.3 Retrieved Units

The results of using the mixture model showed that the optimal value for the length prior parameter, β, was systematically lower than for the model used in the previous chapters. At first glance, one could imagine that the result of this is that there is less bias toward long elements in our mixture model runs. This would fit well with our original intuition that the mixture model is good for promoting relatively short elements.

But does the mixture model return shorter elements? Table 6.4 shows the most frequent tag-names in the top-10 results. We see that the mixture model does indeed not return shorter elements than our model used in the previous chapters. On the contrary, the mixture model retrieves considerably more articles

Table 6.3: *Mixture models:* Performance for each vintage of the INEX collection, using the overall optimal parameter settings. Improvement is calculated relative corresponding results for the element+collection model (Table 4.9 on page 78; and Table 5.8 on page 97)

(a) Full element index (strict)

$\lambda_e = 0.05$, $\lambda_d = 0.10$, $\beta = 1.0$

	MAP	Impr.
2002	0.0913	4.6%
2003	0.1592	26%**
2004	0.1566	20%*
2005	0.0403	-26%

(b) Full element index (gen)

$\lambda_e = 0.05$, $\lambda_d = 0.40$, $\beta = 0.5$

	MAP	Impr.
2002	.1040	0.9%
2003	.1050	15%
2004	.1452	28%***
2005	.0851	14%

(c) Full element index (sog2)

$\lambda_e = 0.05$, $\lambda_d = 0.25$, $\beta = 0.5$

	MAP	Impr.
2002	.0599	14%
2003	.0803	20%**
2004	.1048	23%***
2005	.0884	8.1%

(d) Length based index (strict)

$\lambda_e = 0.05$, $\lambda_d = 0.05$, $\beta = 1.0$

	MAP	Impr.
2002	.0930	3.9%
2003	.1624	26%***
2004	.1544	17%*
2005	.0580	7.4%

(e) Length based index (gen)

$\lambda_e = 0.05$, $\lambda_d = 0.20$, $\beta = 0.5$

	MAP	Impr.
2002	.1033	6.7%
2003	.1167	18%**
2004	.1537	20%***
2005	.0975	9.1%*

(f) Length based index (sog2)

$\lambda_e = 0.05$, $\lambda_d = 0.20$, $\beta = 0.5$

	MAP	Impr.
2002	.0623	8.5%
2003	.0959	19%**
2004	.1092	23%***
2005	.0975	9.1%*

and bodies than our baseline model (See Table 4.10 for comparison). Thus, the mixture model itself seems to introduce a considerable bias toward long elements. This is contrary to our motivation of using the mixture model, where we argued that localized smoothing would help estimating the relevance of relatively short elements (such as sections and paragraphs).

In our chapter on selective indexing (Chapter 5) we observed that whole articles and bodies were essential for achieving good evaluation results. In this chapter we have witnessed further evidence of this behavior. I.e., we have observed a correlation between significantly improved retrieval performance and increased presence of articles and bodies in the top ranks of our retrieval results. The evaluation framework seems to have a bias toward the retrieval of whole articles and bodies—even for the INEX 2005 vintage where negligible a number of articles and bodies were assessed as both highly exhaustive and highly specific.

As mentioned in our discussion in Section 5.3 there can be several factors that

Table 6.4: Most frequent tag-names appearing in the top 10 results. The Avg. freq. is the number of results in the top-10 that have the corresponding tag, averaged over all topics. Standard deviation is shown in brackets.

(a) Full element index

$\lambda_e = 0.05$, $\lambda_d = 0.25$, $\beta = 0.5$			$\lambda_e = 0.05$, $\lambda_d = 0.10$, $\beta = 1.0$		
Tag	Avg. freq.		Tag-name	Avg. freq.	
article	2.71	(1.43)	article	4.05	(1.76)
bdy	1.99	(1.08)	bdy	2.52	(1.20)
sec	1.51	(1.06)	sec	1.17	(1.13)
p	0.82	(0.97)	p	0.39	(0.77)
bm	0.40	(0.56)	bm	0.38	(0.59)
atl	0.35	(1.12)	atl	0.25	(0.96)
ss1	0.34	(0.62)	ss1	0.22	(0.52)
it	0.24	(0.98)	ip1	0.14	(0.42)
ip1	0.21	(0.50)	bibl	0.12	(0.39)
bibl	0.15	(0.40)	app	0.10	(0.33)

(a) Length-based element index

$\lambda_e = 0.05$, $\lambda_d = 0.20$, $\beta = 0.5$			$\lambda_e = 0.05$, $\lambda_d = 0.05$, $\beta = 1.0$		
Tag-name	Avg. freq.		Tag-name	Avg. freq.	
article	2.86	(1.34)	article	3.88	(1.88)
bdy	2.09	(1.02)	bdy	2.41	(1.25)
sec	1.76	(1.08)	sec	1.24	(1.10)
p	1.08	(1.15)	p	0.75	(1.13)
ss1	0.42	(0.69)	bm	0.34	(0.60)
bm	0.39	(0.53)	ss1	0.29	(0.60)
ip1	0.33	(0.61)	ip1	0.26	(0.55)
bibl	0.18	(0.42)	app	0.13	(0.38)
bb	0.16	(0.67)	bibl	0.13	(0.41)
bib	0.16	(0.40)	bib	0.12	(0.40)

play a role in the apparent article/body bias in the evaluation. First, many of the topics may be general effectively asking for full articles about a subject. Second, the scope of the documents in the collection may be narrow and thus making them natural retrieval units. Third, there may be an "unjustified" article bias in the assessments due to misunderstandings of the assessment framework. Fourth, there may be an "unjustified" article bias in the evaluation metrics. As mentioned before, we will study some of these issues further—in Chapter 7—by looking at alternative ways to evaluate the systems-oriented element retrieval task.

Table 6.5: *Mixture model for sections and paragraphs*: Optimal values for the mixture model parameters λ_e and λ_d. We report results for all three quantizations: strict, generalized, and specificity-oriented generalized. Improvement is calculated relative corresponding results for the element+collection model (Table 5.6 (sp) on page 95).

	λ_e	λ_d	β	MAP	Impr.
strict	.05	.10	1.0	.0700	20%*
gen	.05	.20	0.5	.0767	16%***
sog2	.05	.20	0.5	.0644	17%***

6.4 Experiments for Sections and Paragraphs

In Section 6.2 we have seen that our mixture model significantly improves retrieval effectiveness. In Section 6.3 we have seen that the mixture model runs have a considerable bias toward the retrieval of whole articles and article bodies. In this section we study the robustness of our mixture model by investigating its effect on a retrieval task that is not affected by an article/body bias. We apply our mixture model to the task of retrieving sections and paragraphs. I.e., we use our index of sections and paragraphs (see Section 5.2) to estimate the element model (see Section 6.1).

Table 6.5 shows the results of applying the mixture model on our index of sections and paragraphs. We see that the mixture model gives significantly better performance, compared to applying our baseline model (with length priors) to the same index (see Section 5.2). It is interesting to note that as we saw in the previous section the optimal value for the length-prior parameter is lower, compared to our baseline model. This seems to indicate that there is some length bias effect in the mixture model.

Vintage Table 6.2 shows the optimal mixture model parameter settings for each vintage of the INEX test collection. The mixture model gives improvements for all vintages and all quantization methods. For the strict quantization the improvement is significant only for the 2004 vintage. For the two generalized quantizations the improvement is significant for the 2002, 2003, and 2004 vintages.

Overfitting Table 6.7 shows the results of applying the overall optimal settings to individual vintage of the collection. The performance of the overall optimal parameter settings is quite similar to the results of optimizing for each vintage separately. The main difference is that the overall settings do not result in significant improvement for the 2002 vintage.

Table 6.6: *Mixture model for sections and paragraphs*: Optimal values for the mixture model parameters λ_e and λ_d for each vintage of the INEX collection. Improvement is calculated relative corresponding results for the element+collection model (Table 5.7 (sp) on page 96)

(a) Strict quantization

	λ_e	λ_d	β	MAP	Impr.
2002	.05	.05	1.0	.0586	5.2%
2003	.05	.15	1.0	.0916	23%
2004	.05	.10	1.0	.0876	28%**
2005	.10	.05	0.0	.0453	1.8%

(b) Generalized quantization (gen)

	λ_e	λ_d	β	MAP	Impr.
2002	.20	.40	1.0	.0489	8.1%*
2003	.05	.35	0.5	.0726	19%**
2004	.05	.20	0.5	.1048	26%***
2005	.05	.15	0.5	.0754	7.5%

(c) Specificity-oriented generalized quantization (sog2)

	λ_e	λ_d	β	MAP	Impr.
2002	.20	.30	1.0	.0314	8.0%*
2003	.05	.30	0.5	.0623	21%**
2004	.05	.20	0.5	.0824	30%***
2005	.05	.15	0.5	.0754	7.5%

6.5 Conclusions

Let us conclude our mixture model experiments by looking at the research questions we laid out in the introduction to this chapter. First, we asked ourselves about the *effectiveness* of our mixture model approach:

Does the mixture model improve retrieval effectiveness?

We have seen that the mixture model significantly improves the retrieval performance. We have also seen that the optimal value for the length-prior parameter is consistently smaller than for our baseline model. There seems thus to be some length-bias effect in our meaning that a smaller value for the length prior is needed.

Next we asked ourselves how the mixture model affected the *unit of retrieval*:

How does the mixture model affect the type of retrieved elements?

For our mixture model runs based on our full element index and the length-based selective index, whole articles and bodies are more prominent among the top ranked results. This is yet another sign of an article-body bias in our evaluation.

Table 6.7: *Mixture models for sections and paragraphs:* Performance for each vintage of the INEX collection, using the overall optimal parameter settings. Improvement is calculated relative corresponding results for the element+collection model (Table 5.8 (sp) on page 97)

(a) Strict quantization (strict)

$\lambda_e = 0.05$, $\lambda_d = 0.10$, $\beta = 1.0$		
	MAP	Impr.
2002	0.0555	1.8%
2003	0.0914	24%
2004	0.0876	47%**
2005	0.0453	-0.9%

(b) Generalized quantization (gen)

$\lambda_e = 0.05$, $\lambda_d = 0.20$, $\beta = 0.5$		
	MAP	Impr.
2002	.0453	4.9%
2003	.0719	18%**
2004	.1048	26%***
2005	.0750	7.3%

(c) Specificity-oriented generalized quantization (sog2)

$\lambda_e = 0.05$, $\lambda_d = 0.20$, $\beta = 0.5$		
	MAP	Impr.
2002	.0296	3.7%
2003	.0616	19%**
2004	.0824	30%***
2005	.0750	7.3%

Finally, we asked ourselves about the *robustness* of our mixture model:

> How robust is our mixture model w.r.t. different indices?

We applied the mixture model to three types of element indices: our full element index, length-based selective index, and an index of sections and paragraphs only. We got the same results across the tree indices. I.e., retrieval effectiveness improved significantly and there seems to be a length-bias effect in the mixture model which results in a lower value needed for the optimal smoothing parameter. These observations lead us to conclude that the mixture model is a robust approach to improve element retrieval effectiveness.

As for our main research question:

> Can we improve XML element ranking by incorporating an element's context into the retrieval model?

We have seen that taking an element's context into account does indeed improve the element ranking. For the indices where full articles and article bodies were included the mixture-model has a strong bias toward retrieving those element types. We will take a closer look at this issue and more evaluation-related issues in the next chapter.

Chapter 7

Topic Classes, Overlap and Document Ranking

In the previous chapters we have implemented and evaluated an XML element retrieval system. Our evaluation has been base on the so-called "thorough" retrieval task which is a systems-oriented retrieval of—possibly overlapping—highly exhaustive and highly specific XML elements. We have performed a rigorous evaluation of the task using official INEX evaluation measures. In this section we look at alternative evaluations of the thorough task. I.e., we continue evaluating the same task as before, but using different evaluation settings or metrics. This chapter is further divided into three main sections.

Retrieval performance is known to differ from one topic to another. This prompts us to ask whether there is a class of topics for which our system performs better than for other classes:

What is the impact of different topic classes on retrieval performance?

In Section 7.1 we classify the INEX topics based on their characteristics. We look at our system's retrieval performance on these classes in order to see if our system performs better on one class rather than another. Our main finding is that the generalized quantizations give rather stable performance over different topic classes, but the strict quantization is more unstable. Furthermore, we show how these findings may be used to gain insight into the—somewhat surprising—importance of articles and bodies in the evaluation (see Section 5.4). This section is a continuation of work initiated in our INEX 2004 publication [Sigurbjörnsson et al., 2005].

An XML document is a hierarchy of elements nested within one another—the XML elements overlap.

What is the impact of overlap on our evaluation?

In Section 7.2 we take a look at how the nesting nature of XML elements affects the evaluation. We look at overlapping elements in the so-called strict recall-base—i.e., the set of elements assessed as both highly exhaustive and highly

specific,—we look at the overlapping elements in our runs, and finally we evaluate our overlapping runs using a metric which does not reward retrieval of overlapping content. We will see that there is indeed overlap in the strict recall base and in our runs. Our experiments with evaluating without rewarding overlap are inconclusive. An overall outcome of this work is that the appropriate evaluation setup depends on the end-usage of the element retrieval.

In Chapters 1 and 2 we motivated our XML retrieval effort by its usefulness to produce a ranked list of documents, augmented with *direct linking* and *structured result lists*. We ask ourselves if elements can provide a document ranking:

How can element relevance be used to rank documents?

In Section 7.3 we look at the effectiveness of using element retrieval to rank documents. Our main finding is that our element retrieval runs can serve as a reasonable—but not optimal—basis for ranking documents.

In Section 7.4 we will draw conclusions from our alternative evaluation.

7.1 Evaluation over Topic Classes

In this section we classify the INEX topics based their characteristics and look at the performance of our system over the different topic classes. It is well known that system performance differs from one topic to another [Harman, 2005]. Performance of a system is usually reported using a single number—the average performance over all topics. The average performance gives us good means to compare the quality of one retrieval approach to another. However, when averaging over many topics we loose information about performance for individual topic or topic classes. The purpose of this section is to take a closer look at individual classes of topics. The main goal of this section is to answer the following question: *What is the impact of different topic classes on retrieval performance?* This section is a continuation of work initiated in our INEX 2004 publication [Sigurbjörnsson et al., 2005], where we analyzed our best performing run and concluded that our system performed best on topics with the following characteristics: there are few relevant elements; the relevant elements contain long text; and there is considerable overlap among the relevant elements. In this section we extend this analysis. We look at the performance of a selection of our runs over different classifications of the topics. More precisely, we look at the following retrieval approaches (runs):

- *"Vanilla" baseline*: using uniform element prior and $\lambda = 0.15$.
- *Optimal smoothing*: using uniform element prior and optimal parameter settings from Table 3.3.
- *Baseline length-prior*: we use the value $\beta = 1.0$ for our length-prior parameter and use optimal parameter settings form Table 4.4.

Table 7.1: Classification of topics based on the average length of elements assessed highly exhaustive and highly specific. Class boundaries are based on bin boundaries from Table 4.1 (Bins). The *Tagnames* column list tagnames for which the average element length is within the class boundaries.

		Topic count					
Class	Bins	2002	2003	2004	2005	Total	Tagnames
I	4–6	0	0	1	6	7	p, ip1, bb
II	7–9	3	3	5	13	24	ss2
III	10–12	7	15	13	6	41	sec, ss1, bm, bib
IV	13–16	13	9	6	1	29	article, bdy

- *Flexible length-prior*: using optimal parameter settings from Table 4.7.
- *Mixture-model*: using optimal parameter settings from Table 6.1

All our experiments in this section are done using the title-only topics. We use three different classification criteria to classify the topics:

- Average length of relevant elements (Section 7.1.1)
- Total number of relevant elements (Section 7.1.2)
- Set-based overlap among relevant elements (Section 7.1.3)

We use the strict assessments for our classification. I.e., an element is considered relevant if and only if it is assessed as highly exhaustive and highly specific.

7.1.1 The Length of Relevant Elements

In this section we classify the topics based on the average length of elements assessed highly exhaustive and highly specific. We define four classes whose boundaries are defined using the bins in Table 4.1. Table 7.1 shows how we define the classes and the number of topics that fall in each class. I.e., class I consists of bins 4–6 and contains 7 topics; class II of bins 7–9 and contains 24 topics; etc. The table also lists a selection of tag-names for which the average element length is within the particular class. E.g., the average length of paragraph elements (p) is 28 terms and falls within class I; the average length of section elements (sec) is 406 terms and falls within class III; etc.

Table 7.2 shows the performance of different retrieval approaches for the different topic classes and different quantizations. For each approach we use the optimal settings for the corresponding quantization. We will now look at the different approaches and analyze if the performance is different from one topic class to another.

Table 7.2: Performance (MAP/MAep) of various retrieval approaches for different assessment-length classes. For each retrieval approach we use the overall optimal parameter settings. Improvements are calculated relative to the previous line.

(a) Strict quantization (strict)

Run	I		II		III		IV	
"Vanilla" baseline	.1518	–	.0388	–	.0366	–	.0125	–
Optimal smooth.	.1497	-1.4%	.0410	5.7%	.0510	39%***	.0425	240%***
Baseline length-pr.	.1508	0.7%	.0416	1.5%	.0695	36%***	.1381	225%**
Flexible length-pr.	.1458	-3.3%	.0230	-45%*	.0749	7.8%	.1881	36%*
Mixture model	.0892	-39%	.0270	17%	.0865	15%*	.2256	20%**

(b) Generalized quantization (gen)

Run	I		II		III		IV	
"Vanilla" baseline	.0272	–	.0439	–	.0514	–	.0239	–
Optimal smooth.	.0408	50%	.0668	52%***	.0804	56%***	.0524	119%***
Baseline length-pr.	.0473	16%	.0920	38%**	.1021	27%***	.0881	68%***
Flexible length-pr.	.0349	-26%	.0865	-6.0%	.0989	-3.1%	.1075	22%**
Mixture model	.0519	49%	.1004	16%	.1170	18%**	.1173	9.1%

(c) Specificity-oriented generalized quantization (sog2)

Run	I		II		III		IV	
"Vanilla" baseline	.0269	–	.0355	–	.0446	–	.0185	–
Optimal smooth.	.0400	49%	.0510	44%***	.0651	46%***	.0375	103%***
Baseline length-pr.	.0448	12%	.0707	39%**	.0825	27%***	.0664	77%***
Flexible length-pr.	–		–		–		–	
Mixture model	.0548	22%	.0797	13%	.0932	13%***	.0859	29%***

Optimal smoothing The optimal smoothing run refers to the case where we used an uniform element prior and looked at the effect of changing the smoothing parameter λ. Recall—from Section 3.4—that for all quantizations the optimal value for the smoothing parameter was $\lambda = 0.95$, i.e., little smoothing was performed. We see that these smoothing settings generally improve for most topic classes and quantizations. The improvement is greatest—and significant—for topics where the average length of relevant elements is the greatest (classes III and IV).

Baseline length prior If we apply the baseline length prior we get improvement in performance for all topic classes and quantizations. Again, the improvement is greatest—and significant—for topics where the average length of relevant elements is the greatest (classes III and IV).

Table 7.3: Classification of topics based on the total number of elements assessed highly exhaustive and highly specific.

Class	Assessment count	Topic count 2002	2003	2004	2005	Total
I	$x \leq 5$	2	3	5	9	19
II	$5 < x \leq 25$	4	8	8	8	28
III	$25 < x \leq 75$	11	8	6	5	30
IV	$75 < x$	6	8	6	4	24

Flexible length prior The flexible length prior does not lead to improvement over all topic classes and quantizations. Recall—from Section 4.2—that for the specificity-oriented quantization the flexible length prior did not improve the overall performance. Hence there is no results for the flexible length prior in Table 7.2 (c). The topics with the greatest average length of relevant elements (class IV) is the only class to significantly benefit from the flexible length prior.

Mixture model Applying the mixture model generally improves performance over all quantizations and topic classes. The improvement is, however, not always significant. Note that the result for class I using the strict quantization is an exception: there is a large—but non-significant—decrease in performance. However, class I is quite small and the performance is dominated by a single topic—topic 230 which has a MAP of 1.0 for all approaches except the mixture model where its performance drops to 0.6.

Classes Let us now compare the performance of our system across the different topic classes. We look at our best performing run—the mixture model run. If we use the strict quantization to optimize our performance we seem to be tuning our system to the class of topics where the average length of relevant elements is the greatest (class IV). I.e., our MAP on that class is far greater than for the remaining classes. If we use either of the other two quantizations, we get a more even performance over the different topic classes. However, the performance for class I is worse than the performance over the remaining classes when we use the generalized quantizations.

7.1.2 The Number of Relevant Elements

In this section we classify the topics based on the number of elements assessed highly exhaustive and highly specific. Table 7.3 shows our class definition. Class I contains all topics having 5 or fewer (strictly) relevant elements; class II contains all topics having between 5 and 25 relevant elements; etc.

Table 7.4 shows the performance of different retrieval approaches for our four

Table 7.4: Performance (MAP/MAep) of various retrieval approaches for different assessment-count classes. For each retrieval approach we use the overall optimal parameter settings. Improvements are calculated relative to the previous line.

(a) Strict quantization (strict)

Run	I		II		III		IV	
"Vanilla" baseline	.0785	–	.0307	–	.0240	–	.0328	–
Optimal smooth.	.0864	10%	.0491	60%*	.0404	68%**	.0471	44%**
Baseline length-pr.	.1644	90%	.0626	27%**	.0849	110%**	.0619	31%
Flexible length-pr.	.1690	2.8%	.0955	53%**	.1008	19%	.0495	-20%
Mixture model	.1660	-1.8%	.1052	10%	.1195	19%*	.0699	41%**

(b) Generalized quantization (gen)

Run	I		II		III		IV	
"Vanilla" baseline	.0462	–	.0503	–	.0344	–	.0303	–
Optimal smooth.	.0674	46%***	.0799	59%***	.0657	91%***	.0508	68%***
Baseline length-pr.	.0996	48%**	.0928	16%**	.0995	51%***	.0752	48%**
Flexible length-pr.	.0988	-0.8%	.0980	5.6%	.1087	9.2%	.0672	-11%
Mixture model	.1138	15%	.1100	12%**	.1144	5.2%	.0958	43%*

(c) Specificity-oriented generalized quantization (sog2)

Run	I		II	III	IV			
"Vanilla" baseline	.0341	–	.0447	–	.0282	–	.0275	–
Optimal smooth.	.0493	45%**	.0638	43%***	.0501	78%***	.0432	57%***
Baseline length-pr.	.0725	47%**	.0761	19%***	.0775	55%***	.0620	44%**
Flexible length-pr.	–		–		–		–	
Mixture model	.0865	19%	.0881	16%*	.0864	11%**	.0795	28%**

assessment-count classes. We will now look at each approach and analyze its effect on different classes.

Optimal smoothing For all topic classes and all quantizations, the optimal smoothing settings give considerable improvement compared to our "vanilla" baseline. The improvements are nearly always significant (class I using strict quantization is an exception).

Baseline length prior Our baseline length prior also gives consistent improvements for all topic classes and quantizations. The improvements are nearly always significant (classes I and IV using strict quantization are exceptions).

Flexible length prior For the flexible length prior, results are mixed. For topics having very few relevant elements (class I) the flexible length prior has

little effect on performance. For topics having very many relevant elements (class IV) the flexible length prior degrades performance—however, not significantly. For the two middle classes (classes II and III) the flexible length prior improves performance—however, usually not significantly.

Mixture model The mixture model gives consistent performance improvements over all the topic classes. The improvement is the largest—and always significant—for topics having very many highly exhaustive and highly specific elements (class IV). For the strict quantization the effect of the mixture model seems to be dependent on the number of relevant elements. I.e., there is a small decrease in performance for the class containing very few relevant element (class I). For the remaining classes the mixture model improves performance and the performance increase grows as the number of relevant elements grows.

Classes Let us now look at the performance of our best performing run—the mixture model run—over different assessment count classes. If we optimize our system using the strict quantization we seem to be tuning our system for the class of topics having very few relevant elements (class I). If we use either of the generalized quantizations the scoring is more even over the topic classes. However, the performance for class IV topics is somewhat lower than for the remaining classes when the generalized quantization is used.

7.1.3 Overlap among Relevant Elements

In this section we classify topics based on the overlap among their strictly relevant elements.[1] We look at the overlap in the strict recall-base—i.e., the set of elements which were assessed as highly exhaustive and highly specific. We look at the so-called set-based overlap—i.e., the number of elements in the strict recall-base that overlap with some other element in the same recall-base. We define four classes—described in Table 7.5. Class I contains topics with no overlap at all; class II contains topics where the overlap is less 50%—less than half of the elements in the strict recall base overlaps with another element in the same recall base; class III contains topics with overlap between 50% and 90%; and class IV contains topics where set-based overlap is greater than 90%.

Table 7.6 shows the performance of different retrieval approaches for different topic classes and quantizations. We will now look at the effect of different retrieval approaches for each of the topic classes.

Optimal smoothing The optimal smoothing run gives a consistent improvement over the "vanilla" baseline. For the strict quantization there is, however, a considerable—but not significant—decrease in performance which can to a large

[1] Two elements are said to overlap if one is contained within the other

Table 7.5: Classification of topics based on the set-based overlap in the strict recall-base.

Class	Assessment count	Topic count 2002	2003	2004	2005	Total
I	$x = 0\%$	3	1	0	9	13
II	$0\% < x \leq 50\%$	8	5	4	9	26
III	$50\% < x \leq 90\%$	9	12	6	5	32
IV	$90\% < x \leq 100\%$	3	9	15	3	30

Table 7.6: Performance (MAP/MAep) of various retrieval approaches for different assessment-overlap classes. For each retrieval approach we use the overall optimal parameter settings. Improvements are calculated relative to the previous line.

(a) Strict quantization (strict)

Run	I		II		III		IV	
"Vanilla" baseline	.0104	–	.0173	–	.0190	–	.0888	–
Optimal smooth.	.0068	-35%	.0283	64%	.0313	65%***	.1177	33%***
Baseline length-pr.	.0151	123%	.0481	70%	.0759	143%	.1679	43%**
Flexible length-pr.	.0231	53%	.0527	10%	.0879	16%	.1871	11%
Mixture model	.0286	23%	.0504	-4.4%	.1065	21%	.2091	12%

(b) Generalized quantization (gen)

Run	I		II		III		IV	
"Vanilla" baseline	.0142	–	.0313	–	.0527	–	.0453	–
Optimal smooth.	.0307	116%*	.0527	69%***	.0813	54%***	.0778	72%***
Baseline length-pr.	.0624	103%*	.0717	36%***	.1008	24%**	.1127	45%***
Flexible length-pr.	.0806	29%	.0725	1.1%	.1044	3.7%	.1073	-4.7%
Mixture model	.0676	-16%	.0833	15%	.1207	16%*	.1356	26%**

(c) Specificity-oriented generalized quantization (sog2)

Run	I		II		III		IV	
"Vanilla" baseline	.0139	–	.0281	–	.0444	–	.0359	–
Optimal smooth.	.0284	105%*	.0431	54%***	.0629	42%***	.0586	63%***
Baseline length-pr.	.0501	76%*	.0582	35%***	.0781	24%*	.0886	51%***
Flexible length-pr.	–	–	–	–	–	–	–	–
Mixture model	.0544	8.7%	.0664	14%*	.0906	16%**	.1093	23%***

extent be explained by a single topic—topic 227—whose performance drops from 0.1111 to 0.0417.

Baseline length prior The baseline length prior gives a consistent improvement over the optimal smoothing run. This holds for all overlap classes and all

quantizations. The improvements are always significant for the two generalized quantizations, but the improvements for the strict quantization are only significant for class IV—the class of topics where there is the most overlap among the relevant elements.

Flexible length prior The results for the flexible length prior are mixed. The greatest improvement is for the class of topics having no overlap at all. None of the improvements of the flexible length prior are, however, significant.

Mixture model For the two classes where overlap is the most (class III and IV) the mixture model gives a consistent improvement—the improvement is significant for the generalized quantizations. For the classes where overlap is the least (class I and II) the results are mixed. The results for those classes are, however, not significant.

Classes Let us now look at the performance of out best performing run—the mixture model—over the four overlap classes. Our system performs best on the class of topics where overlap is the most (class IV); we get second best performance for the class of topics where overlap is the second most (class III); etc. These results hold independently of which quantization we use to optimize our system. However, the difference is greater for the strict quantization than for the generalized quantizations.

7.1.4 Summary

Let us now summarize our evaluation of system performance over different topic classes. We classified the topic based on three different criteria: the average length of relevant elements, the total number of relevant elements, and the set-based overlap among the relevant elements. In our classification we used the strict assessments. I.e., we considered an element to be relevant if and only if it was assessed as highly exhaustive and highly specific.

When we use the *strict* quantization to optimize the parameters of our system we seem to optimize our system to perform well for topics with the following characteristics:

- The average relevant element is long (in length-class IV);
- There are few elements assessed relevant (in count-class I);
- The set based overlap among the relevant elements is high (in overlap-class IV).

For each classification there is a single class for which the performance is clearly better than for the remaining classes. When we use either of the *generalized*

quantizations to optimize the parameters we get a more even performance over different classes. We can summarize the characteristics of our well performing topics—using the generalized quantizations—as follows:

- The average relevant element is not very short (not in length-class I);
- There are not very many elements assessed relevant (not in count-class IV);
- The set based overlap among relevant elements is fairly high (in overlap-class III or IV).

For each classification there is a single class (or two) for which the performance is clearly worse than for the remaining classes. In sum, the generalized quantizations seem to be more stable than the strict quantization. This is most likely due to the fact that more assessment data-points are used in the evaluation.

In the past few chapters we have seen that there is a correlation between good retrieval performance and long elements—e.g., whole articles and article bodies—being prominent at early ranks (see the discussion in Section 5.4). In the present section we have seen that for the strict quantization these long-element retrieval strategies—i.e., strategies that have a considerable bias toward retrieval of whole articles and bodies—seem to be rewarded only for topics where the long elements are frequently assessed as highly exhaustive and highly specific. However, for the generalized quantizations these long-element retrieval strategies seem to be rewarded even if the long elements are not frequently assessed as both highly exhaustive and highly specific. In this sense, the generalized quantizations may be too lenient. That is, there is too high payoff for the "safe" strategy of concentrating on the retrieval of the highly exhaustive articles (see the discussion in Section 5.4).

In this section we have not tried to classify the topics based on their underlying information need. This sort of classification would be more meaningful than the classifications we have provided in this section. However, it is difficult to classify the existing topics based on characteristics of their underlying information needs without additional information from the original topic authors. This issue is currently being addressed in the ongoing INEX cycle (INEX 2006) [Kamps and Larsen, 2006].

7.2 Nested Structures (a.k.a. overlap)

The hierarchical nature of XML documents means that all elements—except root elements—are nested within some other element(s). Text contained within an element is thus also contained within all ancestors of that element. Hence, any element in an XML document will overlap some other element in that document—as long as the document contains more than one element. This overlap behavior

makes XML element retrieval evaluation more complicated than the evaluation of retrieval tasks where retrieval units are non-overlapping.

One of the—at least implicit—assumptions of traditional document retrieval evaluation is that the ranked lists of documents that are being evaluated is the same ranked list of documents that would be presented to the user if it was put into action as part of an operational system. This assumption sounds fairly reasonable in the case where the retrieval units are full documents. This assumption does, however, not sound as well if it is carried over to the XML element retrieval task. Presenting a list of overlapping XML element retrieval results to a user is not a reasonable thing to do [Kazai et al., 2004a]. Empirical evaluation has shown that users are irritated by receiving a ranked list of overlapping elements [Tombros et al., 2005a,b]. Hence, at a first glance, one may argue that the retrieval of overlapping elements should not be rewarded—or even that it should be punished—in XML element retrieval evaluation.

On the other hand, studies have shown that users appreciate it when a retrieval system exploits the hierarchical structure of XML documents by using XML element retrieval results to provide a structured overview of relevant documents [Kamps and Sigurbjörnsson, 2006, Larsen et al., 2006a, Sigurbjörnsson et al., 2006]. Technology savvy users even expect retrieval systems to show the relationship between hierarchically related elements [Betsi et al., 2006]. Hence, at second glance, one may argue that overlap should not be punished—or even that it should be rewarded—in XML element retrieval evaluation.

In this section we take a look at the role of overlap in our evaluation in this thesis. The goal of this section is not to settle the long-standing "overlap issue"—since active research on evaluation framework is beyond the scope of this thesis. Instead, the main goal is twofold: to determine various basic statistics on overlap, and speculate how that might affect evaluation. In this section we evaluate the same task as before. Our task is a system-oriented retrieval of—possibly overlapping—highly exhaustive and highly specific XML elements. This section is further organized as follows.

Overlap in the Recall-base In Section 7.2.1 we look at the overlap in the strict recall-base, i.e., the recall-base consisting of all elements assessed highly exhaustive and highly specific. We ask ourselves simple questions like:

- How much overlap is there in the strict recall-base?

We answer it by reporting overlap numbers for different vintages of the INEX collection. We also ask ourselves more involved questions, like:

- What sort of element types overlap?

As a partial answer to this question, we provide a typology of "different types of overlap" which occurs in the strict recall-base. In the manner, we provide the first detailed description of overlap in the INEX collection.

Table 7.7: Set-based overlap in the strict recall-base. *Left:* the fraction of elements in the strict recall-base that overlap with at least one other element from the strict recall-base. *Right:* the fraction of articles which contain overlapping elements in the strict recall-base. Mean and median are calculated over all topics having at least one element assessed strictly relevant.

	Elements			Articles		
	mean	stdev	median	mean	stdev	median
2002	0.4914	0.3166	0.5161	0.3815	0.2682	0.3333
2003	0.7358	0.2705	0.8294	0.6330	0.2869	0.6000
2004	0.8351	0.2163	0.9273	0.7187	0.2580	0.8182
2005	0.3493	0.3563	0.2956	0.3096	0.3583	0.2313
Total	0.6052	0.3491	0.7222	0.5137	0.3384	0.5000

Overlap in Runs In Section 7.2.2 we look at the amount of overlap in the runs produced by our system.

Evaluation without rewarding overlap In Section 7.2.3 we evaluate our system using metrics which do not reward overlap.

Finally, we discuss our main findings in Section 7.2.4.

7.2.1 Overlap in the Strict Recall-base

In this subsection we will look at the overlap which is present in the strict recall base. I.e., the overlap among elements that are assessed as both highly exhaustive and highly specific for a given topic.

Intuitively, one would not expect much overlap in the strict recall-base. Given a highly exhaustive and highly specific element, one would expect most of its ancestors to be equally exhaustive, but less specific. Similarly, one would expect its descendents to be equally specific, but less exhaustive. But what do the assessments say? Table 7.7 shows the fraction of elements in the strict recall base that overlap with another element in the the strict recall base (left); and the fraction of articles containing overlapping strict assessments compared to the total number of articles containing a strict assessment (right). The mean and the median are calculated over all topics having a strict assessment. We see that the overlap in the strict recall-base has changed from one year to another. In 2002 about half of the strictly relevant elements overlapped with another strictly relevant element. In 2003 and 2004, this ratio increases considerably and about tree quarters or more of the relevant elements overlap with another relevant element. In 2005 the overlap decreases again and only about a third of the relevant elements overlap with another relevant element. The fraction of articles containing overlapping strictly relevant elements follows a similar pattern from one year to another. On

Table 7.8: Frequency of the "type" of overlap in the strict recall-base. For each overlap-type we report the total frequency of such a type (#); the fraction of documents in which such a type occurs in the strict recall base (Documents); and the fraction of topics for which such a type occurs in the strict recall-base (Topics).

Overlap-type	#	Documents	Topics
long	303	20%	51%
long/medium	**2,853**	18%	49%
medium	2,556	**28%**	**74%**
medium/short	1,570	9.4%	36%
short	154	2.2%	13%
long/short	602	3.8%	23%

average over the four years, half of the articles—containing a strict assessment—contained overlapping strict assessments. The fraction is lower in 2002 and 2005, but higher in 2003 and 2004.

These high overlap numbers are quite surprising, given the expectation that the ancestors and descendents of highly exhaustive and highly specific elements would either be less specific or less exhaustive. Let us take a closer look at the overlap among the assessments, and consider the following question:

- What sort of ancestor-descendant pairs appear in the strict recall-base?

We classify the tag-names into three classes, based on the length of its content.

Long: Whole articles and bodies (e.g., `article`, `bdy`)

Medium: Sections, paragraphs, etc. (e.g., `sec`, `ss1`, `p`, `fm`, `bm`, etc.)

Short: Emphasis, titles, etc. (e.g., `atl`, `it`, `ref`, etc.)

Intuitively, we would not expect much overlap between elements of different classes. However, in some situations one could expect to see overlap within the same class. Say, in the case of a one-paragraph abstract that is highly exhaustive and highly specific, the abstract and the paragraph should both be assessed as strictly relevant.

Table 7.8 shows the type of overlap which occurs in the strict recall-base for the INEX 2002–2005 collections. In terms of absolute frequency of overlap-pairs we see that most overlap occurs among the medium-sized elements and between long and medium-sized elements. However, if we look at document frequency and topic frequency, overlap among the medium-sized elements is the most common form of overlap. As mentioned before, it is not so surprising to see overlap among elements in the same class. E.g., given a complete article which is highly exhaustive and highly specific, it is quite reasonable that its entire body is also highly exhaustive

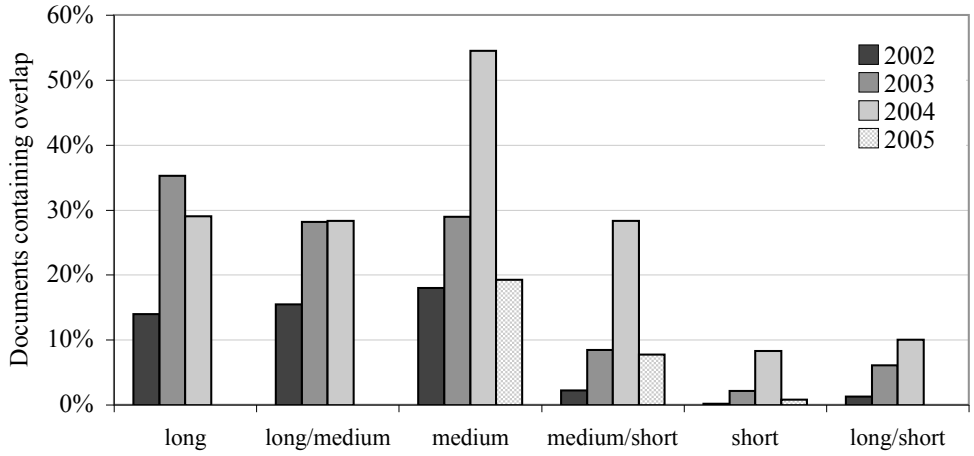


Figure 7.1: Overlap between tag-classes for each vintage of the INEX collection. The y-axis shows the fraction of documents in which such an overlap occurs.

and highly specific. It is, however, more surprising that 18% of the documents—which contain a highly exhaustive element—contain both a long element which is highly exhaustive and highly specific; and a medium-sized element which is highly exhaustive and highly specific. Before we discuss overlap in more detail, let us look at the overlap within each vintage of the INEX test collection.

Figure 7.1 shows the type of overlap that occurs in the strict recall-base for individual vintage of the INEX test collection. The figure shows the number of documents containing an overlap-pair of the particular type, as a percentage of the total number of documents containing strict assessments. In 2002 and 2003 the total overlap is distributed over three overlap classes, within long, between long and medium-sized, and within medium sized elements. In 2004 there is a big increase in overlap among medium-sized elements; and also in overlap among medium-sized and short elements. As so often before, the 2005 vintage is different from the previous vintages. As we saw in Section 5.1 the long elements disappeared almost altogether from the strict recall-base. Consequently, there is no overlap either among long elements or between long and medium-sized elements.

For readers who are interested in overlap, a more detailed analysis of the most frequently overlapping elements can be found in Appendix A.1.

7.2.2 Overlap in Runs

We have seen that there is considerable overlap in the strict recall-base of the INEX assessments. The strict recall-base is the basis of the evaluation of our runs in the previous chapters—both when using strict and generalized quantizations. It is thus interesting to see how the overlap also appears in our runs.

There are several different ways one can measure overlap in a ranked list of elements. We will look at two:

Table 7.9: Overlap in the retrieval runs presented in the previous chapters. The calculation is based on the top 20 retrieved elements for each topic.

	Optimized using strict		Optimized using sog2	
	List-based	Set-based	List-based	Set-based
"Vanilla" baseline	4.7%	34%	4.7%	34%
Optimal smoothing	11%	53%	11%	53%
Baseline length pr.	30%	69%	29%	69%
Flexible length pr.	39%	79%	–	–
Selective (len)	41%	80%	34%	77%
Selective (absp)	37%	75%	31%	71%
Selective (sp)	18%	49%	18%	51%
Selective (sec)	0%	0%	0%	0%
Mixture model	52%	85%	54%	85%
Mixture model (len)	49%	85%	59%	91%
Mixture model (sp)	24%	51%	27%	61%

- *Set-based overlap* For each element in the ranked list we check if it overlaps with an element appearing at any location within the ranked list. We report the portion of overlapping elements, as a ratio of the total number of elements.

- *List-based overlap* For each element in the ranked list we check if it overlaps with an element appearing earlier in the ranked list. We report the portion of overlapping elements, as a ratio of all elements.

Each of the overlap measures can be applied to ranked lists at various lengths. In our experiments we look at a ranked list of *20* elements. We choose this number because we are mostly interested in the top-part of the ranking. We report the mean overlap over all topics. The set-based overlap measure is the "official" overlap measure used at INEX [Kazai et al., 2005].[2] We also report the list-based overlap to show that that the absolute overlap numbers can vary, depending on how overlap is defined.

Table 7.9 shows the overlap in the runs presented in the previous chapters. We report results for runs based on the optimal settings for two quantizations, the strict and the specificity-oriented generalized (sog2). We see that our "vanilla" baseline, where $\lambda = 0.15$ and $\beta = 0.0$, has a low amount of overlap among the top-20 retrieved elements. When we improve our retrieval performance, using optimal smoothing and length priors settings, the overlap of our ranked list increases. For our most aggressive selective indexing strategies (sp and sec) the overlap decreases again—but so did our retrieval performance. This stresses further the

[2]The official overlap numbers at INEX are calculated over the whole ranked list—i.e., as much as 1500 top ranked elements.

Table 7.10: Type of overlap present among the top-20 elements of our ranked lists of elements. For each overlap type we report the number of documents containing the overlap type. We average over the 146 available CO topics. The highest overlap number for each run is boldface.

(a) Optimizing using strict

	long	lon/med	medium	med/sho	short	lon/sho
"Vanilla" baseline	0.08	0.12	**1.88**	0.66	0.19	0.02
Baseline length pr.	2.19	1.79	**2.24**	0.49	0.08	0.32
Flexible length pr.	**4.18**	2.27	1.58	0.26	0.04	0.21
Mixture model	**4.85**	2.38	1.23	0.27	0.05	0.28

(b) Optimizing using sog2

	long	lon/med	medium	med/sho	short	lon/sho
"Vanilla" baseline	0.08	0.12	**1.88**	0.66	0.19	0.02
Baseline length pr.	2.12	1.81	**2.31**	0.49	0.08	0.31
Flexible length pr.	–	–	–	–	–	–
Mixture model	**3.55**	2.64	1.72	0.39	0.04	0.43

relationship between performance and overlap that we saw in the previous section (Section 7.1). I.e., the better the performance of our runs, the more overlapping elements they contain.

Table 7.10 shows statistics of the type of overlap present among the top-20 elements of our result lists. We see that for our "vanilla" baseline, most of the overlap is among the medium-sized elements. As we apply the baseline length prior the overlap among the medium-sized elements increases slightly—but overlap among the long elements; and between long and medium-sized elements increases considerably. When we apply the mixture model, the overlap among long elements; and between long and medium-size elements increases again—but the overlap among middle-sized decreases slightly. Thus, there seems to be a correlation between good retrieval performance and overlap among the long elements; and between long and medium-sized elements. This does not necessarily mean that the good performance is a result of the overlap, but it does raise some questions about the evaluation framework. Note that for the generalized quantizations, the mixture model also gave an improvement for the 2005 vintage where the strict recall-base did not contain any overlap among the long elements; or between long and medium-sized elements. This indicates that if the evaluation framework is causing an "unwanted" bias toward long and overlapping elements, the problem is probably not only due to overlap in the strict recall base, but also due to how the metrics handles overlapping—strictly or partly—relevant elements. In the next section we will look at the effect of evaluating our runs using a metric which does not reward overlap.

Table 7.11: Optimal values for the smoothing parameter λ and the length-prior parameter β, when overlapping results are not rewarded (overlap=on).

	λ	β	MAep	Diff.	Overlap	Diff.
Full index	.15	1.5	.0525	21%**	78%	12%
absp	.15	1.5	.0605	11%	75%	5.6%
sp	.30	1.5	.0650	4.4%	49%	-3.9%
sec	.15	1.0	.0797	–	0%	–

7.2.3 Evaluation Without Rewarding Overlap

Previously in this chapter we have seen evidence that our evaluation framework prefers runs with a high degree of overlap. Now we look at evaluating our retrieval system using metrics which do not reward overlap. We use the specificity-oriented generalized (sog2) quantization and evaluate our runs using a mean average effort precision (MAep) metric where overlap is not rewarded (in the EvalJ lingo, `overlap=on` [EvalJ]). We do not change the retrieval task we are addressing, i.e., we use the same runs as in the previous chapters, but we evaluate them with a different metric. Our main question is:

- How does it effect the evaluation of our retrieval approaches if we do not reward overlapping results?

We analyze this effect by comparing the optimal parameter settings learned in the previous chapters—where overlap was rewarded—and the optimal parameter settings for metrics which do not reward overlap. We look at two types of effect: the effect on optimal parameter settings for individual retrieval approaches, and the effect on the relative ranking of retrieval approaches.

Effect on optimal parameter settings Table 7.11 shows the results of optimizing our system using the MAep metric which does not reward overlap. In the table we only report results using the specificity-oriented quantization. The table shows results for a selection of indices used in Chapter 4 and Chapter 5. From the table we can read what happens if we optimize our system using a metric which does not reward overlap, compared to optimizing our system using a metric which does reward overlap. We can make two interesting observations:

- the *overlap* in the runs generally *increases*; and
- the optimal value for the *length prior* parameter *increases*.

This is surprising since one the motivation for the ideal recall-base and the metrics which do not reward overlap was to decrease overlap in runs and discourage the retrieval of long texts [Kazai et al., 2004a, p.75]:

Table 7.12: Relation between the length-prior parameter and overlap in runs. The table shows type of overlap, list-based overlap, and set-based overlap. All runs use $\lambda = 0.15$ and we look at the top-20 elements retrieved.

β	long	lon/med	medium	med/sho	short	lon/sho	List	Set
0.0	0.08	0.12	1.88	0.66	0.19	0.02	4.7%	34%
0.5	0.36	0.44	2.31	0.71	0.14	0.12	11%	46%
1.0	2.19	1.79	2.24	0.49	0.08	0.32	30%	69%
1.5	4.31	2.12	1.47	0.25	0.05	0.21	39%	78%
2.0	5.50	1.79	0.88	0.09	0.00	0.08	40%	77%
2.5	5.84	1.34	0.49	0.03	0.00	0.01	38%	73%
3.0	5.94	1.03	0.32	0.03	0.00	0.01	36%	69%
3.5	5.97	0.82	0.18	0.01	0.00	0.01	35%	67%

> "The aim of returning XML fragments instead of whole documents is to reduce the user effort required in viewing large texts by allowing users to focus only on the parts relevant to their query"

What has gone wrong? Why does the ideal recall-base and the metrics which does not reward overlap encourage the retrieval of large and overlapping texts?

In order to try to explain this surprising result let us look at Table 7.13 which shows for each vintage the results of evaluating our runs using the metric which does not reward overlap. The metric which does not reward overlap generally suggests an increased length bias for the 2002–2004 vintages; but a decreased length bias for the 2005 vintage. Recall from Table 5.4—in Section 5.1—that full articles were quite prominent in the 2002–2004 strict assessments. Recall also that in the definition of the ideal recall-base parent elements are preferred over children—when a choice needs to be made between equally relevant parent-child elements the parent is chosen [Kazai et al., 2004a]. This means that for the 2002–2004 collection, whole articles become quite prominent in the ideal recall-base. Hence it is not so surprising that a considerable length bias is needed to get optimal performance. Note, that the increased length bias reduces overlap for the 2002 and 2003 optimal settings. This is most likely the effect of the increased number of article elements among the top-20 retrieved elements—from Table 7.12 we see that there is a peak in overlap around $\beta = 1.5$.

For the 2005 collection we get the "expected" effect of optimizing using a metric which does not reward overlap. I.e., both the length bias and the overlap decreases. This suggests that the 2005 assessments are a good fit to the assumptions that are underlying the definition of the ideal recall-base—but the assumptions seem to be less suitable for the 2002–2004 assessments.

Let us now look at the difference in performance between optimizing using, on the one hand, the metric which rewards overlap, and on the other hand, the metric with does not reward overlap. Since our final evaluation does not reward

Table 7.13: MAep without rewarding overlap. Optimal values for the smoothing parameter λ and the length-prior parameter β for different vintages of the INEX collection.

(a) Using the *full* element index

	λ	β	MAep	Diff.	Overlap	Diff.
2002	.85	3.0	.0639	44%	76%	-6.2%
2003	.65	2.5	.0556	12%	78%	-2.5%
2004	.10	1.5	.0642	6.3%	77%	12%
2005	.75	0.5	.0488	3.4%	56%	-19%

(b) Using the *absp* tag-name based element index

	λ	β	MAep	Diff.	Overlap	Diff.
2002	.65	2.5	.0677	25%	74%	-3.8%
2003	.65	1.5	.0673	5.7%	78%	0%
2004	.10	1.5	.0671	11%	74%	4.2%
2005	.10	0.0	.0768	19%	20%	-72%

(c) Using the *sp* tag-name based element index

	λ	β	MAep	Diff.	Overlap	Diff.
2002	.90	1.5	.0407	0.2%	49%	-2.0%
2003	.20	1.5	.0774	22%*	47%	-7.8%
2004	.25	1.0	.0648	1.3%	50%	-2.0%
2005	.15	0.5	.0857	5.0%	35%	-30%

(d) Using the *sec* tag-name based element index

	λ	β	MAep	Diff.	Overlap	Diff.
2002	.45	1.0	.0600	–	0%	
2003	.20	1.0	.0994	–	0%	
2004	.05	0.5	.0841	–	0%	
2005	.05	0.5	.0749	–	0%	

overlap—of course—we get better results if we use that metric as well in our optimization. Note, however, that the improvement is rarely significant.

Effect on relative performance We have seen that evaluating without rewarding overlapping results suggests different optimal parameter settings, compared to the case where we did reward overlapping results in our evaluation—yet rarely with a significant difference in retrieval performance. But what about the relative performance of different types of runs? Is it sensitive to the different evaluation methods?

Table 7.14 shows the relative performance of different indexing strategies. We report the mean average effort precision for a run using the overall optimal parameter settings for each index. We report the results using the specificity-oriented

Table 7.14: MAep of runs with (off) and without (on) rewarding overlapping results. Improvement is calculated relative to the run at successive rank.

	Overlap rewarded			Overlap not rewarded		
	MAep	Rank	Impr.	MAep	Rank	Impr.
Full index	.0729	2	33%***	.0525	4	–
absp	.0802	1	10%**	.0605	3	15%***
sp	.0548	3	135%***	.0650	2	7%
sec	.0233	4	–	.0797	1	23%***

generalized quantization (sog2), and using metrics which do and do not reward retrieval of overlapping relevant elements. We rank the different runs according to their performance for each of the metrics. We see that the ranking is very sensitive to the metric used. If overlap is rewarded, the indices containing the longest elements (full index, and absp) outperform the indices that only contain medium-size elements (sp and sec). If overlap is not rewarded, the tables turn. The indices of medium-size elements outperform the others. The performance difference between runs at associative ranks is usually significant—with the exception that the sp-run is not significantly better than the absp-run. Note, however, as we saw before in this section we must be careful about averaging over all vintages for the metric which does not reward overlap—since its effect is very different between vintages.

Let us thus look at different vintages separately. In Table 7.15 we break the relative performance results over different vintage of the INEX test collection. For the metric which does reward overlap the results are the same over all vintages. The absp-run outperforms the full-index-run—non significantly; the full-index-run significantly outperforms the sp-run; and the sp-run significantly outperforms the sec-run. For the metric which does not reward overlap the results change from one year to another. For the 2002 collection, absp-run gives the best performance and is significantly better than the next run, the full-index run. For the 2003 collection, the sec-run is the best and significantly outperforms the next run, the sp-run. There is no significant difference between the runs for the 2004 collection. For the 2005 collection, the sp-run is the best and significantly outperforms the next run, the absp run. We will expand on these results in the discussion below.

7.2.4 Discussion

Let us first summarize our main results from this section.

Overlap in the strict recall-base Even in the strict recall-base, there is a considerable amount of overlapping elements (35%–84%, depending on vintage). The most common type of overlap is among medium-sized elements. In the first

Table 7.15: MAep of runs with (off) and without (on) rewarding overlapping results, for different vintages of the INEX collection. Improvement is calculated relative to the run at successive rank.

(a) The 2002 vintage

	Overlap rewarded			Overlap not rewarded		
	MAep	Rank	Impr.	MAep	Rank	Impr.
Full index	.0618	2	112%***	.0639	2	7%
absp	.0643	1	4%	.0677	1	6%*
sp	.0291	3	62%***	.0407	4	–
sec	.0180	4	–	.0600	3	47%***

(b) The 2003 vintage

	Overlap rewarded			Overlap not rewarded		
	MAep	Rank	Impr.	MAep	Rank	Impr.
Full index	.0736	2	42%*	.0556	4	–
absp	.0801	1	9%	.0673	3	21%
sp	.0517	3	80%**	.0774	2	15%
sec	.0288	4	–	.0994	1	28%**

(c) The 2004 vintage

	Overlap rewarded			Overlap not rewarded		
	MAep	Rank	Impr.	MAep	Rank	Impr.
Full index	.0851	2	34%***	.0642	4	–
absp	.0912	1	7%	.0671	2	4%
sp	.0634	3	158%***	.0648	3	1%
sec	.0246	4	–	.0841	1	25%

(d) The 2005 vintage

	Overlap rewarded			Overlap not rewarded		
	MAep	Rank	Impr.	MAep	Rank	Impr.
Full index	.0817	2	17%*	.0488	4	–
absp	.0912	1	12%	.0768	2	3%
sp	.0701	3	212%***	.0857	1	12%*
sec	.0225	4	–	.0749	3	53%

three years overlap among large elements; and between large and medium-sized elements was prominent.

Overlap in runs We look at the overlap percentages for a number of our runs. Our well performing runs have a higher overlap than the less performing runs. In our best run—the mixture model—overlap is mainly among large elements; and between large and medium-sized elements.

Evaluation without rewarding overlap We evaluate our—overlapping—runs using metrics that do not reward overlap. We compare the optimal parameter settings of those metrics to the optimal parameter settings for the metrics that do reward overlap. The surprising result is that for the metrics that do no reward overlap, more length bias is needed to get optimal performance. This result carries over to the first three years of INEX, where whole articles were commonly judged as highly exhaustive and highly specific. The tables turn for the 2005 collection and less length bias gives optimal performance.

Overlap is a challenging issue for XML retrieval evaluation. One of the most challenging factors is that there is no "obvious" mapping from a ranked list of elements to an operational usage. We have seen that if we display overlapping elements as a ranked list of elements, overlap is a problem [Tombros et al., 2005b]. We have also seen that if we display overlapping elements in context of their surrounding article, overlap is a feature [Kamps and Sigurbjörnsson, 2006, Sigurbjörnsson et al., 2006]. What should we do when we evaluate ranked lists of elements? Should we reward overlap? Should we punish overlap? Or should we neither punish it nor reward it? The answer to these questions is most likely application specific.

When evaluating a ranked list of elements for a specific interface—or operational application—we should consider specific evaluation framework. First, the application designer must ask herself whether she is going to exploit nesting information in her application. If not, she should not retrieve overlapping elements and evaluate her element list using a metric which does not reward overlap. If the application does exploit nesting, the designer should ask herself what type of nesting her application exploits—and choose a metric which rewards the corresponding type of overlap. As an example, if you want to use XML element retrieval to highlight relevant text in a long text document, you may consider retrieval of non-overlapping elements and evaluate using a metric which does not reward overlap. However, if you want to use XML element retrieval to create structured result lists—defined in Chapter 1—you may consider a metric which rewards overlap, at least among the elements you are going to display in your result list.

On the other hand, if we want to evaluate a "general" ranked list of elements—to be used as raw material by an unspecified application—we should probably

ignore the overlap question altogether and use a metric which does not punish overlap—and, hence, implicitly rewards it.

In sum, the choice of evaluation framework for XML element retrieval depends on the intended application. For evaluating "general purpose" XML element retrieval one should choose a simple metric which makes weak assumptions about the end task. However, for a specific application of XML element retrieval one needs to be more selective about the retrieval task and choose a metric which is suitable for that particular task.

7.3 Ranking Documents

Recall from Chapter 1 that we defined a specific end goal for the element retrieval task. We wanted to use the element retrieval results to give a more focused access to relevant documents. More precisely, the end product is a document retrieval system which uses the element retrieval system to implement *direct linking* and *structured result list* (See further Chapter 1). In such a system two rankings are needed. First of all, we need to produce a ranked list of elements. Second, we need to produce a ranked list of documents. Hence we need to ask ourselves:

How can element retrieval be used to rank documents?

The task of creating a ranked list of elements has been studied thoroughly in the past chapters. In this Section we will briefly turn our attention to the task of producing a ranked list of documents.

We define three algorithms for using element retrieval results to produce a document ranking. We compare the performance of those algorithms to a baseline document retrieval run—i.e., a run created by using our language model (Section 3.3) on a document index (Section 3.2) using a baseline length prior (Section 4.2). Our goal of this section is not, per-se, to use element retrieval to create document ranking that outperforms a document retrieval system. Our main aim is to show that we can use our ranked list of elements to create a document ranking of similar quality as we would get using a document retrieval system.

We implement the following document ranking algorithms. The algorithms take a ranked list of elements as input (in this thesis we use element lists containing up to 1000 elements per topic). The algorithms return a ranked list of documents as output:

Best element Documents are assigned the score of their highest scoring element.

Sum elements The score of a document is the sum of scores of all retrieved elements from that document.

Sum top-n elements The score of a document is the sum of the scores of the top-n retrieved elements from that document.

Table 7.16: Performance of our four document ranking algorithms when applied on a selection of element retrieval runs. The best performing algorithm for each run is in boldface. We make a comparison between the best performing algorithm and a baseline document retrieval run ($\lambda = 0.35$, $\beta = 1.0$, MAP=0.4155, P@10=0.3869).

(a) Mean average precision (MAP)

	Algorithm				
	Best	Sum	Top 5	Rank	Comparison
Length prior	0.3328	0.3669	0.3900	**0.3943**	-5.1%
Mixture model	0.3692	0.3347	**0.3954**	0.3851	-4.8%
Mixt. model (sp)	0.3401	0.3026	**0.3708**	0.3496	-11%**

(b) Precision at 10 retrieved documents (P@10)

	Algorithm				
	Best	Sum	Top 5	rank	Comparison
Length prior	0.3000	0.3477	0.3598	**0.3850**	-0.5%
Mixture model	0.3486	0.3402	**0.3757**	0.3710	-2.9%
Mixt. model (sp)	0.3252	0.3009	**0.3607**	0.3449	-6.8%*

Rank-based The score of a document is

$$score(d) := \sum_{e \in d} \frac{1}{n_e} \cdot score(e), \tag{7.1}$$

where n_e is the rank of element e among elements contained in document d. I.e., a document d is assigned the full score of its highest scoring element, plus half the score of its 2nd highest scoring element, etc.

7.3.1 Experiments

Table 7.16 shows the performance of the document ranking algorithms when applied on three of our retrieval approaches, the baseline length prior, mixture model, and the mixture model when applied to the task of retrieving sections and paragraphs (sp). For each approach we take the optimal performing parameter settings when evaluated using MAep and the specificity-oriented generalized quantization (sog2).

For the baseline length-prior run the rank-based ranking algorithm performed the best. For the two mixture model approaches the best approach was to take the sum of 5 highest ranking elements for each document.

The performance of our best document ranking algorithms is always below the document retrieval baseline. For the runs using the full element index the performance difference is not significant. However, for the run which only retrieves

sections and paragraphs the performance is significantly worse than the document retrieval baseline.

Discussion

The results in this section may seem out of line with past research on the effectiveness of passage retrieval for ranking documents [Callan, 1994, Kaszkiel and Zobel, 1997]—where passage retrieval gave significant improvements over document retrieval. Note, however, the passage retrieval in those experiments was optimized for the document retrieval task, while the element retrieval in this thesis was optimized for finding highly exhaustive and highly specific elements.

Let us now recall the question we wanted to address in this section:

> How can element retrieval be used to rank documents?

In short, our element retrieval runs based on the full element inex can serve as a reasonable—but not optimal—basis for ranking documents.

7.4 Conclusions

In this chapter we have addressed three research questions. First, we looked at different topic classes:

> What is the impact of different topic classes on retrieval performance?

In Section 7.1 we have evaluated a selection of our runs over different topic classes. Our main findings were that when we optimize our system using the strict quantization we are likely to overfit our system to a single class of topics. However, if we use either of the generalized quantizations to optimize our system we get a more even performance over topic classes. We have used this findings to argue that the—somewhat counter-intuitive—importance of articles and bodies in the evaluation may—to some extent—be explained by the high payoff for the "safe" strategy of concentrating on retrieving the most exhaustive elements.

Next, we looked at the impact of hierarchical structures on the evaluation of XML element retrieval:

> What is the impact of overlap on our evaluation?

In Section 7.2 we explored how nested structured (a.k.a. overlap) affected our evaluation. We have seen that overlap does affect our evaluation, but is difficult to argue what that really means for our evaluation of XML element retrieval. The "appropriate" evaluation framework is namely dependent on the intended usage of the retrieved elements.

Finally, we looked at document ranking:

How can element relevance be used to rank documents?

In Section 7.3 we have evaluated our element retrieval approaches with respect to how good they are as a basis for several document ranking algorithms. Our main finding was that our element retrieval approaches serve as a reasonable—yet not optimal—basis for document ranking. This finding is useful for the application in the next chapter where we use element retrieval to give focused information access to full documents.

This chapter marks the end of our system-oriented evaluation of XML element retrieval. In the next chapter we look at how XML element retrieval can be put into action by building an interface for giving focused information access.

Chapter 8

Element Retrieval in Action

In the previous chapters we have evaluated XML element retrieval using an offline test collection. We have studied the question of optimizing the system for solving the task of finding highly exhaustive and highly specific XML elements. In this thesis we do not consider this systems-oriented element retrieval task as an end goal, but rather as a means to solve a user-oriented task. It is now time to put element retrieval into action and how XML element retrieval can be put to work as a part of an operational system which gives focused information access.

Recall the scenario in Chapter 1 where a user was searching for specific information in a collection of long documents. We asked ourselves how we could assist the user by giving her focused access to the relevant information within the relevant documents. In this chapter we build a user interface that uses our XML element retrieval system to give users focused access to the relevant information. Our aim is to show, by proof-of-concept, that XML element retrieval can be used for this purpose. Our main research question in this chapter is thus:

> How can XML element retrieval be put into action as part of an operational system for giving focused access to relevant documents?

XML element retrieval can potentially be used in a much wider range of applications than the one addressed in this chapter. In this thesis, we do not give a complete survey of all these possible applications, but limit ourselves to giving one example of such an application.

The straightforward implementation of "elements in action" would be to display a ranked list of elements in a similar way as ranked lists of documents are displayed by popular search engines, say, [Google] or [Yahoo!]. There are, however, certain properties of ranked lists of elements that may make things more involved:

Scattered-relevance: Results from the same documents may appear at different locations in the result list.

Reading order: The relevance ordering of elements from the same document within the ranked list may be very different from the natural reading order of the document.

Overlap: Elements at different ranks in the result list may overlap.

An interactive evaluation of such a straightforward implementation did indeed show that these properties need to be taken into consideration [Tombros et al., 2005a,b]. In particular, scattered relevance and overlap proved to cause confusion and frustration among users.

These three issues naturally translate into three sub-questions about how to *design* element retrieval interfaces:

- How do we handle scattered relevance?
- How do we present the two different orderings of elements within a single document, relevance order and reading order?
- How do we handle overlapping elements?

In short, we address these issues by clustering together element results from the same document and sort them in a reading order. We show the relative ranking of elements within a single document using a "relevance heat-map" of each document. Finally, we explicitly show how relevant elements overlap by displaying each document as a partial tree of relevant elements. These design questions are addressed in Section 8.2.

The next set of sub-questions we address is about the *portability* of our interface:

- How do we give focused access to third-party collections where we do not control how documents are displayed?
- Does our system generalize to other semi-structured document collections which are not necessarily in XML format?

In Section 8.3 we show how our interface can be applied to several different collections—both "our own" collection and third-party collections. The format of the collections ranges from high quality XML markup to loosely marked-up structure.

Our final set of sub-questions address the *evaluation* of our interface:

- How satisfied are users with our system?
- How do users interact with our system?

In Section 8.4 we present an evaluation of our interface, using two interactive studies. In Section 8.4.1 we evaluate an application of our interface to the INEX-IEEE Computer Society collection. In Section 8.4.2 we evaluate an application of our interface to the Wikipedia Encyclopedia. For each study the research questions will be addressed by analyzing, respectively, questionnaires that we asked our users to fill in, and logs of user-system interaction.

In addition to the three sections outlined above this chapter is organized as follows. In Section 8.1 we give background on information retrieval interfaces—in particular, interfaces that give sub-document-level access to relevant documents. Finally, in Section 8.5 discuss the lessons learned from our efforts of putting element retrieval into action.

8.1 Information Retrieval Interfaces

In this section we give an overview of related work on information retrieval interfaces. The overview is geared toward related work on focused interfaces, or more precisely, interfaces that use sub-document-level retrieval results. For a more comprehensive overview of different information retrieval interfaces we refer to an overview by Hearst [1999].

Today, probably the most familiar information retrieval interfaces are the simple—and yet powerful—interfaces provided by the popular web search engines such as Google. In a response to a query a ranked list of web documents is presented using few bells or whistles. Each web document is presented using a document title, query dependent summary—a text snippet.

The SuperBook [Remde et al., 1987] is an example of an early search interfaces for hypertext. It was designed to improve the way people obtain and use information from books—in particular, books used for learning and reference. The key feature in the document rendering part was a Table of Contents with hyperlinks to the appropriate locations within the book. Search results were presented by associating search-term counts to the entries of the Table of Contents. Hence, the user could see how the search terms were distributed throughout the book—giving information about which sections are likely to be of interests.

Hearst [1995] introduced a visualization paradigm, called TileBars, which takes document structure into account when visualizing retrieval results. The TileBars indicate relative document length, query term frequency and distribution of query terms w.r.t. each other and the document as a whole. The TileBars are based on automatic segmentation of documents into, so called, TextTiles [Hearst, 1997]. Each TextTile is a multi-paragraph segment which discusses one sub-topic of the overall document topic. The interface indicates the frequency of each query term within each tile and allows users either to enter each document at its beginning, or jump directly to the beginning of a desired tile.

Großjohann et al. [Großjohann et al., 2002, Fuhr et al., 2003b] introduce

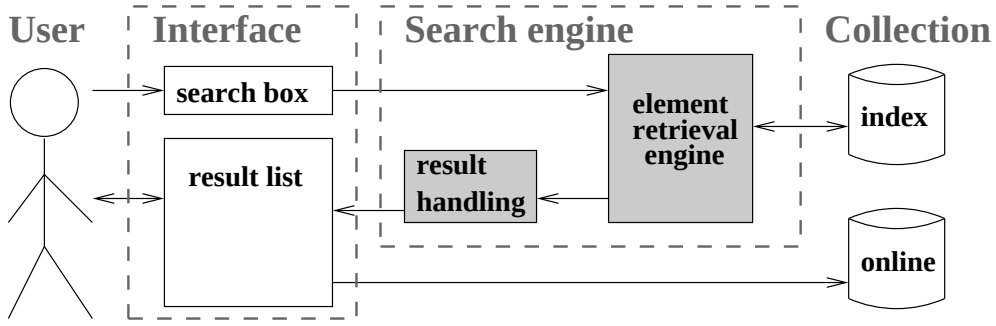


Figure 8.1: Simplified image of the interaction taking place when a user interacts with our focused retrieval system via our interface.

an interface for displaying XML element retrieval results. The interface uses so-called Partial Treemaps to present document structure and TextBars—an extension of Hearst's TileBars—to present the relative relevance of elements within a document.

Harper et al. [2004] introduced result visualization based on so-called relevance profiles. The relevance profiles are used to calculate retrieval status values (RSVs) over a tiling of the results. A single result is visualized as a bar-graph where each bar represents the RSV of the corresponding tile. The interface can be viewed as a simplification of the TileBar interface introduced by Hearst [1995]. The tiling used in the relevance profiling interface uses a linear segmentation of each document.

The interface we present in this chapter can be seen as cross-breed between the interface of popular search engines and the interface proposed by Großjohann et al. Our interface brings the tree structure into the result list, while at the same time trying to preserve the simplicity of popular search engines.

8.2 Focused Retrieval Interface

The task of an information retrieval interface is to use an information retrieval engine to give users access to documents that answer the users' information needs. Figure 8.1 shows a simplified image of of the interaction between the user and an interface. The figure extends Figure 1.4 to include the interaction between the user and the "online" version of the collection. We make a distinction between two forms of the collection. First, the online form of the collection refers to the form of the collection that can be accessed by users. In the examples we present in this thesis, the online form of the collection is an HTML view of each document. Second, the index form of the collection refers to the document and element indices described in Section 3.2.

The figure shows the actors with which the interface interacts, i.e., the user, the retrieval engine, and the collection in its online form. The main role of the

interface is to give access to the online collection using the results provided by the retrieval engine. The interface is thus dependent on the access methods (or "ingredients") made available by the two components:

Retrieval engine: provides a ranked list of elements. I.e., it returns a list of elements together with a retrieval status value for each element.

Online collection: provides a user-friendly view of the collection together with anchors in the text that can be used as possible entry points.

We have seen in Chapter 5 that we can adjust the efficiency and effectiveness of our retrieval engine by carefully selecting the granularity of the retrieval results. This means that the retrieval engine returns as results only a limited set of element-types. Granularity does also play a role on the online collection side. It can vary from one content provider to another what sort of entry-points are offered into the collection. As an example, the IEEE Computer Society provides access to their digital library[1] at the article and section level only. I.e., the only entry points offered are at the beginning of each journal article and at the beginning of each top-level section (`<sec>`). Hence, even if the retrieval engine provides results at a very fine granularity, say individual paragraphs, the interface can only give access to individual sections of the IEEE Computer Society digital library.

The interface has thus some core ingredients that can be used to present the result to the user:

- A *ranked list* of elements, provided by the retrieval system;
- *Entry-points* into the online collection, provided by the information provider;
- The *source* of each document in the collection, provided by the information provider.

These ingredients can be mixed and presented in a number of different ways. In the remainder of this section we will present and motivate how we chose to mix the ingredients to make an information retrieval interface which gives focused access to the relevant documents.

8.2.1 Design Principles

In her overview chapter on user interfaces and visualization Hearst [1999] identifies three design principles that are important for information retrieval interfaces.

Offer informative feedback Interfaces should offer *"users with feedback about the relationship between their query specification and documents retrieved,*

[1] `http://www.computer.org/portal/pages/csdl/content/index.html`

about relationships among the retrieved documents, and about about relationships between retrieved documents and metadata describing collections." (page 259). In our interface we focus on the first type of feedback, i.e., the relationship between the query specification and the *elements* retrieved. We address the second type of feedback by displaying the relationships among elements retrieved from the same document. The only feedback we give on the inter-document relations is to provide a ranking of the documents according to relevance. Feedback on collection metadata is beyond the scope of this thesis.

Reduce working memory load *"One key way information access interfaces can help with memory load is to provide mechanisms for keeping track of choices made during the search process, allowing users to return to temporarily abandoned strategies, ..." (page 259).* Reduction of memory load—other than the default functionality of back/forward buttons of standard web browsers—is beyond the scope of this thesis.

Provide alternative interface for novice and expert users *"An important tradeoff in all user interface design is that of simplicity versus power. Simple interfaces are easy to learn, at the expense of less flexibility ... Powerful interfaces allow knowledgeable users to do more and have more control over the operation of the interface, but can be time consuming to learn..." (page 259).* In our interface we opted for simplicity rather than power. We try to make use of element level relevance in a simple and intuitive manner to make an interface which is easy to use for novice users, or at least users who are familiar with the simple interface provided by common online search engines such as Google and Yahoo!.

The design of the interface was based on an evaluation of the—then state-of-the-art—XML element retrieval system HyREX[2] which was used in the INEX 2004 interactive track [Tombros et al., 2005a,b]. The interface to HyREX, used in the interactive experiment, displayed results as a ranked list of XML elements, without taking the internal document structure into account.[3] The HyREX system served also as a content provider. This means that the system could tightly integrate the result list and the collection. I.e., the interface was not limited by the access points provided by an external content provider. When the user clicked on an element result, she was brought to a window where the text of that element was shown, together with a table of contents for the whole article. Further details can be found in [Tombros et al., 2005a].

One of the major problems identified in the INEX 2004 interactive track was how the HyREX interface handled overlapping elements. I.e., the users were

[2] `http://www.is.informatik.uni-duisburg.de/projects/hyrex/`

[3] A graphical interface to HyREX did also exist, but was not used in the main interactive track.

irritated by overlapping elements appearing at different locations in the result list [Tombros et al., 2005b, p.48]:

> Overlapping components, i.e., components from the same document at different ranks in the hit list, frustrated many users but might instead be exploited to achieve a better and more comprehensible presentation of results, e.g., by hierarchical hit lists or for highlighting parts of documents. An issue to be investigated is how to present components in the hit list, so that users get a good impression of whether the component might be worth examining further or not.

One of our major goals was to develop an interface which handled more gracefully multiple relevant elements—overlapping or not—coming from the same document. We opted for displaying elements in document context. I.e., group together all element results from the same document and display as a single document result. This design decision can be motivated by the outcome of the INEX 2004 interactive track [Tombros et al., 2005b, p.47]:

> The presence of the logical structure of the documents alongside the contents of the accessed components was a feature that searchers commented positively on. The Table of Contents of each document [...] seemed to provide sufficient context to searchers in order to decide on the usefulness of the document or not.

In a sense our result list gives a structured overview of each document. Our intuition was that, this way, overlap could be a feature of the interface rather than a problem. In addition to giving a structured overview, we give a summary of the content of the relevant elements using element based snippets. This can, again, be motivated by the outcome of the INEX interactive experiment [Tombros et al., 2005b, p.47]:

> The hit list presentation in the system used in this study did not include any kind of XML document, or element, summarization; only the title and authors of the document were displayed in addition to the XPath of the component and its similarity to the query. This implied that searchers had little clues available to decide on the usefulness or not of retrieved elements.

Our interface is web-based and can in principle be used with any element retrieval system. In our experiments we use our own in-house XML retrieval extension of Lucene as its retrieval back-end. The interface accepts queries from the user and displays a ranked list of relevant documents. The interface uses a ranked list of elements to decorate documents in the result list with element relevance information.

8.2.2 Interface Functionality

The two main features of web search engines, such as Google and Yahoo! are:

- *Document links*: A link is given to the beginning of each relevant document.
- *Document snippets*: A query dependent summary of the document content.

These two features can be naturally extended to element retrieval:

- *Element links*: The user is given the option to jump directly to the relevant portions of the document via links to the beginning of each relevant element.
- *Element snippets*: query dependent text snippets are generated for individual relevant elements.

Each document in our result list contains a hierarchy of element link-snippet pairs—one pair for each relevant element. For each document its relevant elements are sorted in document order—i.e., reading order. We also show the relevance ordering of elements by introducing a "heat map:"

- *Heat map*: The relative relevance of elements within a single document is displayed with a colored heat-map where elements with high retrieval status values (RSVs) are displayed as "hotter" than elements with a lower RSV. This feature is similar to the TextBars used by Fuhr et al. [2003b] and relevance profiles used by Harper et al. [2004].

We will further explain the functionality of our interface in terms of an example. Suppose a user is interested in software licensing. In particular she wants to know more about the GNU and GPL software licenses. She uses our focused search engine to satisfy her information need. Let us follow the search process through the different stages shown in Figure 8.1. The user types the query *GNU GPL* in to the search box. The query is sent to the XML element retrieval engine which produces the ranked list of elements shown in Figure 8.2. The result handler takes the ranked list of elements as input, clusters them by document, and within each document cluster the elements are sorted in document order. The document clusters are then ordered using some of the methods introduced in Section 7.3. The result—a new ranked list of elements—is shown in Figure 8.3. The ranked list in Figure 8.3 is then visualized as shown in Figure 8.4 (on page 152). Each document result presented in the interface has the following features:

Document entry-point: At the top, the title of the document is shown. In Figure 8.4, "Open Source Software Development: An Overview" is the title of the first document. Clicking on the title gives the user access to the beginning of the document.

Document	Element	Rank	RSV
co/2001/r6033	/article[1]/bdy[1]	1	8.203884
so/1999/s1020	/article[1]/bdy[1]/sec[4]	2	8.191401
co/2001/r6033	/article[1]	3	8.128398
...			
so/1999/s1020	/article[1]/bdy[1]/sec[4]/p[5]	9	7.998796
...			
so/1999/s1020	/article[1]/bdy[1]	16	7.616654
co/2001/r6033	/article[1]/bdy[1]/sec[2]	17	7.612980
...			
so/1999/s1020	/article[1]	24	7.518584
...			
co/2001/r6033	/article[1]/bdy[1]/sec[5]	56	6.643469
...			
co/2001/r6033	/article[1]/bdy[1]/sec[4]	77	6.136657
...			
co/2001/r6033	/article[1]/bdy[1]/sec[4]/p[6]	94	5.534552
co/2001/r6033	/article[1]/bdy[1]/sec[4]/p[5]	95	5.534548

Figure 8.2: A selection of element retrieval results for the query *GNU GPL*.

Document	Element	Rank	RSV
co/2001/r6033	/article[1]	3	8.128398
co/2001/r6033	/article[1]/bdy[1]	1	8.203884
co/2001/r6033	/article[1]/bdy[1]/sec[2]	17	7.612980
co/2001/r6033	/article[1]/bdy[1]/sec[4]	77	6.136657
co/2001/r6033	/article[1]/bdy[1]/sec[4]/p[5]	95	5.534548
co/2001/r6033	/article[1]/bdy[1]/sec[4]/p[6]	94	5.534552
co/2001/r6033	/article[1]/bdy[1]/sec[5]	56	6.643469
so/1999/s1020	/article[1]	24	7.518584
so/1999/s1020	/article[1]/bdy[1]	16	7.616654
so/1999/s1020	/article[1]/bdy[1]/sec[4]	2	8.191401
so/1999/s1020	/article[1]/bdy[1]/sec[4]/p[5]	9	7.998796
...			

Figure 8.3: A selection of element retrieval results for the query *GNU GPL*, clustered by document, and ranked in document order.

Element entry-points: Below the document title is a hierarchical view of the relevant portion of the document. For the top-ranked document in Figure 8.4, seven elements are shown: the root element (article[1]), the body of the article (bdy[1]), sections 2, 4, and 5 from the body (sec[2], sec[4], sec[5]), and two paragraphs from Section 4 (p[5] and p[6]). For each element a link is provided which gives the user access to the beginning of that element. I.e., the user enters the relevant document at the beginning of the element she clicked on. As an example, Figure 8.5 (on page 153) shows the situation if the user chose "/sec[4] (LICENSING MODELS)" as her entry-point into the document.

Element snippets: For elements that do not have a relevant descendant, an element snippet is shown. For the top-ranked document in Figure 8.4, four element snippets are shown: for Section 2 (sec[2]), for the two paragraphs from Section 4 (p[5] and p[6]), and for Section 5 (sec[5]).

Heat-map: To the left of the hierarchical document view in Figure 8.4 is the heat-map—which shows the relative relevance ranking of the elements within a single document. The heat-map consists of a set of tiles—one tile for each element. The tiles are colored in a shade of red. The more relevant the element is, the darker the color of the tile.

In this section we will not go into more details of the actual system implementation. Instead, we refer to a technical report describing the implementation of the first version of the system [Bakker et al., 2005].

Let us now briefly re-cap how this interface tackles the three main outcomes of the INEX 2004 interactive track [Tombros et al., 2005b]. Our interface handles the "overlap problem" by clustering together elements from the same document and explicitly show the user how the elements overlap. We keep the "Table of Contents feature" by displaying the relevant elements as a partial tree. Finally, we solve the "lack of summary problem" by including element snippets.

8.3 Adaption to Different Collections

When applying the interface to different collections one needs to consider various aspects of the collection at hand. In this section we look at two aspects.

Entry-points How is our interface limited by the access methods available for the collection? The access methods of the collection should determine the access methods provided by the interface. The output of the retrieval engine should, in turn, be adapted to the access methods used by the interface. As an example, if individual sections are the only access points provided by the online collection then it is of little use if our system retrieves elements with

finer granularity—e.g., paragraphs—since the interface cannot give access to those elements.

Format Can we apply our interface and element retrieval engine to semi-structured collections even if they are not in XML format?

We look at three different adaptations of our interface to different collections. First, we look at the INEX-IEEE corpus where we, as search interface providers, have full control over the online document access. Second, we look at the case when we do not have control over the online access to the collection. For this we use the INEX-IEEE corpus to give access to the actual online IEEE Digital Library. We include this application to stress the dependence relation between entry-points and retrieval units. Third, we look at how out interface can be applied to a completely different semi-structured (but not XML) collection. For this purpose we use Wikipedia, the free online encyclopedia [Wikipedia].

8.3.1 The INEX-IEEE Collection

In our first adaptation of the interface we use the INEX-IEEE collection. We look at three steps that need to be taken when the interface is adapted to a new collection.

Online Collection Access Methods

We consider the case where we have full control over the access methods for the online collection. The INEX-IEEE collection is distributed in an XML format which is not an accessible reading format for humans (as opposed to computers). Thus, the collection is rendered using an XSLT style-sheet. Figure 8.5 shows a rendering of an example document. The screen is split horizontally into two parts: document meta-data (top), and the article body (below). The article body is further divided into two parts: table of contents (left) and the article text (right). The article text rendering part of the interface is designed in such a way that the screen can be positioned at the beginning of any arbitrary element. I.e., the document can be "entered" at the beginning of any arbitrary element. As an example, the screenshot in Figure 8.5 is taken after a user has followed an element-link from the result screen to the beginning of the 4th section of the article. Thus, the user is given direct access to the relevant information, but the relevant element is displayed in the context of the full article and the user can freely investigate other parts of the article, either by using the table of contents or by scrolling.

Retrieval Settings

The interface allows us to give access to any arbitrary element. In principle, we can thus use our full element index as a basis for our element retrieval. However,

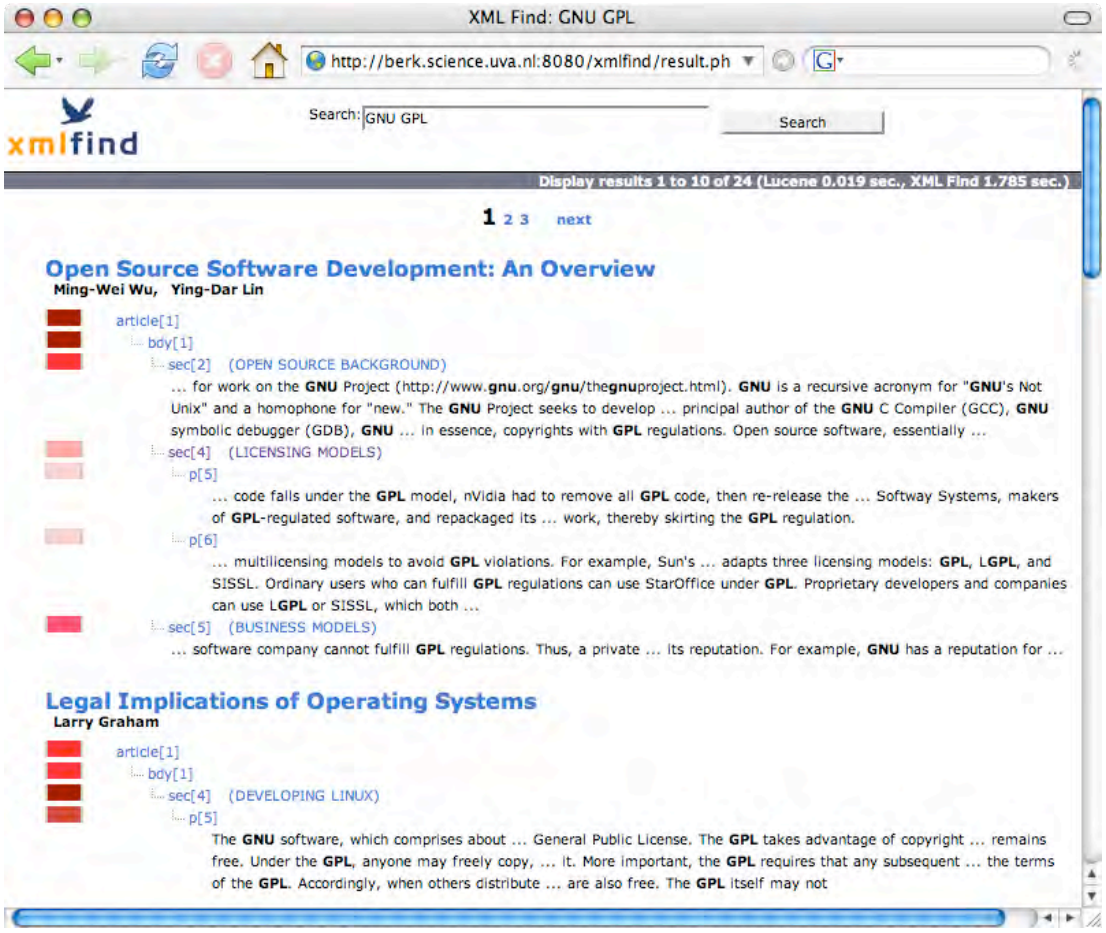


Figure 8.4: Screenshot for the INEX-IEEE search interface. The interface presents a list of relevant documents, with additional links to relevant portions within the documents.

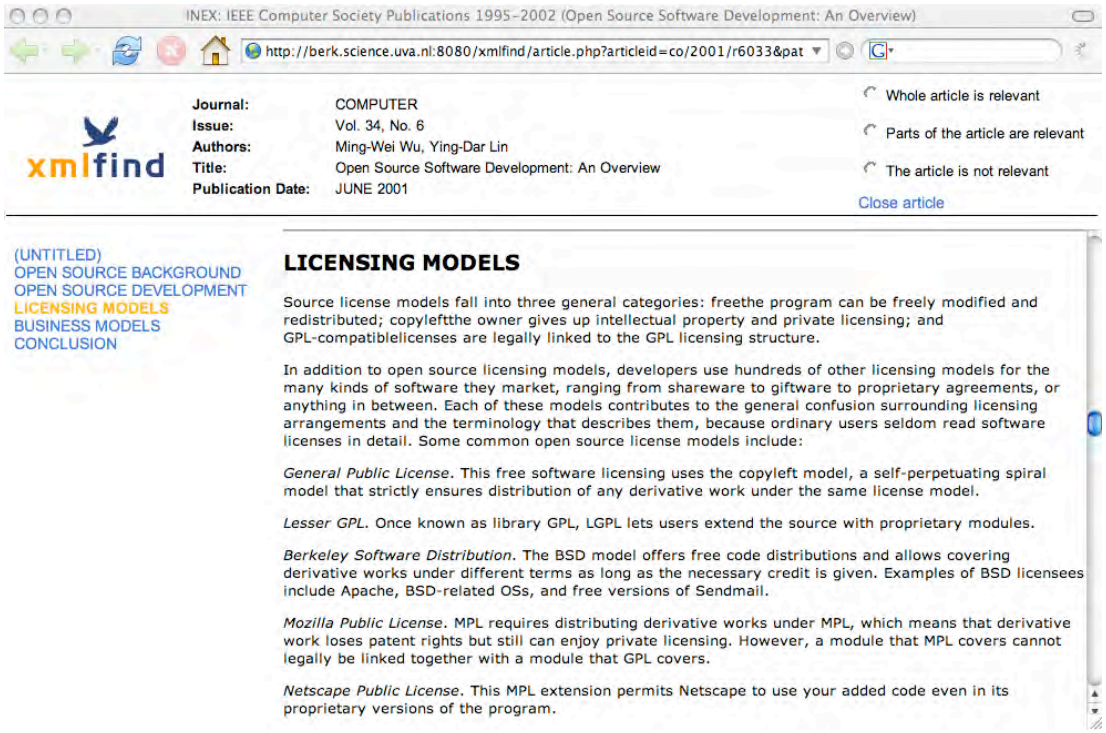


Figure 8.5: Screenshot of the document rendering part of our interface.

the use of selective indexing improves retrieval efficiency considerably, without decreasing retrieval effectiveness. Since speed is an important aspect of interactive systems we use a selective index rather than the full element index. For this particular application of the interface we use a selective index which is based on available INEX relevance assessments, i.e., we used available relevance assessments to choose which element-types to include in our index (see further information of the index in [Sigurbjörnsson and Kamps, 2006]).

Interface Functionality

In this adaption of the interface we made use of three element retrieval features: element entry-points, element snippets, and the heat-map. Figure 8.4 shows a screenshot of our INEX-IEEE application of the interface.

8.3.2 The IEEE Digital Library

Search engines are commonly used to search third party collections. This means that the access methods provided by the search engine interface depend on the access methods provided by the third party content owners. We look at this issue in the context of the IEEE Digital Library.

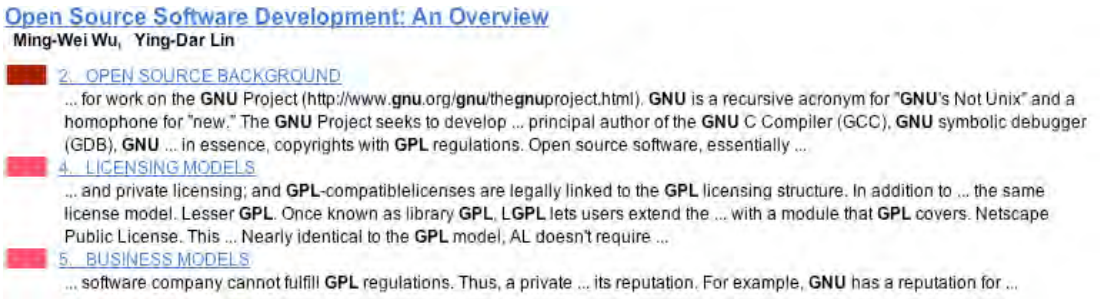


Figure 8.6: A screenshot of a single relevant document, using an interface where access is given only to beginning of articles and top-level sections.

Online Collection Access Methods

The IEEE Digital Library renders its articles in HTML format. The HTML code allows access being given to the beginning of each article and to the beginning of each (top-level) section of the article. The HTML code does not allow access to nested sections or other elements. The implication for our interface is that the element entry-points can only be at article- or section-level.

Retrieval Settings

Since we can only give access to sections we adapt our selective indexing strategy in such a way that the retrieval units match the access granularity of the online collection. Hence, sections are our units of retrieval, i.e., we only index and retrieve sections.

Interface Features

As with our previous adaption we use three element features: element entry-points, element snippets, and the heat-map. Figure 8.6 shows a screenshot of a single document result from our application of the interface to the IEEE Digital Library. The search result corresponds to the same document as the result shown in Figure 8.4.

8.3.3 Wikipedia

Our third and final adaption of the interface is to the Wikipedia encyclopedia.[4] As for the IEEE Digital Library, Wikipedia is a third party collection where the search engine does not have control over the access method available for the documents in the collection. Although the content of the Wikipedia pages is not in XML format, it is semi-structured and can easily be interpreted as a hierarchy of text objects. In particular, the wiki syntax for nested section captions can be used

[4] `http://wikipedia.org`

to identify section boundaries and nesting levels. In the adaption described in this section we use the "dumped" version of Wikipedia[5] to give access to the online version[6], but not on the INEX 2006 XMLified version of Wikipedia [Denoyer and Gallinari, 2006].

Online Collection Access Methods

Wikipedia pages are displayed in HTML format. Each page is either a single text object or a hierarchy of sections and sub-sections. The HTML code provides entry-points at the beginning of each section of the page. Access can be given to sections at any nesting level. I.e., to sections, sub-sections, sub-sub-sections, etc.

Retrieval Settings

Since the content of Wikipedia pages is not marked up in XML, we have created a simple parser for the Wikipedia syntax which allowed us to index the collection as if the pages were in XML format. Our indexing units are either (sub)-sections (if present) or complete pages (in the absence of section structure). Our index is non-overlapping, where each text token is only indexed as part of its most deeply nested ancestor.

Interface Features

Our Wikipedia interface uses two element features: element-entry points and element snippets. A screen-shot of the interface can be seen in Figure 8.7. Clicking on a link will bring the user to the on-line version of Wikipedia. Figure 8.8 shows the result of clicking the 'Demographics' link in Figure 8.7.

8.4 Interface Evaluation

We evaluate our interface in two interactive experiments. We ask a group of users to use our system to perform simulated work tasks [Borlund and Ingwersen, 1997]. In the first interactive experiment we use the INEX-IEEE interface application described in Section 8.3.1. In the second we use our Wikipedia application of the interface described in Section 8.3.3. The goal of the evaluation is to provide a *proof-of-concept* that element retrieval can be used to give focused access to semi-structured documents.

In our evaluation we explicitly address two main questions:

- How satisfied are the users with our focused interface?
- How do users interact with our focused interface?

[5] `http://download.wikimedia.org/`
[6] `http://wikipedia.org/`

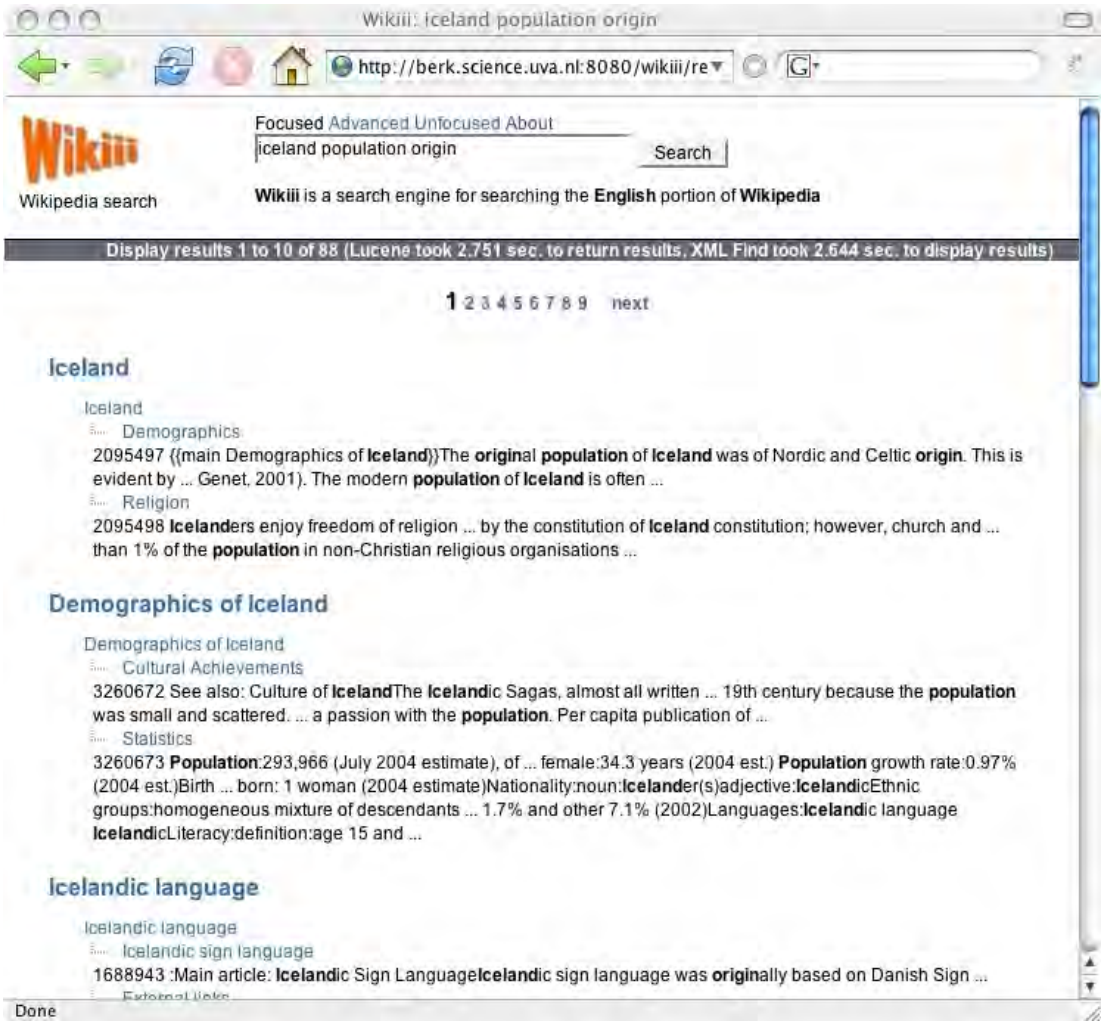


Figure 8.7: Screenshot of Wikipedia search interface

We address the first question by analyzing data collected via questionnaires where we asked the users how satisfied they were with using our system. We address the second question by analyzing logs of the interaction between the users and the system. The questions will be addressed with the aim of evaluating the interface. I.e., our aim is to evaluate how the system was of use to its users, rather than building models of information seeking behavior of users. Such modeling is beyond the scope of this thesis.

8.4.1 Case study: The INEX-IEEE Collection

In this section we discuss the results of evaluating the application of our interface to the INEX-IEEE collection (see Section 8.3.1). Our experiments were performed within the framework of the INEX interactive track [Larsen et al., 2006a].

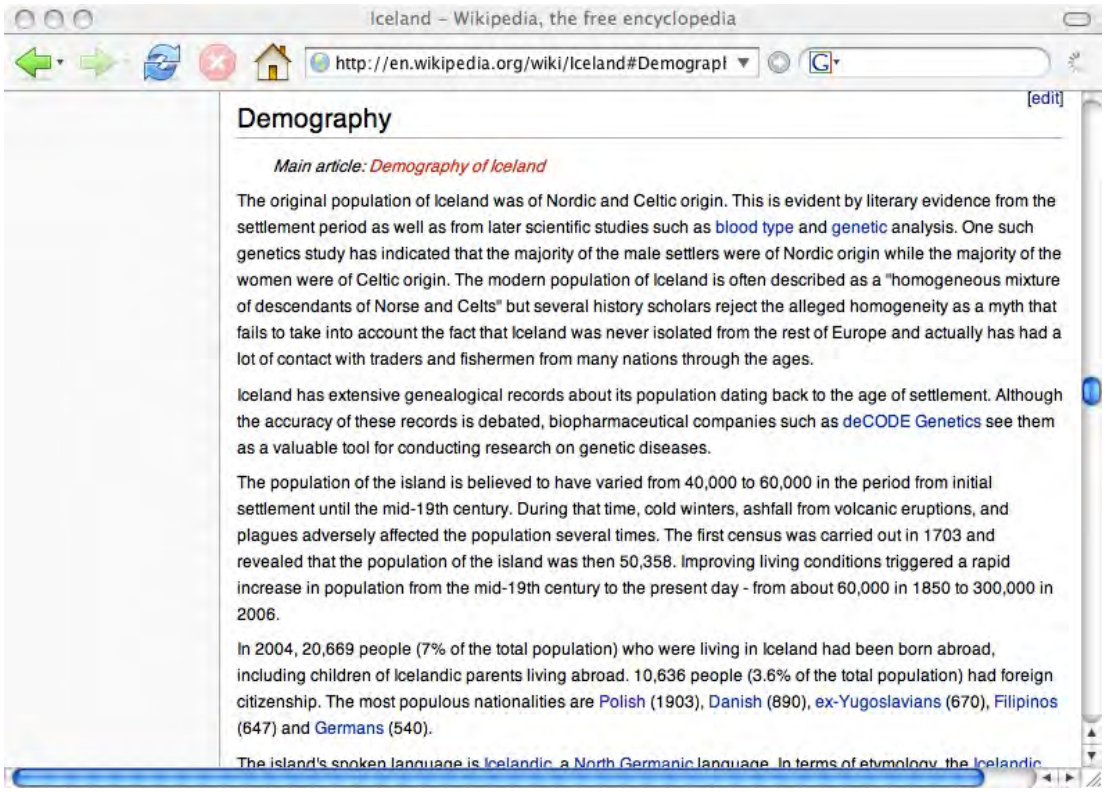


Figure 8.8: Wikipedia page which has been accessed by clicking on the 'Demographics' link in Figure 8.7

In short, the experiment involved users solving simulated work tasks using our system. We refer to Appendix B.1.1 for details on the experimental setup. The results of our interactive experiment are organized in two sets. First, we evaluate the interface by looking at how satisfied users were with the system. Second, we list some observations of how users interacted with the system.

User Satisfaction

Our post-task questionnaire included 5 questions asking the user to which extent the system was useful in solving their task:

Q3.9 How well did the system support you in this task?

Q3.10 On average, how relevant to the search task was the information presented to you?

Q3.11 Did you in general find the presentation in the result list useful?

Q3.12 Did you find the parts of the documents in the result list useful?

Q3.13 Did you find the Table of Contents in the Full Text view useful?

Table 8.1: Results of how users rated the usefulness of our system, based on answers to questions Q3.9–Q3.13.

	Q3.9	Q3.10	Q3.11	Q3.12	Q3.13
Mean	3.5	3.3	3.5	3.5	3.6
Stdev	1.4	1.2	1.0	1.2	0.8

Table 8.2: A selection of answers to question Q3.15. (In what ways (if any) did you find the system interface useful in this task?)

- Title display of the article helped. Splitting the article helps.
- You can see how the page is build. The structure is good.
- very clear; simple but effective search field.
- That I did not have to read the whole articles.
- Overview of the table of contents.
- The highlighted paragraphs can be useful, but the best thing were the search results. The top article had all the information necessary. But it can be nice when searching for long and boring articles.
- It showed the right information in a high-lighted paragraph.
- Its so simple. Just one input box and a search button. The results are listed so good, just like an XML tree model. This is what a search engine must look like in my opinion.

All questions had to be answered on a scale of 1 to 5 where 1 stood for 'Not at all', 3 for 'Somewhat' and 5 for 'Extremely.' Table 8.1 shows how users rated the usability of our system. We show average ratings over all tasks (each user performed multiple tasks).

In the post-task questionnaire we included open questions where users could express their opinion about the system:

Q3.15 In what ways (if any) did you find the system interface useful in this task?

Table 8.2 shows a selection of the answers to question Q3.15 (a full list of answers can be seen in Table B.2 on page 187). Out of 14 users, 8 mentioned the use of structure as a useful feature. It was not always clear, however, to which structural aspect they were referring. Some seem to be referring to the result list and others to the table of contents in the collection rendering interface. Three users mentioned 'quality of ranking' and two users praised the interface's simplicity.

We also asked users to list things they found not useful:

Q3.16 In what ways (if any) did you find the system interface *not* useful in this task?

Table 8.3 shows a selection of the the answers to question Q3.16 (the full list of answers can be seen in Table B.3 on page 188). The most common comment,

Table 8.3: A selection of answers to question Q3.16. (In what ways (if any) did you find the system interface *not* useful in this task?)

• Showing much not useful information.
• Parts that is not important, such as `<blabla> <p> <article> <p><blabla>`
• Relevance indication often confusing.
• I don't think its useful that it shows where in the article the words are (in which body, section, etc.).
• A bit unclear after you click on 'search': a lot of text at once, unorganized in front of you, causing confusion.
• The description of the relevant part of the article is too short.
• It could be nice if it was possible to remove the shown paragraphs in the search results so more results fit on a screen.
• It also gives the XML layout. That isn't important for someone who doesn't understand XML-code.

mentioned by 7 users, was that for some of the documents too much information was shown, making the result-list rather complex.

The main take home message from the evaluation of our user interface is that users do indeed appreciate the fact that the interface takes structure into account (Q3.11 and Q3.15). In particular, many of the users appreciated that the result list showed a structured overview of the relevant documents. However, there is a trade-off between showing a useful amount of structure and making the interface too complicated (Q3.16). While the structured overview is helpful we should be more selective about the number of elements we use for each document.

User Interaction

We will now list a number of observations we can obtain from our system-user interaction logs. We will exclusively look at how users accessed the returned documents.

Articles vs. elements How do users access the relevant documents? We start by looking at the distribution between access via article-links and access via element-links. Out of a total of 172 result visits, 95 were via article-links (55%) and 77 via element-links (45%). The popularity of article-clicks is surprising and perhaps disappointing from the perspective of XML element retrieval. However, if we take a closer look at individual search sessions we see that 14 search sessions (61%) had more visits via element-links than article-links. This means that in the majority of the search sessions element access was preferred over access via article-links. Figure 8.9 shows the distribution of article- and element-clicks for each each session (i.e., for a single user solving a single task).

Table 8.4: Distribution of *element clicks* over different element characteristics. Left: Distribution over the tag names. Middle: Distribution over depth of elements in the XML tree. Right: Distribution over "hotness" of elements.

Tag-name	Clicks	(%)
article	4	5.2%
bdy	3	3.9%
sec	27	35.1%
ss1	16	20.8%
ss2	2	2.6%
ip1	4	5.2%
p	21	27.3%

Depth level	Clicks	(%)
0	4	5.2%
1	3	3.9%
2	25	32.5%
3	35	45.5%
4	7	9.1%
5	3	3.9%

Hotness	Clicks	(%)
HOT	36	47%
Hot	5	6%
hot	17	22%
WARM	12	16%
Warm	2	3%
warm	5	6%
cold	0	0%

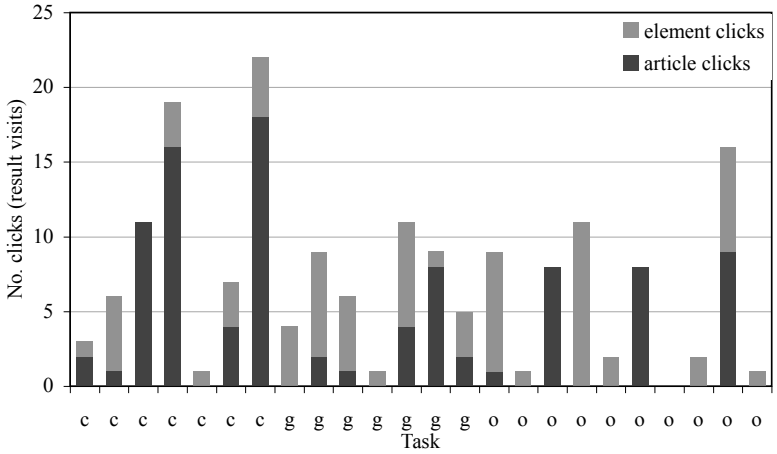


Figure 8.9: Total number of result visits for each search session. One column stands for one search session (i.e., user-and-task-type combination). Each column shows the distribution between clicks on article-links and element-links; the labels on the x-axis stand for the task type: challenging (c), general (g), own (o).

Tag-names What sort of elements are visited? First, we consider the tag name of the XML elements that users visit. Table 8.4 (left) shows the distribution of visits over different tag names. Sections are the most popular (35%), followed by paragraphs (32%) and sub-sections (21%).[7] When users decide to follow a link deep into an article, their granularity choice ranges quite widely from whole sections to individual paragraphs. In particular, users do not consider paragraphs to be too small to be worth a click. However, we do not know if users consciously choose a certain element type. We could be observing effects of other variables, such as the presence of snippets (see Figure 8.4 on page 152).

Depth Next, we focus on the depth of XML elements that users visit. Table 8.4 (middle) shows the distribution of visits over different depth levels. We see that most of the visited elements reside at depth 2–3 (78%). Both shallower (levels 0–1) and deeper levels (levels 3–5) are less commonly visited: 9% and 13% respectively. In short, users who go for elements, select reasonably deep ones.

Hotness Now, we look at the "hotness" of the elements that users visit. Recall that a heat-map was used to present the relative relevance of elements. In our interface there are 6 heat levels, based on the ratio between the retrieval status value (RSV) of an element and the highest RSV of any element in the same article. Hotness of elements is expressed in terms of different shades of red. Table 8.4 (right) shows the distribution of element clicks over different hotness-levels. Half of the clicks were on the hottest elements. The remaining clicks were on somewhat 'cooler' results. Other features than the heat-map may, however, have contributed to the users' decision to follow a link. Likely suspects are titles of sections and the snippets.

Multiple entry-points We have seen that users visit multiple results in each session. We also know that users were given multiple entry-points into each relevant document. Do the they make use of this facility? Do they visit the same article multiple times in the same search session? Despite the fact that users have element-level access to articles, we did not observe a "enter through one element", zoom out, "enter through another element" behavior—the vast majority of the articles are only visited once during a search session (82%), while 15% of the articles were visited twice. Thus, users choose a single entry-point into documents, even if they are offered more. This does not mean, however, that there should not be more than one entry-point offered.

[7]Note that the table only shows element-clicks, and 'article' in this table means a click on the element called 'article'—the root element.

Summary

In the evaluation of the interface, the test persons showed appreciation of the interface. In particular they appreciated that the document structure was used when presenting results, but often remarked that in some cases this resulted in a too complicated result. Overlapping elements did not surface as a problem in the evaluation. This indicates that when displaying elements from the same article in one cluster, overlap is not considered a problem.

From the logs of user-system interaction there are two observations that are worth special emphasis. First, the majority of users makes use of direct access to individual XML elements as their entry-points. Second, even if presented with a choice of entry-points, users predominantly access document via a single entry-point.

8.4.2 Case study: Wikipedia

In this section we will look at the results of evaluating the application of our interface to the Wikipedia Encyclopedia. In short, the experiment involved users performing simulated work tasks using our focused retrieval system (Figure 8.7). For comparison, users also performed tasks using a baseline document retrieval system which did not have element links nor element snippets (Figure 8.10). The details of the experimental setup can be found in Appendix B.2.1 and [Sigurbjörnsson et al., 2006]. As for the previous evaluation we will look at two aspects: user satisfaction and user interaction.

User Satisfaction

In the post-task questionnaires there were two questions which addressed how the user experienced using the system for solving the task. One question asked about the user's satisfaction and the other about the user's effort.

Satisfaction: How satisfied are you with the answers given by this system?

The answers were given on a scale with range 1 to 5, where 1 stood for "very dissatisfied" and 5 for "very satisfied". The results for this question can be found in Table 8.5 (a). The system satisfaction is mixed between the two tasks. Overall, there is little difference between the two systems.

Effort: The answers to the task-questions were in this system difficult/easy to find.

The answers were given on a scale with range 1 to 5, where 1 stood for "very difficult to find" and 5 stood for "very easy to find". The results for this question are reported in Table 8.5 (b). Overall, there is very little difference between the two systems.

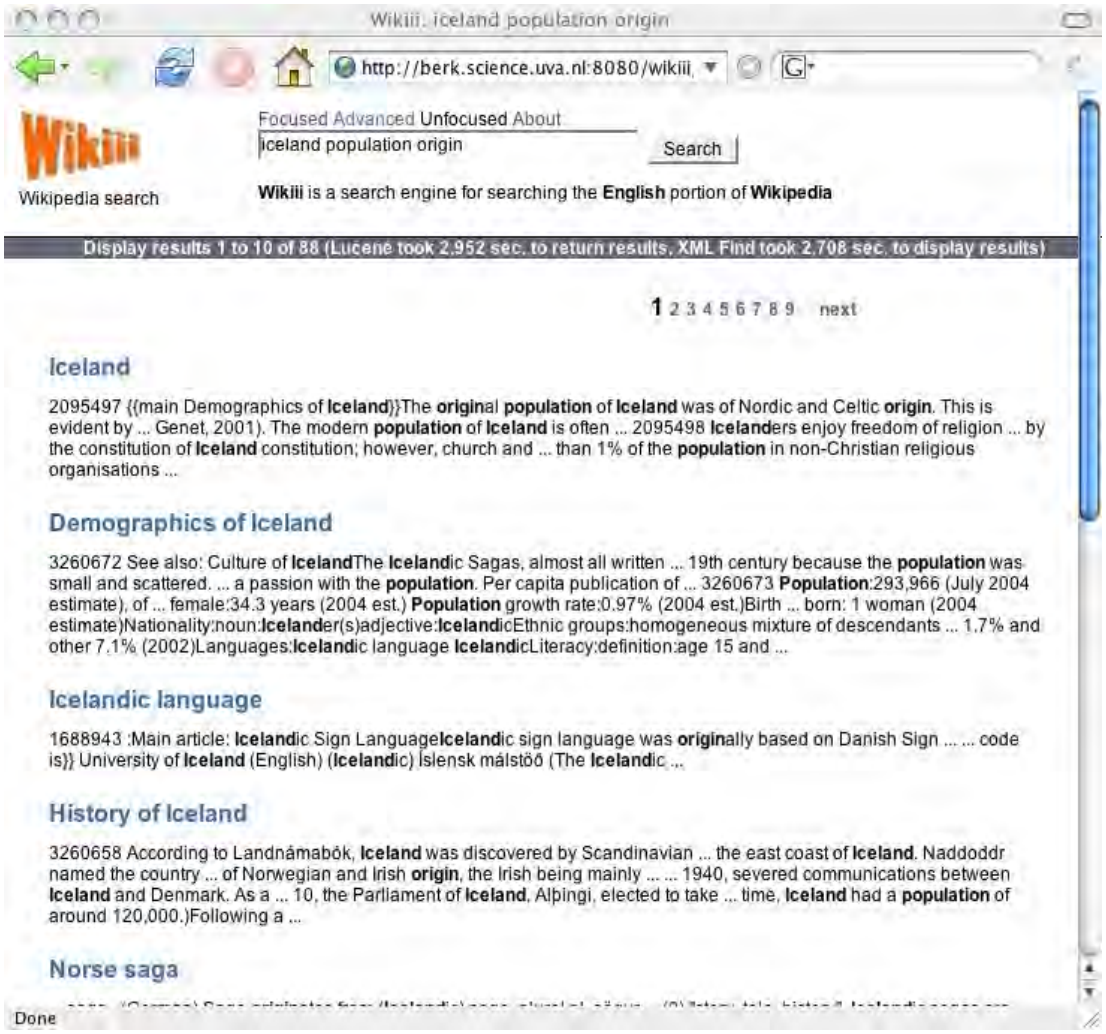


Figure 8.10: Screenshot of the baseline Wikipedia search interface

In the post-task questionnaire users were also asked how suitable they thought that the particular system was for answering respectively two types of questions, namely *specific questions* and *general questions*.

Specific: How well do you find this system suitable for specific questions?

Table 8.5 (c) shows how users rated the system's suitability for answering specific questions. The users found the focused system more suitable for specific tasks than the baseline system. Note, however, that the mean rating of the focused system is only slightly better than "neutral".

General: How well do you find this system suitable for general questions?

Table 8.5 (d) shows how suitable the users rated the system's suitability for answering general questions. Now, both systems get a rating better than "neutral".

Table 8.5: Responses on user experience. Mean rating and standard deviation (in brackets). Answers were on a 5-point scale, ranging from 1 to 5.

(a) How satisfied are you with the answers given by this system? Answers ranged from 1 ("very dissatisfied") to 5 ("very satisfied")

	Task I		Task II		Overall	
Baseline	4.17	(0.75)	3.00	(1.26)	3.58	(1.16)
Focused	3.67	(1.41)	3.67	(0.52)	3.67	(0.65)

(b) The answers to the task-questions were in this system difficult/easy to find. Answers ranged from 1 ("very difficult") to 5 ("very easy").

	Task I		Task II		Overall	
Baseline	3.17	(0.75)	2.83	(0.75)	3.00	(0.74)
Focused	2.67	(1.05)	3.50	(1.05)	3.08	(1.16)

(c) How well do you find this system suitable for specific questions? Answers ranged from 1 ("very unsuitable") to 5 ("very suitable").

	Task I		Task II		Overall	
Baseline	2.50	(0.48)	2.50	(1.05)	2.50	(0.90)
Focused	3.00	(1.03)	3.17	(0.75)	3.08	(1.08)

(d) How well do you find this system suitable for general questions? Answers ranged from 1 ("very unsuitable") to 5 ("very suitable").

	Task I		Task II		Overall	
Baseline	3.83	(0.75)	3.67	(1.03)	3.75	(0.87)
Focused	3.33	(1.21)	3.33	(1.03)	3.33	(0.98)

The baseline system is rated above the focused system.

The notions of "specific questions" and "general questions" were not linked directly to the simulated work tasks performed, and may have been interpreted differently by each of the test persons. Still, the answers given do correspond to the expectation that focused search is particularly useful for specific information needs that could be answered with a relatively short amount of text [Reid et al., 2006a].

In the post-experiment questionnaire we asked the users which of the two systems they preferred. Most users chose the focused system. In their justification they argued that using the focused system their answers were found more quickly. They also complained that while using the baseline system too much text had to be read before the right answer was found. There were, however, several users that noted that there was little difference between the two systems.

Table 8.6: Time spent per search task (minutes): mean time and standard deviation (in brackets). Each search task was divided into three distinct search assignments.

	Task I		Task II		Overall	
Baseline	31.2	(13.8)	27.0	(15.6)	29.1	(13.7)
Focused	23.3	(7.8)	22.5	(9.2)	22.9	(8.1)

Table 8.7: Page-link clicks vs. focused-link clicks in the focused interface: mean number of clicks and standard deviation (in brackets). Each search task contained three distinct search assignments.

	Task I		Task II		Overall	
Page-links	5.67	(5.85)	5.67	(4.59)	5.67	(5.02)
Focused-links	2.67	(1.03)	6.67	(4.63)	4.67	(3.82)

User Interaction

We explore the user-system interaction by mining the interaction logs provided by the systems.

Time First, we look at the time users spent solving their tasks. Table 8.6 shows the average number of minutes needed to complete each search task. We see that the users of the focused system finish their tasks quicker than the users of the baseline system. The difference is not significant, however.

Page-links vs. focused links Let us zoom in now on the interaction with the focused interface. Recall from Figure 8.7 (on page 156) that there are two types of links in the focused interface: *page-links* that bring you to the beginning of

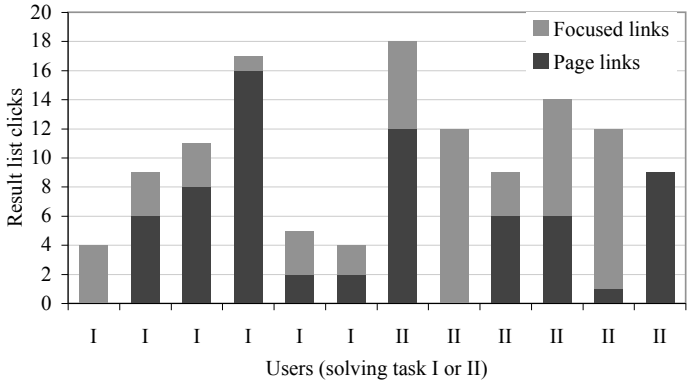


Figure 8.11: Number of result clicks per user in the focused interface. Dark: Clicks on page-links. Light: Clicks on focused-links.

Table 8.8: Analysis of focused-clicks in the focused interface. *Left*: Type of element clicked (hierarchical depth). *Right*: Section number (in the Wikipedia source) of the of the sections clicked (linear depth).

Level	Clicks	
Root	17	30%
Section	31	55%
Subsection	8	15%

Section nr.	Clicks	
Section 1	16	52%
Section 2	5	16%
Section 3	5	16%
Section 4	4	13%
Section 9	1	3%

the page, and *focused-links* that take you to the relevant sections within a page. Let us look at whether users rather click on page-links or focused-links. Table 8.7 shows the average number of page-link and focused-link clicks for each search task. Overall, there is little difference between the popularity of the two access methods. If we look at each task separately, results are mixed. Users who used the focused system in in their first task preferred page-links over focused-links. Users who used the focused system for their second task had a slight preference for focused links. Figure 8.11 shows the ratio between page-link and focused-link clicks for each user. We see that the click-behavior is very user dependent.

Depth Let us now take a closer look at the focused-links that were clicked. How deep into the documents do users dive? Table 8.8 shows both hierarchical and linear depth of user visits. The left part of the table shows where in the hierarchy the clicks are. No less than 70% of all clicks on focused links give access to sections or subsections, and the remaining 30% of the clicks are on the root element. The right part of the table shows a closer look at the section clicks. Specifically, it shows how far into the document the section clicks go. About half of the links go to the first section of the Wikipedia article, while the other half goes deeper. This may seem a bit shallow access, but the collection itself is also rather shallow. About 560,000 pages—out of c.a. million pages—are divided up into sections. Of these pages 224,000 have only one section, and 140,000 have two sections.

Summary

The majority of the users preferred the focused system over the baseline system. Their main justification was the ability to more quickly answer their information need. From the user-system interaction logs, one of our main findings is that when using the focused system users were indeed able to find their answers quicker than with the baseline system.

8.5 Conclusions

In this chapter we have seen that XML element retrieval can be put into action as a part of an operational retrieval system. Namely, XML element retrieval can be used to provide focused access to semi-structured document collections. The main difference between our system and popular commercial search engines—such as Google and Yahoo!—is the introduction of direct-linking into the documents and element level snippets.

Let us now recall our research questions from the introduction to this chapter. Our first set of questions addressed the *design choices* that need to be taken when element retrieval is put into action. We have shown that scattered relevance and overlap can be handled gracefully by presenting the element results for a single document clustered together as a partial tree. We have also shown how the relevance ordering among elements within a single document can be displayed using a heat-map.

Our second set of questions addressed the *portability* of our interface. We have identified some of the issues that need to be addressed when providing access to different document collections. We have seen that the search application is dependent on the access methods provided by the information providers. Furthermore, we have seen that methods developed for XML element retrieval can be applied to more generally to semi-structured collections even if they are not in XML format.

Our third set of questions addressed the *evaluation* of our interface. Based on our questionnaire and interaction log data there are two aspects that deserve special attention: structured overview of relevant documents and quick access to relevant information.

Structured overview of relevant documents In the evaluation of the INEX-IEEE collection the users were very positive toward the structured overview of the relevant documents provided by the element retrieval results. The evaluation revealed that there is an important trade-off between the detailed structural overview and simplicity of the interface. While the users found the structured overview useful they found it in some cases too detailed. The lesson we can draw from this is that we should keep the structured overview, but put a limit on the number of elements shown for each document.

Access to relevant information In both interactive experiments users mentioned the reduced effort and time needed to access the relevant information as one of the main advantages of our focused retrieval interface. The time aspect effect was particularly clear for the Wikipedia experiment. Note that the task of that experiment was to find very specific information. The lesson we can draw from this is that, at least in the case of a specific information need, element retrieval can provide quick access to relevant information.

Our main research question in this chapter was:

> How can XML element retrieval be put into action as part of an operational system for giving focused access to relevant documents?

Our evaluation presented in this chapter—focusing on a specific application of XML retrieval to give focused access to relevant documents—already suggests a promising future for the XML element retrieval task. In our INEX–IEEE evaluation we asked our users to comment on the more general applicability of XML element retrieval. In addition to specific questions on the interface, the tasks, and the search engine, we also gathered data on the general opinions on the usefulness of the focused XML element retrieval engines. In the post-experiment questionnaire there were two reflective questions which addressed the 'use of structure' and 'element links.'

Q4.13 Did you like the idea that the search engine takes into account the structure of the documents? Why?

Q4.14 Do you find it useful to be pointed to relevant parts of long articles?

Table 8.9 shows the responses, where each row represents the same test person. The answers to both of these questions was unanimously positive. In response to the first question 5 users mentioned the good overview of each document, 4 users mentioned that it saves time, 4 users mentioned a decrease in search effort, and 1 user mentioned better navigation. In response to the second question 3 users again mentioned the good overview of documents, 5 users mentioned that it saved time, and 5 users mentioned a decrease in search effort. Although one may expect some test persons to give a socially desirable answer, the overwhelming appreciation is striking, and the responses highlight many of the hoped advantages of an XML element retrieval system.

This chapter marks the end of the evaluation performed in this thesis. In the next and final chapter we will conclude our work.

Table 8.9: User attitude toward the 'use of structure' and 'element links'.

Q4.13. *Did you like the idea that the search engine takes into account the structure of the documents? Why?*	Q4.14. *Do you find it useful to be pointed to relevant parts of long articles? Why?*
Yes, you will have a good overview of the total article/document.	Yes, because you are able to see which articles are worth reading and which are not.
Yes, for specific information this is very useful.	Yes, gives the user an idea about the article in question.
Yes, easier to see how long the article is.	You don't need to see other parts.
Yes, its easier to see the contents of the document, better navigation.	Yes, you don't have to dig into the article yourself.
Yes, it didn't bother me.	Yes, it's more easy to find what you're looking for.
Yes, less reading time, clear overview.	Yes, saves time.
Yes, it shortens search time.	Yes, because if scan-read long articles, you easily miss some relevant parts.
Yes, saves work.	Yes, works faster.
Yes, because its much faster.	Yes, its faster.
Yes, this way of finding information takes less time.	Yes, now you don't have to read the whole article. You can get straight to the part where the information is.
Yes, its easier to see where relevant information is located.	Yes, it takes less time to find the relevant parts.
Yes, it makes it easier to find specific paragraphs.	Yes, if programmed right it can save time.
Yes, it makes it a lot easier to find what you are looking for.	Yes, it is lots more easier.
Yes, because makes me have to search less.	Yes, to search less.

Chapter 9

Conclusions

In this chapter we list the main conclusions and contributions of this thesis. Recall from Chapter 1 the scenario where a user had an information need that was answered by a relatively short part of a longer document—the user wanted to know about hiking in Northern Europe, and the search engine returned a rather long travel guide with short isolated sections about hiking. We asked ourselves if we could help the user satisfying her information need by giving her *focused* access to the relevant information, instead of merely giving her access to the relevant documents. We turned this question into the main research question of this thesis, where we restricted our attention to semi-structured documents and the XML retrieval task:

> How can we give focused information access to semi-structured documents using XML element retrieval techniques?

We further broke this question up into two sub-questions. A *system-oriented* question:

> How do we rank individual XML elements?

and a *user-oriented* question:

> How do we design an appropriate interface for providing focused information access?

In the bulk of this thesis we have addressed the system-oriented question. In Chapters 3–7 we modeled the XML retrieval task using language models, and used the modeling to implement an XML retrieval engine which we evaluated using the INEX test collection. In Chapter 8 we addressed the user-oriented question.

This chapter is further organized as follows. In Section 9.1 we discuss our conclusions and contributions regarding the *system-oriented* question. In Section 9.2 we do the same for the *user-oriented* question. Our conclusions relating to this central research question are discussed in Section 9.3. Finally, in Section 9.4 we discuss how the work in this thesis can be extended in future work.

9.1 XML Element Retrieval

In this section we will go through the XML retrieval research questions that we introduced in Chapter 1 and list our findings and conclusions. Our first question asked how XML retrieval compared to a better known task:

> How is XML element retrieval different from document retrieval?

The obvious first attempt to address this question was to simply take a document retrieval system and adapt it to the XML element retrieval task and look at the outcome. In Chapter 3 we have taken our "off-the-shelf" language model-based document retrieval engine and looked at how sensitive our retrieval performance is to changes in the most basic parameter of our retrieval model—the smoothing parameter. Our main finding was that in our baseline system the smoothing parameter serves—unexpectedly—as a means to control the length of retrieved elements. Recall that the purpose of smoothing is to account for data sparseness of—in our case—the element model. Intuitively, one would expect elements to benefit substantially from smoothing since elements contain very sparse data, i.e., we expected similar results as for document retrieval. Our results showed, however, that best element retrieval results were achieved using very little smoothing. A further analysis revealed the relation between the smoothing parameter and length of retrieved elements, i.e., applying less smoothing leads to retrieval of longer elements. These findings prompted us to take a closer look at the relation between element length and retrieval performance.

Length Normalization As a follow-up on our baseline experiments, we asked ourselves:

> What is the impact of length normalization for XML retrieval?

In Chapter 4 we have analyzed the strict recall-base of the INEX assessments, i.e., the set of elements assessed highly exhaustive and highly specific. We have seen that the average element in the strict recall-base is much longer than the average element in the collection as whole. This length-bias is variable between different vintages of the INEX test collection. Through the years the average length of the elements in the strict recall-base has decreased considerably, but it is still far greater than the collection average.

We have compared the distribution of document lengths and element lengths and seen that the distribution of document lengths is close to being a normal distribution while the distribution of element length is a very skewed distribution, where most of the elements are very short. If we restrict our analysis to relevant documents/elements we have seen that the length distribution of both relevant documents and relevant elements resembles a normal distribution. In

sum, relevant documents have a similar length distribution as the whole document collection. However, relevant elements have a radically different distribution when compared to the whole element collection.

We have experimented with the effect of using so-called *length priors* for bridging the length gap between the element collection as a whole and the relevant elements. We have seen that our baseline length prior gives significant improvement for all vintages and query formats. A flexible implementation of the length priors gives significant improvement only for the most verbose query format—a combination of titles and descriptions. An overall conclusion is that length normalization is important for XML element retrieval. Compared to document retrieval—where length normalization is also important—the significance of length normalization is greater for XML element retrieval.

We have also seen in Chapter 5 that length priors continue to be useful even if we decrease the "skewed-ness" of the element collection by removing the shortest elements from the index (len); by removing the shortest and the longest elements—section and paragraph index (sp); or by indexing and retrieving only top-level sections (sec).

The Unit of Retrieval In our next research question we addressed more explicitly the notion of *unit of retrieval*.

> Are all element types equally important retrieval units?

In Chapter 5 we have analyzed the *tag-names* of elements assessed highly exhaustive and highly specific—a.k.a. the strict recall-base. Our main finding is that sections and paragraphs appear consistently in the strict recall base. Their absolute frequency in the strict recall-base is high and they appear in the recall-base of most of the topics. Whole articles and article bodies are also quite frequent in the strict recall-base for the 2002–2004 vintages. In 2005 those element types disappear almost completely from the strict recall-base.

In Chapter 5 we have also experimented with using *selective indices* as a basis for our retrieval. We based our selection both on element length and tag-names. We have seen that we can improve the effectiveness of our retrieval system by excluding the smallest elements from our index—and hence from our retrieval runs. Removing the very long element from our index—whole articles and and article bodies—results in a significant decrease in performance.

Whole articles and bodies thus seem to be essential for achieving good retrieval scores. We conjecture that this—somewhat counter-intuitive—finding may be explained by a combination of factors. First, there may be *topic factors*: the information need behind many topics may be general and hence answered well by a complete article. Second, there may be *evaluation framework factors*: the assessors may have difficulty with understanding how to assess specificity; and/or the evaluation—even together with specificity-oriented quantization—may favor

exhaustivity over specificity. Third, there may be *system factors*: there may be considerable variance in the specificity assessments, meaning that the retrieval engine gets best results by going for the "safe" option—namely, go for exhaustivity.

Context Our next research question addressed the use of element context to improve XML element retrieval performance:

> Can we improve XML element ranking by incorporating element's context into the retrieval model?

In Chapter 6 we have used document language models to provide localized smoothing in addition to the standard collection model smoothing. Our experiments have shown that the mixture model gives significant improvements in retrieval performance. We have verified that this result holds for a range of indices, including both our full element index and our index of only sections and paragraphs.

Evaluation Evaluation frameworks for XML element retrieval have been and continue to be an active research area. Research on the evaluation framework is beyond the scope of this thesis, but we cannot ignore it altogether since our empirical work in this thesis relies on an evaluation framework being present. We did thus state a "passive" research question about evaluation.

> How does our system's performance change when we use a different evaluation setup?

In Chapters 5 and 6 we have seen the effect of changing the retrieval task from retrieving arbitrary elements to retrieving sections and paragraphs. Our main findings are that our length-priors and our mixture model continue to be important for achieving good retrieval performance.

In Chapter 7 we tried to dig deeper and tried to find explanations for the successes and failures of earlier chapters. To this end we examined whether our retrieval approach is tailored more toward certain *class of topics*. Our main findings are that using the strict quantization to optimize our system can lead to uneven performance over topic classes. Optimizing using the generalized quantizations does, however, lead to more even performance over different topic classes.

In Chapter 7 we have also looked at how *overlap* affects our evaluation. We have seen that considerable overlap exists both in the strict recall-base and in our retrieval runs. There is still further research needed on how this overlap should be handled by the evaluation framework—depending on whether or not we intend to exploit overlap when we turn the ranked list of elements into a user-oriented application. We have evaluated our system using a metric which does not reward retrieval of overlapping elements. That metric seems to be less stable than the metric which does reward overlap. Further research is needed to investigate if the

evaluation can be made more stable by rewarding some—but not all—overlap. This issue ties in with the question of whether we intend to exploit some kind of overlap in our end-usage of element retrieval results.

9.2 An Interface for Focused Information Access

In this thesis, the study of XML retrieval has been performed with a specific goal in mind, namely to use it as a back-end engine for an operational system which gives focused access to information. In our introductory chapter we formulated this in a research question:

> How can we put XML retrieval into action as a part of an operational system for giving focused information access?

In Chapter 8 we have explored how XML element retrieval can be put to action to give focused access to relevant information. We presented an interface where we use XML element retrieval to provide structured result lists and direct linking to relevant portions of relevant documents.

We have shown that there are many important *design choices* that need to be addressed when element retrieval is put into action. We have shown that the overlapping nature of XML elements can be exploited by presenting relevant documents as a partial tree. The partial tree preserved the reading order—a.k.a. document order—of the relevant elements. We have shown how the relevance ordering of elements can be displayed using a heat-map of the partial tree.

We have argued that *portability* is an important aspect of interfaces for focused information access. In case of a third-party document collection, the access points of the interface need to be in sync with the access points provided by the document collection. In turn, the retrieval units of the retrieval engine need to be in sync with the access points provided by the interface.

9.3 Focused Information Access using XML Element Retrieval

In the previous two sections we have listed our main conclusions and contributions for the two sub-questions of the main research question of this thesis. In this section we will reflect on how we can combine the answers to the two sub-questions to provide an answer to our main question:

> How can we give focused information access to semi-structured documents using XML element retrieval techniques?

This question was based on a scenario where we had a user who had a rather specific information need which was answered by a relatively small portion of a relevant document. We have built an effective XML element retrieval engine which retrieves the most relevant portions of documents and we have built an interface which uses the engine's results to provide users with structured result lists and direct links to the relevant information. A preliminary evaluation of our interface has shown that users find the structured result lists useful since it gives them a good overview of the retrieved documents. The users also appreciate the direct linking since it reduces the effort needed to access the relevant information. In sum, we have successfully utilized XML element retrieval to give focused access to information.

In this thesis we have addressed a rather specific application of XML element retrieval. We have directly addressed frequent concerns about the usefulness of XML element retrieval—see e.g., [Baeza-Yates et al., 2002, panel discussion] and [Trotman, 2005]. Our evaluation of the core XML element retrieval task has, however, been more general and our results do thus apply to a greater variety of applications.

9.4 Future Work

In this section we will discuss several directions in which our work in this thesis can be extended. We discuss the following issues: XML retrieval modeling and implementation, the transition from XML element retrieval to focused information access, the Cranfield assumptions, the definition of the notion of specific information need, and finally the operationalization of focused information access.

XML retrieval modeling and implementation In Section 3.3 we justified our choice of language modeling framework in terms of two benefits: first, language models offer an intuitive framework for mixture of evidence from various levels of the XML hierarchy, and second, language models offer flexible framework for incorporating non-content features into the retrieval using various priors. In this thesis we have made use of those benefits to stress the importance of two issues: *length normalization* using length priors and the use of *document context* in our mixture model. However, our usage has been simple and we have not exploited the full flexibility of the language modeling framework. There is plenty of room to extend XML element retrieval within the language modeling framework. For instance, the length-prior usage can be extended and other non-content priors can be introduced. Furthermore, one can extend the mixture modeling by taking into account more levels of the document hierarchy.

From XML element retrieval to focused information access In this thesis we have focused on a specific application of XML element retrieval, and eval-

uated a more general XML element retrieval task. However, in Chapter 7 we argued that if we have a specific application in mind we should adapt our retrieval task and evaluation framework to the specific application. Thus, we ask ourselves:

> How much of our XML general element retrieval evaluation results are useful when we put XML element retrieval it into action?

In Chapter 8 we have argued that we need to sync the unit of retrieval with the access points of the collection. We have seen in the experimental part of this thesis that the main retrieval features—the length priors and the mixture model—are useful for a variety of different indices—and hence different ranges of retrieval granularities. In that sense, we can say that some of the basic findings in Chapters 3–6 can be carried over to the operational setting.

However, there are some aspects that require further research. Our implementation of focused information access depends on the assumption that the notion of relevance coincides with the notion of (best) entry points [Reid et al., 2006a,b]. Based on the available data, we cannot judge whether this assumption is realistic. This issue is, however, currently being addressed as part of the INEX 2006 evaluation cycle [INEX].

In our evaluation of the XML element retrieval task we evaluated ranked lists of *elements*. In our interface, however, we gave focused access to *documents*. We only briefly addressed this transition between elements and documents by our evaluation in Section 7.3—where we evaluated the task of ranking documents based on element results. It remains as future work to evaluate this transition more thoroughly using for example the so-called Fetch-And-Browse task of INEX 2005 [Malik et al., 2006] or the so-called All-in-Context task of INEX 2006 [INEX].

Cranfield assumptions In Section 2.2.1 we introduced the Cranfield paradigm for laboratory evaluation of information retrieval. It is thus natural to ask:

> Are the Cranfield assumptions consistent with XML element retrieval?

The usual disclaimers—that the Cranfield assumptions are not true in practice [Voorhees, 2002]—hold for XML retrieval as well as other retrieval tasks. However, XML element retrieval brings some additional issues that may affect the assumptions. First, let us look at the assumption that assessment of one document is independent from the assessment of other documents. When assessing topical relevance of documents, the validity of this assumption can be enforced fairly easily—although one can may always expect some learning effect when assessors are performing assessments. However, for the assessments of XML elements the topical relevance of one element is clearly not independent from the relevance of all other elements—e.g., if an element is topically relevant then all its ancestors are topically relevant as well.

The discussion about topical relevance brings us to another Cranfield assumption, namely, that relevance assessments can be captured by topical similarity. The notion of specificity in XML element retrieval evaluation is dependent on the notion of topical similarity—i.e., topical similarity is a necessary, but not sufficient, requirement for an element to be considered as specific. Surely, there is something more to specificity.

This brings us to the next issue, namely, the relation between specificity and the assessor's background knowledge. One of the Cranfield assumptions says that a single set of assessments should be representative for the whole user population. For document retrieval this assumption is rarely true since background knowledge influences the assessment of topical similarity. For XML element retrieval background knowledge does not only influence the assessment of topical similarity, but may also influence the assessment of specificity. The more background knowledge the user has about the topic at hand, the less context—less text—she may need in order to judge a topical similarity. Hence, the assessment of specificity may also be influenced by her background knowledge.

We have thus seen that the same disclaimers hold for document retrieval and XML element retrieval—but their extent is different. Despite the disclaimers, the Cranfield paradigm is still considered as a valid method for comparing the performance of document retrieval systems. There is still further research needed to verify that—despite the additional "violations" of assumptions—the Cranfield is a valid framework for evaluating XML element retrieval.

Apart from assessment issues there are also questions about the overall setup of the experiment, in particular, the role of the ranked list. For many of the document retrieval evaluation metrics there is an assumption that the ranking of documents corresponds to the order in which the documents are utilized by a user. In our application of XML element retrieval this assumption is not true since we re-order the elements when we make the transition from XML element retrieval to focused information access. However, as mentioned above, it remains as future work to see if we can overcome this problem by considering a more appropriate element retrieval task.

Specific information needs Throughout this thesis we have used the notion of a *specific information need* without defining precisely what that notion means. We usually use the notion to refer to the scenario where the users' information need is answered by a relatively small portion of a longer document. Hence—in our terminology—a specific information need is a combination of three things: user's information need, the document collection, and the user's background knowledge—and perhaps also user's work task. It remains as future work to define this notion more precisely—or at least define precisely the types of (specific) information needs for which XML retrieval and focused information access are potentially useful.

The issue of specific-ness of information needs is being addressed within the 2006 cycle of the INEX initiative [Kamps and Larsen, 2006]. During the INEX 2006 topic development phase the topic author's view on the specific-ness of her information need was recorded. Preliminary results show that there is a relation between user's knowledge of the subject matter and the specific-ness of their search requests. Further analysis of this data is promising for providing insight into the notion of specific information need, and its relation to focused information access and XML element retrieval.

For additional inspiration—when defining the notion of specific information need—it might be useful to look to other tasks where relatively short textual answers are required—such as the "other questions"-task at TREC [Voorhees, 2005] and the CLEF 2006 WiQA task [WiQA].

Operationalizing focused information access We have motivated our approach as an extension of the state-of-the-art search engines. It is therefore natural to ask

> Will state-of-the-art search engines adopt our extension?

In principle, our XML retrieval techniques can be applied in the more general web setting where state-of-the-art search engines operate. However, there are some issues that need to be addressed before this can happen. One is the fact that document authors generally do not give multiple entry-points into their documents. Hence, with the current browser technology search engines can only give focused access to a small portion of the set of documents out there.

Appendix A

Additional Assessment Analysis

A.1 Overlap in the Strict Recall-base

In this section we show additional analysis of overlap in the strict recall base. This analysis is a follow up on the analysis performed in Section 7.2.1

Table A.1 shows the most frequent element types that overlap each other in the strict recall-base. The table shows the absolute number of cases in which the element types overlap (#); the fraction of topic-document pairs for which such an overlap pair exists (Documents); and the fraction of the topics for which such an overlap pair exists (Topics). The table is sorted by the number of documents containing the overlap pair. We see that for 20% of the documents which contain a strictly relevant element, both the complete article and the complete body have been assessed as highly exhaustive and highly specific. Furthermore, over half of the topics contain a document for which such an assessment pair exists. In terms of document frequency, this is the most frequent form of overlap in the strict recall-base. In terms of total frequency, sections (sec) and paragraphs (p) make up the element type pair to overlap. Sections and paragraphs overlap in 12% of the documents and this overlapping pair occurs in the strict recall-base for almost half of the topics.

Table A.2 shows the most frequent overlap pairs, broken up for each vintage of the INEX test collection. We see that the article-bdy overlap pair is frequent in the recall-base for the 2002-2004 topics. Is occurs in the recall-base of over 60% of the topics in each of those years. In 2003 and 2004 the pair is present in the recall base of roughly 30% of the documents which contain a strict assessment. The section-paragraph (sec-p) overlap pair is also frequent in the 2002–2004 collections. It occurs in the recall-base of over half of the topics each year. Its document frequency is, however, considerably lower than for the article-body pair. In 2005 the overlap picture is quite different from previous years. No overlap pair appears in the recall-base of a great number of topics, nor is there any pair with a high document frequency. In Section 5.1 we saw that articles

Table A.1: Most frequent element types which overlap in the INEX strict recall-base. For each ancestor-descendant pair we report the total frequency of such a pair (#); the fraction of documents in which such a pair occurs in the strict recall base (Documents); and the fraction of topics for which such a pair occurs in the strict recall-base (Topics).

Ancestor	Descendant	#	Documents	Topics
article	bdy	303	20%	52%
article	sec	448	15%	44%
bdy	sec	416	13%	45%
sec	p	697	12%	48%
sec	ss1	314	8.9%	46%
article	p	531	7.8%	29%
ss1	p	347	6.5%	40%
bdy	p	462	6.4%	30%
article	ss1	184	5.5%	33%
bdy	ss1	191	5.4%	30%
sec	ip1	120	5.1%	32%
abs	p	61	3.9%	21%
article	ip1	104	3.6%	23%
bdy	ip1	87	3.0%	21%
ss1	ip1	77	2.9%	22%
ss1	ss2	69	2.4%	23%
fig	art	52	2.4%	11%
bb	atl	53	1.8%	7.9%
sec	ss2	57	1.8%	19%
article	abs	27	1.7%	12%
article	fm	26	1.7%	14%

Table A.2: The most common overlapping element types in the strict recall-base for each vintage of the INEX test collection.

(a) 2002 assessments

Anc.	Dec.	#	Docs	Topics
article	bdy	88	14%	61%
article	sec	135	12%	48%
sec	p	71	7.5%	52%
bdy	sec	97	7.2%	35%
article	p	60	5.2%	26%
sec	ss1	46	4.6%	48%
ss1	p	39	3.5%	44%
abs	p	20	3.2%	22%
bdy	p	36	3.0%	22%
article	ss1	23	2.9%	35%
bdy	ss1	17	2.1%	22%
sec	ip1	13	1.7%	26%
ss1	ip1	12	1.6%	22%
article	fm	9	1.4%	17%
article	ip1	13	1.3%	22%

(b) 2003 assessments

Anc.	Dec.	#	Docs	Topics
article	bdy	134	35%	82%
bdy	sec	172	24%	82%
article	sec	168	23%	78%
sec	ss1	89	13%	67%
sec	p	167	13%	63%
bdy	ss1	68	10%	52%
article	ss1	67	9.7%	52%
article	p	137	8.9%	48%
bdy	p	117	8.7%	56%
sec	ip1	47	6.1%	37%
ss1	p	48	6.1%	41%
article	ip1	47	5.8%	33%
bdy	ip1	38	5.0%	33%
ss1	ss2	23	3.7%	30%
ss1	ip1	25	3.7%	30%

(c) 2004 assessments

Anc.	Dec.	#	Docs	Topics
sec	p	444	30%	56%
article	bdy	81	29%	64%
bdy	sec	147	22%	60%
article	sec	145	21%	48%
article	p	334	19%	40%
sec	ss1	142	17%	56%
bdy	p	309	17%	40%
ss1	p	228	17%	48%
sec	ip1	55	15%	48%
bdy	ss1	106	12%	44%
article	ss1	94	11%	44%
article	ip1	44	9.3%	36%
bdy	ip1	42	8.2%	36%
abs	p	23	8.2%	20%
bb	atl	39	7.5%	16%

(d) 2005 assessments

Anc.	Dec.	#	Docs	Topics
sec	ss1	37	3.8%	12%
abs	p	9	3.5%	23%
ss1	p	32	3.5%	27%
p	it	56	3.1%	15%
p	ref	12	3.1%	15%
fig	art	8	2.7%	15%
item	p	12	1.9%	19%
sec	p	15	1.9%	19%
ss1	ss2	7	1.9%	3.8%
fig	fgc	6	1.9%	15%
p	url	8	1.5%	7.7%
sec	ip1	5	1.5%	15%
li	p	6	1.5%	12%
ip1	ref	3	1.2%	7.7%
app	sec	2	0.8%	3.8%

and bodies disappeared from the strict recall-base in 2005. Consequentially, the overlap pair article-body also disappeared.

Appendix B

Additional Interface Evaluation Data

In this appendix we provide additional evaluation data for our two interactive evaluation studies. In Section B.1 we provide data on the INEX-IEEE case study. In Section B.2 we provide data on the Wikipedia case study.

B.1 Case study: INEX-IEEE Collection

In this section we provide additional data for the INEX-IEEE interactive study. The Section contains two types of additional data. The experimental setup of the interactive study is described in Section B.1.1 and additional results on users' answers to open questions are presented in Section B.1.2.

B.1.1 Experimental Setup

Our experiments were performed within the framework of the INEX interactive track [Larsen et al., 2006a]. The results presented in this study are derived from data gathered as part of the 'Task B' of the track. The purpose of the task was to compare the iTrack baseline system, Daffodil[1], to a home-grown system—in our case the adaption of our interface to the INEX-IEEE collection (see Section 8.3.1). Our test persons were 14 first year computer science students. Each test person searched for two simulated work tasks (a 'general' task and a 'challenging' task) [Borlund, 2003]. Additionally, all test persons were asked to think up search topic of their own. For details of the simulated tasks, we refer to [Larsen et al., 2006a]. The experimental matrix is shown in Table B.1. First, every test person searches for two simulated tasks, each one with a different system. Next, the test persons search for their own topic with a system of their choice.

The test persons filled in a pre-experiment questionnaire, as well as pre-task and post-task questionnaires before and after each task, and a post-experiment

[1] `http://www.is.informatik.uni-duisburg.de/projects/daffodil/index.html.en`

Table B.1: Experimental matrix for the comparative experiment. Tasks are general (g1, g2, or g3), challenging (c1, c2, or c3), or own (o); systems are Daffodil (D), our system (X), or a choice of system (?).

	Test-person													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Rotation	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Task 1	g1	c1	g1	c1	g2	c2	g2	c2	g3	c3	g3	c3	g1	c1
System 1	D	D	X	X	D	D	X	X	D	D	X	X	D	D
Task 2	c1	g1	c1	g1	c2	g2	c2	g2	c3	g3	c3	g3	c1	g1
System 2	X	X	D	D	X	X	D	D	X	X	D	D	X	X
Task 3	o	o	o	o	o	o	o	o	o	o	o	o	o	o
System 3	?	?	?	?	?	?	?	?	?	?	?	?	?	?

questionnaire after the whole experiment. In addition to the questionnaires, a wealth of system-user interaction was logged by the system. Specifically, we logged all links clicked on in the result window. Notably, we make a distinction between article-links and element-links. In terms of our example result (Figure 8.4 on page 152), we refer to the link on top (article title) as article-link and all the links below (XML element names) as element-links. For the element-links we logged various information such as the rank of the article, the RSV of the element, the highest RSV of the article, depth, etc. For all articles in which an element was viewed in detail, we also asked test persons to assess whether the whole article was regarded as relevant, or whether only parts of the article were relevant, or whether the whole article was non-relevant. The assessments were collected via a simple assessment interface (see the top right-hand side corner of the screenshot in Figure 8.5 on page 153). Although our experiment was set up as a comparative experiment, we focus on the data collected for our system only.[2] Out of 14 test persons, 10 chose to use our system for their 'own task.' Hence, in total we have 24 search sessions of 14 users.

B.1.2 Experimental Results

In this section we present the full answers to the open questions asked in our interactive evaluation of the INEX-IEEE collection.

First, we consider the question about which aspects of the system the users found useful:

> In what ways (if any) did you find the system interface useful in this task?

[2]The test persons appreciated both systems: on a 5-point scale Daffodil scored on average 2.8 and our system 3.5. There were external factors (system slow-down and reboot) that may have affected the data collected on Daffodil. Hence, the value of the comparison is unclear.

Table B.2: Answers to question Q3.15. (In what ways (if any) did you find the system interface useful in this task?)

• Title display of the article helped. Splitting the article helps.
• Best articles listed first.
• Best articles listed on top.
• You can see how the page is build. The structure is good.
• On the left side it has a relevance indication; It says what parts of the article the relevant information is located, and you can click on it; Nice overview of the document.
• very clear; simple but effective search field.
• very clear; simple but effective search field.
• It got me where I wanted eventually.
• That I did not have to read the whole articles.
• Overview of the table of contents.
• I only needed to type 'wifi' and the first two hits gave already the answer.
• It is very nicely ordered.
• I like the pop-up screen so that the desired result isn't displayed in the same window.
• The highlighted paragraphs can be useful, but the best thing were the search results. The top article had all the information necessary. But it can be nice when searching for long and boring articles.
• It showed the right information in a high-lighted paragraph.
• Its so simple. Just one input box and a search button. The results are listed so good, just like an XML tree model. This is what a search engine must look like in my opinion.
• Its perfect, same as task 2.
• The navigation was easy because of the text on the left.
• It gives the names of different paragraphs.

Table B.2 shows the full list of answers that the users gave to the question. Note that the 19 answers are collected using post-task questionnaire for all 24 search sessions of the 14 users who took part in the experiments. Hence, some of the answers that look quite similar may have originated from the same user.

Next, we consider the question about which aspects of the system the users found not useful:

> In what ways (if any) did you find the system interface *not* useful in this task?

Table B.3 shows the full list of answers that the users gave to this question. The table contains 11 answers are collected using post-task questionnaires for all 24 search sessions of the 14 users who took part in the experiments. Some users may

Table B.3: Answers to question Q3.16. (In what ways (if any) did you find the system interface *not* useful in this task?)

• The scroll function in the article is not working.
• Not.
• Showing much not useful information.
• Only had articles, no pictures, or comparison tables between the videocards.
• Parts that is not important, such as `<blabla> <p> <article> <p><blabla>`
• Relevance indication often confusing.
• I don't think its useful that it shows where in the article the words are (in which body, section, etc.).
• I don't think its useful that it shows where in the article the words are (in which body, section, etc.).
• Took me a very long time to find a good article.
• It still takes a long time to find something useful, hard to define good searching terms.
• A bit unclear after you click on 'search': a lot of text at once, unorganized in front of you, causing confusion.
• The description of the relevant part of the article is too short.
• It could be nice if it was possible to remove the shown paragraphs in the search results so more results fit on a screen.
• In xmlfind you can't find for just a title, but it is still much better than Daffodil.
• It also gives the XML layout. That isn't important for someone who doesn't understand XML-code.

be responsible for two answers—one from each post-task questionnaire—while others did not answer the question at all.

B.2 Case study: Wikipedia

B.2.1 Experimental Setup

This section describes the experimental setup for the interactive evaluation of the application of our interface to Wikipedia [Sigurbjörnsson et al., 2006].

In order to answer our research questions we set up an interactive experiment where we asked people to perform simulated work tasks [Borlund and Ingwersen, 1997]. An example of a simulated work task can be seen in Figure B.1. The actual work tasks that were used in the experiment can be found in Figures B.2 and B.3. Each of the actual work tasks consisted of three related search assignments. Each search assignment resembled a factoid question or a list question.

We have created two interfaces to our Wikipedia search engine. One is a simple baseline interface which gives access to the start of Wikipedia pages only,

Motivation Suppose you have just seen a report on the news about the recent earthquake in Pakistan. The report makes you want to get a better understanding of the Pakistan earthquake region.
Task Please use the Wikipedia search engine to find the answer to the following questions:

- Where is Pakistan precisely?
- In which parts of Pakistan is there a great risk of earthquakes?
- What causes the earthquakes in Pakistan?
- Is there a difference between the cause of earthquakes in Pakistan, compared to other earthquake areas, such as California, Japan, or Iceland?

Figure B.1: Example of a possible simulated work task.

while the other is a focused interface which gives access to individual sections of Wikipedia pages (see Section 8.3.3 for more information on the focused interface). Our *baseline* search interface is a Google-like one where each result is presented as a pair: a link to the relevant page, and a short query dependent summary of the page in the form of a snippet. A screen-shot of the interface is shown in Figure 8.10 on page 163.

The element entry-points and element snippets are the only difference between the two wikipedia interfaces. They use the same underlying ranking scheme which means that documents are ranked precisely in the same order. The ranking of the documents is based on aggregated score of the relevant sections. The snippet used in the baseline system is created by concatenating the snippets of relevant sections. This means that both interfaces present precisely the same text to the user.

Each test subject performed two simulated work tasks, but using different system each time. The experiment matrix is shown in Table B.4. Our analysis is based on 12 test persons, evenly distributed between the two rotations.[3]

The rotation removes the bias which is introduced by using one system before the other. The order of the simulated work tasks is always the same, leading to a potential interaction between the results for task I and task II.

In the beginning of the experiment the test person was asked to fill in a pre-experiment questionnaire on her background. After each task the user was asked to fill in a post-task questionnaire on her search experience during the task. Finally, the user was asked to fill in an post-experiment questionnaire after both task had been completed. The experiment, hence, involved the following steps:

[3]In the original experiment there we 16 test cases, but from the system logs we found out that 4 of them did not fully follow the experiment guidelines.

Stel je bereidt je voor op het komende WK voetbal dat dit jaar in Duitsland wordt gehouden. Om in de juiste stemming te komen wil je wat meer weten over het volgende...

1. Wie heeft het eerste WK voetbal gewonnen en heeft dat land daarna ooit nog eens het kampioenschap gewonnen? Zo ja wanneer?

Stel je voor dat je naar een basketbalwedstrijd kijkt, die wordt gehouden tijdens de olympische spelen. Je vraagt je ineens af of basketbal altijd al een olympische sport is geweest. Dit blijkt wel het geval te zijn. Vervolgens stel je jezelf de volgende vraag...

2. Wie heeft de basketbal wedstrijd gewonnen tijdens de eerste olympische spelen?

Stel je voor dat je een vrouw bent en voetbal speelt. Je wilt wel eens weten wat er nou zo bijzonder is aan voetbal spelen op topniveau voor zowel mannen als vrouwen. Je stelt jezelf de volgende vraag...

3. Noem drie verschillen tussen het WK voetbal voor mannen en het WK voetbal voor vrouwen.

Figure B.2: Task I: Simulated work task

Table B.4: Experimental matrix for the interactive experiment.

Rotation	Task I	Task II
1	Baseline	Focused
2	Focused	Baseline

1. Pre-experiment questionnaire
2. Simulated work task I
3. Post-task questionnaire
4. Simulated work task II
5. Post-task questionnaire
6. Post-experiment questionnaire

Our system logs various interactions between the user and the system. This data can be used to better understand how users interact with our system.

- *Queries*: All queries posted by users are logged.

Je probeert voor 't eerst mee te doen met de traditionele superbowl weddenschappen. Maar voor je je inzet kunt bepalen vraag je je af:

1. Welk football team heeft de eerste superbowl gewonnen, En heeft dit team daarna nog eens gewonnen? Zo ja, hoe vaak?

Je staat in de snowboard winkel, en vraagt je opeens af wanneer voor 't eerst snowboarden als olympische sport werd erkend... En je denkt:

2. Wie heeft de eerste olympische snowboard competitie gewonnen? [cat. Men's giant slalom]

Terwijl je op de bank zit te zappen, kom je bij eurosport opeens een sumo wedstrijd tegen. Waarop je je eigenlijk afvraagt hoe dat eigenlijk zit in de Verenigde Staten, bij football. Dus wil je weten:

3. Noem 3 verschillen tussen de Woman's professional football league [WPFL] en de [heren] football league [NFL].

Of

3. Noem 3 verschillen tussen [amateur, IFBB] body building competities tussen heren & dames.

Figure B.3: Task II: Simulated work task

- *Visited Results*: The system stores information about which links on the result pages are clicked on by the user.
- *Site Navigation*: All internal navigation between Wikipedia pages is logged.

Bibliography

D. Ahn, V. Jijkoun, K. Müller, M. de Rijke, and E. Tjong Kim Sang. Towards an offline XML-based strategy for answering questions. In *Accessing Multilingual Information Repositories: 6th Workshop of the Cross-Language Evaluation Forum, CLEF 2005*, volume 4022 of *Lecture Notes in Computer Science*, pages 449–456. Springer-Verlag, 2006.

P. Arvola, M. Junkkari, and J. Kekäläinen. Generalized contextualization method for XML information retrieval. In *CIKM '05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 20–27, New York, NY, USA, 2005. ACM Press.

Ask Jeeves. `http://ask.com`. Accessed in 2006.

R. Baeza-Yates, N. Fuhr, and Y. S. Maarek. Second edition of the "XML and information retrieval" workshop. *SIGIR Forum*, 36(2):53–57, 2002.

T. Bakker, M. Bedeker, S. van den Berg, P. van Blokland, J. de Lau, O. Kiszer, S. Reus, and J. Salomon. Evaluating XML retrieval interfaces: xmlfind. Technical report, University of Amsterdam, 2005.

M. Beaulieu, S. E. Robertson, and E. M. Rasmussen. Evaluating interactive systems in TREC. *Journal of the American Society for Information Science*, 47(1):85–94, 1996.

N. J. Belkin, A. Keller, D. Kelly, J. P. Carballo, C. Sikora, and Y. Sun. Support for question-answering in interactive information retrieval: Rutgers' TREC-9 interactive track experience. In *The Ninth Text REtrieval Conference (TREC 9)*, pages 463–489, 2001.

A. Berger and J. Lafferty. Information retrieval as statistical translation. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference*

on Research and Development in Information Retrieval, pages 222–229. ACM Press, 1999.

S. Betsi, M. Lalmas, A. Tombros, and T. Tsikrika. User expectations from XML element retrieval. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 611–612. ACM Press, 2006.

S. Bird, Y. Chen, S. Davidson, H. Lee, and Y. Zheng. Extending XPath to support linguistic queries. In *Proceedings of Programming Language Technologies for XML (PLANX)*, pages 35–46, 2005.

P. Borlund. The IIR evaluation model: a framework for evaluation of interactive information retrieval systems. *Information Research*, 8(3), 2003. `http://informationr.net/ir/8-3/paper152.html`.

P. Borlund and P. Ingwersen. The development of a method for the evaluation of interactive information retrieval systems. *Journal of Documentation*, 53(3): 225–250, 1997.

T. Bray, J. Paoli, and C. M. Sperberg-McQueen, editors. *Extensible Markup Language (XML) 1.0.* W3C, 1998. `http://www.w3.org/TR/1998/REC-xml-19980210`.

A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

O. Buyukkokten, H. Garcia-Molina, and A. Paepcke. Focused web searching with pdas. In *Proceedings of the 9th International World Wide Web Conference on Computer Networks: The International Journal of Computer and Telecommunications Netowrking*, pages 213–230. North-Holland Publishing Co., 2000.

J. P. Callan. Passage-level evidence in document retrieval. In *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302–310, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

D. Carmel, Y. Maarek, and A. Soffer. XML and information retrieval: a SIGIR 2000 workshop. *SIGIR Forum*, 34(1):31–36, 2000.

D. Carmel, Y. S. Maarek, Y. Mass, N. Efraty, and G. M. Landau. An extension of the vector space model for querying XML documents via XML fragments. In *Proceedings SIGIR 2002 Workshop on XML and Information Retrieval*, pages 14–25, 2002.

D. Carmel, Y. S. Maarek, M. Mandelbrod, Y. Mass, and A. Soffer. Searching xml documents via xml fragments. In *SIGIR '03: Proceedings of the 26th*

Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 151–158. ACM Press, 2003.

C. W. Choo, B. Detlor, and D. Turnbull. Information seeking on the web – an integrated model of browsing and searching. *First Monday*, 5(2), 2000. `http://firstmonday.org/issues/issue5_2/choo/index.html`.

C. L. Clarke and P. L. Tilker. MultiText experiments for INEX 2004. In Fuhr et al. [2005], pages 85–87.

C. Cleverdon. The Cranfield tests on index language devices. In *ASLIB Proceedings*, volume 19, pages 173–194, 1967.

W. S. Cooper. Some inconsistencies and misidentified modeling assumptions in probabilistic information retrieval. *ACM Transactions on Information Systems*, 13(1):100–111, 1995.

M. Cutler, Y. Shih, and W. Meng. Using the structure of HTML documents to improve retrieval. In *USENIX Symposium on Internet Technologies and Systems*, pages 241–252, 1997.

A. de Vries, G. Kazai, and M. Lalmas. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *Proceedings of RIAO (Recherche d'Information Assistée par Ordinateur (Computer Assisted Information Retrieval))*, pages 463–473, 2004.

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

B. Dervin. An overview of sense-making research: Concepts, methods, and results to date. In *International Communication Association Meeting, Dallas, TX*, 1983.

S. T. Dumais and N. J. Belkin. The TREC interactive tracks: Putting the user into search. In E. M. Voorhees and D. K. Harman, editors, *TREC: Experiment and Evaluation in Information Retrieval*, pages 123–152. MIT Press, 2005.

D. Ellis and M. Haugan. Modelling the information seeking patterns of engineers and research scientists in an industrial environment. *Journal of Documentation*, 53(4):384–403, 1997.

D. Ellis, D. Cox, and K. Hall. A comparison of the information seeking patterns of researchers in the physical and social sciences. *Journal of Documentation*, 49(4):356–369, 1993.

EvalJ. INEX evaluation package. `http://evalj.sourceforge.net/`. Accessed in 2006.

N. Fuhr and K. Großjohann. XIRQL: A query language for information retrieval in XML documents. In D. H. Kraft, W. B. Croft, D. J. Harper, and J. Zobel, editors, *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 172–180. ACM Press, New York NY, USA, 2001.

N. Fuhr, N. Gövert, G. Kazai, and M. Lalmas, editors. *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2002)*, 2003a. ERCIM.

N. Fuhr, K. Großjohann, and S. Kriewel. *A Query Language and User Interface for XML Information Retrieval*, volume 2818 of *Lecture Notes in Computer Science*, pages 59–75. Springer, 2003b.

N. Fuhr, M. Lalmas, and S. Malik, editors. *INEX 2003 Workshop Proceedings*, 2004.

N. Fuhr, M. Lalmas, S. Malik, and S. Szlávik, editors. *Advances in XML Information Retrieval: Third Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2004)*, volume 3493 of *Lecture Notes in Computer Science*, 2005. Springer.

N. Fuhr, M. Lalmas, S. Malik, and G. Kazai, editors. *Advances in XML Information Retrieval and Evaluation: Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005)*, volume 3977 of *Lecture Notes in Computer Science*, 2006. Springer-Verlag.

K. Fujimoto, T. Shimizu, N. Terada, K. Hatano, Y. Suzuki, T. Amagasa, H. Kinutani, and M. Yoshikawa. Implementation of a high-speed and high-precision XML information retrieval system on relational databases. In Fuhr et al. [2006], pages 254–267.

S. Geva. GPX – gardens point XML information retrieval at INEX 2004. In Fuhr et al. [2005], pages 211–223.

Google. `http://google.com`. Accessed in 2006.

N. Gövert and G. Kazai. Overview of the initiative for the evaluation of XML retrieval. In Fuhr et al. [2003a], pages 1–17.

N. Gövert, M. Abolhassani, N. Fuhr, and K. Grossjohan. Content-based XML retrieval with HyRex. In Fuhr et al. [2003a], pages 26–32.

B. F. Green, A. K. Wolf, C. Chomsky, and K. Laughery. Baseball: an automatic question answerer. *Computers & thought*, pages 207–216, 1963.

W. Greiff and W. Morgan. Contributions of language modeling to the theory and practice of information retrieval. In W. Croft and J. Lafferty, editors, *Language Modeling for Information Retrieval*, pages 73–93. Kluwer Academic Publishers, 2003.

K. Großjohann, N. Fuhr, D. Effing, and S. Kriewel. A user interface for XML document retrieval. In *Informatik bewegt: Informatik 2002 - 32. Jahrestagung der Gesellschaft für Informatik e.v. (GI)*, pages 166–170, 2002.

D. A. Grossman and O. Frieder. *Information Retrieval: Algorithms and Heuristics*. Sringer, 2004.

T. Grust. Accelerating XPath Location Steps. In *Proc. SIGMOD*, pages 109–120. ACM Press, 2002.

D. K. Harman. The TREC test collections. In E. M. Voorhees and D. K. Harman, editors, *TREC: Experiment and Evaluation in Information Retrieval*, pages 79–98. MIT Press, 2005.

D. J. Harper, I. Koychev, Y. Sun, and I. Pirie. Within-document retrieval: A user-centred evaluation of relevance profiling. *Information Retrieval*, 7(3-4): 265–290, 2004.

M. A. Hearst. TileBars: visualization of term distribution information in full text information access. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 59–66. ACM Press/Addison-Wesley Publishing Co., 1995.

M. A. Hearst. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23:33–64, 1997.

M. A. Hearst. *Modern Information Retrieval*, chapter 10. Addison Wesley Longman Publishing Co. Inc., 1999.

M. A. Hearst and C. Plaunt. Subtopic structuring for full-length document access. In *Proceedings of the 16th Annual International ACM SIGIR Conference*, pages 56–68. ACM Press, New York NY, USA, 1993.

W. R. Hersh. The TREC 2002 interactive track report. In *The Eleventh Text REtrieval Conference (TREC 2002)*, 2003.

W. R. Hersh and P. Over. The TREC-9 interactive track report. In *The Ninth Text REtrieval Conference (TREC 9)*, pages 42–50, 2001.

W. R. Hersh and P. Over. The TREC 2001 interactive track report. In *The Tenth Text REtrieval Conference (TREC 2001)*, 2002.

D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, 2001.

ILPS-Lucene. The ILPS extension of the Lucene search engine. `http://ilps.science.uva.nl/Resources/`. Accessed in 2006.

INEX. Initiative for the evaluation of xml retrieval (inex). `http://inex.is.informatik.uni-duisburg.de/`. Accessed in 2006.

P. Ingwersen. Cognitive perspectives of information retrieval interaction. *Journal of Documentation*, 52(1):3–50, 1996.

J. Kamps and B. Larsen. Understanding differences between search requests in XML element retrieval. In *SIGIR 2006 Workshop on XML Element Retrieval Methodology*, pages 13–19, 2006.

J. Kamps and B. Sigurbjörnsson. What do users think of an XML element retrieval system? In Fuhr et al. [2006], pages 411–421.

J. Kamps, M. de Rijke, and B. Sigurbjörnsson. Topic Field Selection and Smoothing for XML Retrieval. In A. P. de Vries, editor, *Proceedings of the 4th Dutch-Belgian Information Retrieval Workshop*, pages 69–75, 2003a.

J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjörnsson. XML Retrieval: What to Retrieve? In *Proceedings of the 26th ACM SIGIR Conference*, pages 409–410, 2003b.

J. Kamps, M. de Rijke, and B. Sigurbjörnsson. Length normalization in XML retrieval. In M. Sanderson, K. Järvelin, J. Allan, and P. Bruza, editors, *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 80–87. ACM Press, New York NY, USA, 2004a.

J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjörnsson. Best-match querying from document-centric XML. In S. Amer-Yahia and L. Gravano, editors, *Proceedings of the Seventh International Workshop on the Web and Databases (WebDB 2004)*, pages 55–60, 2004b.

J. Kamps, M. de Rijke, and B. Sigurbjörnsson. The importance of length normalization for XML retrieval. *Information Retrieval*, 8(4):631–654, 2005a.

J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjörnsson. Structured queries in XML retrieval. In *CIKM'05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 2–11. ACM Press, New York NY, USA, 2005b.

J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjörnsson. Articulating information needs in XML query languages. *ACM Transactions on Information Systems*, 2006. To appear.

N. Kando, T. Koyama, K. Oyama, M. Kageura, K.and Yoshioka, T. Nozue, A. Matsumura, and K. Kuriyama. NTCIR: NACSIS test collection project. In *The 20th Annual Colloquium of the British Computer Society Information Retrieval Specialist Group*, 1998.

M. Kaszkiel and J. Zobel. Passage retrieval revisited. In *SIGIR '97: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185, New York, NY, USA, 1997. ACM Press.

G. Kazai and M. Lalmas. INEX 2005 evaluation metrics. In Fuhr et al. [2006], pages 16–29.

G. Kazai, M. Lalmas, and A. de Vries. The overlap problem in content-oriented XML retrieval evaluation. In *SIGIR '04: Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval*, pages 72–79. ACM Press, 2004a.

G. Kazai, M. Lalmas, N. Fuhr, and N. Gövert. A report on the first year of the initiative for the evaluation of XML retrieval (INEX'02). *Journal of the American Society for Information Science and Technology*, 55(6):551–556, 2004b.

G. Kazai, M. Lalmas, and A. de Vries. Reliability tests for the XCG and INEX-2002 metrics. In Fuhr et al. [2005], pages 60–72.

M. Kearns, Y. Mansour, A. Y. Ng, and D. Ron. An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27(1):7–50, 1997.

W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 27–34, 2002.

M. Lalmas and T. Rölleke. Modelling vague content and structure querying in XML retrieval with a probabilistic object-relational framework. In *Proceedings of the 6th International Conference on Flexible Query Answering Systems, FQAS 2004*, volume 3055 of *Lecture Notes in Computer Science*, pages 432–445. Springer, 2004.

B. Larsen, S. Malik, and T. Tombros. The interactive track at INEX 2005. In Fuhr et al. [2006], pages 398–410.

B. Larsen, T. Tombros, and S. Malik. Is XML retrieval meaningful to users? Searcher preferences for full documents vs. elements. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 663–664. ACM Press, 2006b.

LaTeX. `http://www.latex-project.org/`. Accessed in 2006.

J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger. The role of context in question answering systems. In *CHI '03: CHI '03 Extended Abstracts on Human Factors in Computing Systems*, pages 1006–1007, New York, NY, USA, 2003a. ACM Press.

J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger. What makes a good answer? The role of context in question answering. In *Proceedings of the Ninth IFIP TC13 International Conference on Human-Computer Interaction (INTERACT-2003)*, 2003b.

J. List, V. Mihajlović, A. de Vries, G. Ramírez, and D. Hiemstra. The TIJAH XML-IR system at INEX 2003. In Fuhr et al. [2004], pages 102–109.

J. List, V. Mihajlović, G. Ramírez, A. P. de Vries, D. Hiemstra, and H. E. Blok. TIJAH: Embracing IR methods in XML databases. *Information Retrieval*, 8 (4):547–570, 2005.

X. Liu and W. B. Croft. Passage retrieval based on language models. In *CIKM '02: Proceedings of the eleventh International Conference on Information and Knowledge Management*, pages 375–382. ACM Press, 2002.

W. Lu, S. E. Robertson, and A. MacFarlane. Field-weighted XML retrieval based on BM25. In Fuhr et al. [2006], pages 161–171.

Lucene. Lucene, an open-source search software. `http://lucene.apache.org/`. Accessed in 2006.

S. Malik, M. Lalmas, and N. Fuhr. Overview of INEX 2004. In Fuhr et al. [2005], pages 1–15.

S. Malik, G. Kazai, M. Lalmas, and N. Fuhr. Overview of INEX 2005. In Fuhr et al. [2006], pages 1–15.

Y. Mass and M. Mandelbrod. Retrieving the most relevant XML components. In Fuhr et al. [2004], pages 53–58.

Y. Mass and M. Mandelbrod. Component ranking and automatic query refinement for XML retrieval. In Fuhr et al. [2005], pages 73–85.

V. Mihajlović, H. E. Blok, D. Hiemstra, and P. M. G. Apers. Score region algebra: building a transparent XML-IR database. In *CIKM '05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 12–19, New York, NY, USA, 2005. ACM Press.

D. R. H. Miller, T. Leek, and R. M. Schwartz. A hidden markov model information retrieval system. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 214–221. ACM Press, 1999.

MS-Word. `http://office.microsoft.com/word/`. Accessed in 2006.

MSN Encarta. `http://encarta.msn.com`. Accessed in 2006.

MSN Search. `http://search.msn.com`. Accessed in 2006.

S. H. Myaeng, D.-H. Jang, M.-S. Kim, and Z.-C. Zhoo. A flexible model for retrieval of SGML documents. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 138–145. ACM Press, 1998.

R. Navarro-Prieto, M. Scaife, and Y. Rogers. Cognitive strategies in web searching. In *Proceedings of the 5th Conference on Human Factors & the Web*, 1999. URL `http://zing.ncsl.nist.gov/hfweb/proceedings/navarro-prieto/index.html`.

P. Ogilvie and J. Callan. Using language models for flat text queries in XML retrieval. In Fuhr et al. [2004], pages 12–18.

P. Ogilvie and J. Callan. Hierarchical language models for XML component retrieval. In Fuhr et al. [2005], pages 224–237.

P. Ogilvie and J. Callan. Parameter estimation for a simple hierarchical generative model for XML retrieval. In Fuhr et al. [2006], pages 211–224.

P. Over. The TREC interactive track: an annotated bibliography. *Information Processing and Management*, 37(3):369–381, 2001.

PDF. Portable document format. `http://www.adobe.com/products/acrobat/adobepdf.html`. Accessed 2006.

C. Peters and M. Braschler. Cross-language system evaluation: the CLEF campaigns. *Journal of the American Society for Information Science and Technology*, 52(12):1067–1072, 2001.

B. Piwowarski and G. Dupret. Evaluation in (XML) information retrieval: expected precision-recall with user modelling (EPRUM). In *SIGIR '06: Proceedings of the 29th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 260–267, New York, NY, USA, 2006. ACM Press.

B. Piwowarski and M. Lalmas. Providing consistent and exhaustive relevance assessments for XML retrieval evaluation. In *CIKM '04: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, pages 361–370, New York, NY, USA, 2004. ACM Press.

J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281. ACM Press, 1998.

J. Prager, E. Brown, A. Coden, and D. Radev. Question-answering by predictive annotation. In *SIGIR '00: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 184–191, New York, NY, USA, 2000. ACM Press.

D. Raggett, A. L. Hors, and I. Jacobs, editors. *HTML 4.01 Specification*. W3C, 1999. `http://www.w3.org/TR/html4/`.

G. Ramirez, T. Westerveld, and A. P. de Vries. Structural features in content oriented XML retrieval. In *CIKM '05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 291–292, New York, NY, USA, 2005. ACM Press.

G. Ramirez, T. Westerveld, and A. de Vries. Using small XML elements to support relevance. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 693–694, 2006.

J. Reid, M. Lalmas, K. Finesilver, and M. Hertzum. Best entry points for structured document retrieval – part I: Characteristics. *Information Processing and Management*, 42:74–88, 2006a.

J. Reid, M. Lalmas, K. Finesilver, and M. Hertzum. Best entry points for structured document retrieval – part II: Types, usage and effectiveness. *Information Processing and Management*, 42:89–105, 2006b.

J. R. Remde, L. M. Gomez, and T. K. Landauer. Superbook: an automatic tool for information exploration — hypertext? In *HYPERTEXT '87: Proceeding of the ACM conference on Hypertext*, pages 175–188, New York, NY, USA, 1987. ACM Press.

S. Robertson and K. Spark Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *CIKM '04: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, pages 42–49, New York, NY, USA, 2004. ACM Press.

D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proceedings of WWW 2004*, pages 13–19, 2004.

G. Salton, C. Yang, and A. Wong. A vector-space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

G. Salton, J. Allan, and C. Buckley. Approaches to passage retrieval in full text information systems. In *SIGIR '93: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–58. ACM Press, 1993.

M. Sanderson and J. Zobel. Information retrieval system evaluation: effort, sensitivity, and reliability. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 162–169, New York, NY, USA, 2005. ACM Press.

B. Sigurbjörnsson and J. Kamps. The effect of structured queries and selective indexing on XML retrieval. In Fuhr et al. [2006], pages 104–118.

B. Sigurbjörnsson and A. Trotman. Queries, INEX 2003 working group report. In *Proceedings of the 2nd INEX Workshop*, pages 167–170, 2004.

B. Sigurbjörnsson, J. Kamps, and M. de Rijke. An element-based approach to XML retrieval. In N. Fuhr, S. Malik, and M. Lalmas, editors, *INEX 2003 Workshop Proceedings*, pages 19–26, 2004a.

B. Sigurbjörnsson, J. Kamps, and M. de Rijke. Processing content-oriented XPath queries. In *Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management (CIKM 2004)*, pages 371–380, 2004b.

B. Sigurbjörnsson, J. Kamps, and M. de Rijke. Multiple sources of evidence for XML retrieval. In *SIGIR '04: Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval*, pages 554–555. ACM Press, 2004.

B. Sigurbjörnsson, J. Kamps, and M. de Rijke. Mixture-models, overlap, and structural hints in XML element retrieval. In Fuhr et al. [2005], pages 196–210.

B. Sigurbjörnsson, J. Kamps, and M. de Rijke. Focused access to Wikipedia. In F. de Jong and W. Kraaij, editors, *6th Dutch-Belgian Information Retrieval Workshop (DIR 2006)*, pages 73–80, 2006.

A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of the 19th Annual International ACM SIGIR Conference*, pages 21–29. ACM Press, 1996a.

A. Singhal, G. Salton, M. Mitra, and C. Buckley. Document length normalization. *Information Processing & Management*, 32:619–633, 1996b.

D. J. Slone. Internet search approaches: The influence of age, search goals, and experience. *Library & Information Science Research*, 25(4):403–418, 2003.

Snowball. `http://snowball.tartarus.org/`. Accessed in 2006.

F. Song and W. B. Croft. A general language model for information retrieval. In *CIKM '99: Proceedings of the eighth International Conference on Information and Knowledge Management*, pages 316–321. ACM Press, 1999.

J. Tague-Sutcliffe. The pragmatics of information retrieval experimentation, revisited. *Information Processing and Management*, 28(4):467–490, 1992.

T. T. Tang, D. Hawking, N. Craswell, and K. Griffiths. Focused crawling for both topical relevance and quality of medical information. In *CIKM '05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 147–154, New York, NY, USA, 2005. ACM Press.

A. Tombros, B. Larsen, and S. Malik. The interactive track at INEX 2004. In Fuhr et al. [2005], pages 410–423.

A. Tombros, B. Larsen, and S. Malik. Report on the INEX 2004 interactive track. *SIGIR Forum*, 39:43–49, 2005b.

E. Toms, L. Freund, R. Kopak, and J. Bartlett. The effect of task domain on search. In *Proceedings of CASCON 2003*, pages 1–9, 2003.

trec_eval. Text retrieval conference evaluation software (trec_eval). `http://trec.nist.gov/trec_eval`. Accessed in 2006.

A. Trotman. Wanted: Element retrieval users. In *Proceedings of the INEX 2005 Workshop on Element Retrieval Methodology*, pages 63–69. University of Otago, Dunedin New Zealand, 2005.

A. Trotman and B. Sigurbjörnsson. Narrowed Extended XPath I (NEXI). In Fuhr et al. [2005], pages 219–236.

A. Trotman and B. Sigurbjörnsson. NEXI, now and next. In Fuhr et al. [2005], pages 41–53.

E. Voorhees. The philosophy of information retrieval evaluation. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck, editors, *Evaluation of Cross-Language Information Retrieval Systems: Second Workshop of the Cross-Language Evaluation Forum, CLEF 2001*, volume 2406 of *Lecture Notes in Computer Science*, pages 355–370. Springer-Verlag, 2002.

E. M. Voorhees. Question answering in TREC. In E. M. Voorhees and D. K. Harman, editors, *TREC: Experiment and Evaluation in Information Retrieval*, pages 123–152. MIT Press, 2005.

E. M. Voorhees and D. K. Harman, editors. *TREC: Experiment and Evaluation in Information Retrieval*. The MIT Press, 2005.

Wikipedia. `http://wikipedia.org/`. Accessed in 2006.

R. Wilkinson. Effective retrieval of structured documents. In *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 311–317. Springer-Verlag New York, Inc., 1994.

T. Wilson. Models in information behaviour research. *Journal of Documentation*, 55(3):249–2700, 1999.

T. Wilson and C. Walsh. *Information behaviour: an interdisciplinary perspective*. The British Library Board, 1996. `http://informationr.net/tdw/publ/infbehav/`.

WiQA. WiQA: Question answering using Wikipedia. `http://ilps.science.uva.nl/WiQA/`. Accessed in 2006.

I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes*. Morgan Kaufmann, 1999.

World Book. `http://www.mackiev.com/world_book.html`. Accessed in 2006.

Yahoo! `http://yahoo.com`. Accessed in 2006.

C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *Transactions on Information Systems*, 22(2): 179–214, 2004.

Summary

Search engines play an important role in our daily lives. Most of us use search engines every day in search of information or services. State-of-the-art Internet search engines use a simple—yet powerful—interface. The user describes her information need by a few keywords and the search engine returns a list of documents that are likely to answer her information need. Each document is presented using its title and a short summary of the document's content—a.k.a. snippet. By clicking on a document title the user is routed to the corresponding document.

When a user is presented with a ranked list of relevant documents her search task is usually not over. The next step for her is to dive into the documents themselves in search for the precise piece of information she is looking for. When searching long documents this can be a tedious and time consuming task. Thus we ask ourselves: can we give the user a more focused type of access to the relevant information in this scenario?

In this thesis we study this question in the context of semi-structured document collections—more precisely, XML documents. Using the XML language, various document structure can be encoded—such as sections, sub-sections, paragraphs, section titles, author names, italicized text, etc. The XML markup divides the document text into a hierarchy of text objects—a.k.a. elements. In the thesis we ask ourselves whether we can exploit the hierarchical structure to give users a more focused access to the relevant information?

Our approach is twofold. First, we build a search engine that returns relevant XML elements as a response to the user's query. Second, we build an interface that uses the list of relevant elements to give focused access to the relevant documents. We evaluate the XML search engine using the so-called INEX test collection and evaluate the interface in an interactive experiment.

The main contribution of this thesis is to provide extensive evaluation of various methods for retrieving XML elements. We also show how the search engine can be put into action via a simple interface.

Samenvatting

Zoekmachines spelen een belangrijke rol in ons dagelijks leven. Velen van ons gebruiken ze dagelijks om informatie en diensten te vinden. De meest populaire Internet zoekmachines gebruiken een eenvoudig—maar effectieve—interface. De gebruiker beschrijft haar informatiebehoefte door middel van een aantal trefwoorden en de zoekmachine geeft een lijst van documenten weer die de informatiebehoefte beantwoorden. Elk relevant document wordt gepresenteerd met de titel van het document en een korte samenvatting van de inhoud—een zogenaamde "snippet." Door te klikken op de titel van een document wordt de gebruiker naar het corresponderende document gestuurd.

Nadat de zoekmachine de resultaten heeft weergegeven is de gebruiker echter nog niet klaar met haar zoekopdracht. Zij moet de relevante documenten vervolgens bekijken om er de juiste informatie uit te halen. Als de documenten lang zijn kan deze taak langdradig zijn. We stellen daarom de vraag: Kunnen we gefocuste toegang tot de relevante informatie geven?

In dit proefschrift bestuderen wij deze vraag in de context van semi-gestructureerde documenten—om precies te zijn, XML documenten. XML is een taal waarin de structuur van tekstuele documenten beschreven kan worden—bijvoorbeeld secties, sub-secties, paragrafen, titels, namen van auteur, cursieve tekst, enz. De XML-markup verdeelt de tekst in een hiërarchie van tekst-objecten—zogenaamde XML-elementen. In dit proefschrift stellen wij de vraag of we gebruik kunnen maken van deze hiërarchische structuur om voor gebruikers gefocuste toegang tot de relevante informatie te geven.

Onze methode bestaat uit twee delen. Eerst bouwen we een zoekmachine die XML-elementen weergeeft. Vervolgens bouwen we een interface die deze elementen gebruikt om gefocuste toegang tot de relevante informatie te geven. Wij evalueren de XML-zoekmachine met behulp van de zogenaamde INEX-testcollectie en we beoordelen de interface in een interactief experiment.

De belangrijkste bijdrage van dit proefschrift is een uitgebreide evaluatie van verschillende XML-zoekmethodes. Wij tonen tevens aan hoe de zoekmachine aan het werk kan worden gezet via een simpele interface.