DIR'06

Dutch-Belgian Information Retrieval Workshop

PROCEEDINGS

TNO ICT, Delft, The Netherlands
March 13-14, 2006

Franciska de Jong and Wessel Kraaij (eds.)

CIP GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Jong de, F.M.G., Kraaij, W.

DIR 2006
Proceedings of the Sixth Dutch-Belgian Workshop on Information Retrieval
F.M.G. de Jong, W. Kraaij (eds.) - Enschede: Neslia Paniculata.
ISBN-10: 90-75296-14-2
ISBN-13: 978-90-75296-14-3

trefwoorden: information retrieval

Preface

Welcome to the sixth Dutch-Belgian workshop on Information Retrieval. The primary aim of the DIR workshops is to provide an international meeting place where researchers from the domain of information retrieval and related disciplines can exchange information and present new research developments. This year there is a special focus on methods and tools tuned to domain-specific retrieval tasks.

ChengXiang Zhai of University of Illinois, Urbana-Champaign, has been invited to open the workshop with a presentation on Challenges in Applying IR Techniques to Bioinformatics. As the leader of both the IR and Bioinformatics research groups at UIUC, he has the ideal background for an introduction that jumps into the workshop focus: domain-specifc IR.

The second workshop day will be opened with an invited speech by Maarten de Rijke, University of Amsterdam. He will talk about new challenges for question-answering systems e.g. dealing with restrictions on the dataset or answer types.

The workshop programme was prepared by a programme committee consisting of Walter Daelemans (University of Antwerp), Alan Hanjalic (University of Delft), Franciska de Jong (University of Twente/TNO ICT, co-chair), Jaap Kamps (University of Amsterdam), Wessel Kraaij (TNO ICT, co-chair), Marie-Francine Moens (University of Leuven), Martijn Schuemie (Erasmus University Rotterdam) and Arjen de Vries (CWI Amsterdam).

The papers that will be presented at this workshop cover a wide variety of topics within the information retrieval domain, and reflect the different views and ongoing developments in the field. All papers selected have been reviewed by two members of the Review Committee consisting of the PC, plus Stephan Raaijmakers (TNO ICT) and Dolf Trieschnigg (University of Twente). We would like to thank them for their effort.

We are grateful for the sponsoring by the Dutch Working Community on Information Sciences (WGI), the Dutch Research School for Information and Knowledge Systems (SIKS), the Human Media Interaction group of the University of Twente, the Netherlands Bioinformatics Centre (NBIC), NWO Exacte Wetenschappen and the Taalunie-programme STEVIN.

Last but not least we would like to mention the role of Hendri Hondorp (Univerity of Twente /HMI) who maintained the workshop website and supported us in the production of the proceedings, and of the members of our local support team: Frank van Kesteren and Dolf Trieschnigg.

We hope that you will have a fruitful and enjoyable time at DIR2006!

Franciska de Jong and Wessel Kraaij — Delft, March 2006

Sponsors

- TNO Informatie- en Communicatietechnologie
- Dutch Working Community on Information Sciences (WGI)
- Human Media Interaction group, University of Twente
- Dutch Research School for Information and Knowledge Systems (SIKS)
- Netherlands Bioinformatics Centre (NBIC)
- NWO Exacte Wetenschappen
- Taalunie-programme STEVIN

Contents

Opportunities and Challenges in Applying IR Techniques to Bioinformatics

Keynote

ChengXiang Zhai
Department of Computer Science and Institute for Genomic Biology
University of Illinois at Urbana-Champaign
zchai@cs.uiuc.edu

ABSTRACT

Bioinformatics is an emerging interdisciplinary field that has been attracting much attention recently. Bioinformatics not only is a good application domain of IR techniques (due to the need for managing huge amounts of biomedical literature), but also uses many techniques commonly used in IR, such as search, clustering, categorization, and pattern finding, to process DNA/Protein sequences and other biological data. In this talk, I will broadly review some of the major information management problems in bioinformatics and identify opportunities for applying IR techniques to solve them. Many of these problems present unique challenges for IR and a straightforward application of standard IR techniques may not be effective. I will present some recent work on adapting standard information retrieval techniques to perform biomedical information retrieval and discuss how biomedical information retrieval can motivate some new research problems in IR.

Query Intention Acquisition: A Case Study on Automatically Inferring Structured Queries

Leif Azzopardi
Dept. of Computer and Information Sciences
University of Strathclyde, Glasgow G1 1XH
leif@cis.strath.ac.uk

Maarten de Rijke
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam
mdr@science.uva.nl

ABSTRACT

The process of acquiring the user's intentions is an important phase in the querying process. The identification of their intentions enables the selection of appropriate retrieval strategies. In this paper, we first outline this cross-section work in contextual Information Retrieval. We then focus on one particular type of intention: the syntactic and semantic types associated with a query term. We present a case study using the email search task of the TREC Enterprise Track. We build and analyze a data set of query intentions linked to the email's structure, and then attempt to automatically infer structured queries and study the affect that ambiguity of queries and the difficulty of inferring them has on various retrieval models (structured and unstructured). Our study reveals that predicting the intentions is a hard problem due to the inherent uncertainty within the querying process. We also show that automatically inferred queries do not outperform other types of structured retrieval models, because they are not robust enough to handle the ambiguity nor reliable enough to be accurately inferred.

1. INTRODUCTION

It has been a long recognized problem that the query submitted to an information retrieval (IR) system is a sparse and impoverished representation and expression of a user's actual information need [17]. The problem stems from the series of translations that are undergone when an information need first arises and then is surmised as a two or three keyword query. Much of the meaning or intent of the user's information need is lost. For example, a user wanting the homepage of Vandalay Industries, may submit the query, 'Vandalay Industries' or 'Vandalay Industries homepage'. In the first case, any indication of the user wanting a specific home page is lost. Whilst in the second this intention is present, but will usually be treated as another keyword in the query.

A contextual IR system will attempt to develop a better understanding of the user's underlying information need through what we refer to as, *query intention acquisition* (QIA): the process of acquiring a query and analyzing the query to extract the meaning, semantics and nature of the query. The goal of QIA is to find out what intentions the user has, why they formulated that query, and why they chose those query terms. This represents a shift away from the uniform treatment of queries to approaches that utilize the structure within and of queries in order to capture more of the user's intentions (i.e., a move from key word based queries to more structured queries). This should lead to a better understanding of a user's actual information needs by the system, which if utilized effectively can facilitate improvements in retrieval effectiveness [12]. Thus, QIA is an important component of any contextual IR system.

QIA can be performed either in an explicit questioning of the user by providing them with the capability to more adequately describe their information need [12], or implicitly through inferences and assumptions based on query characteristics [5, 9, 4]. Explicit capturing of the query information can be achieved either through using a formal query language for expressing queries or through application specific interfaces (see Figure 1 for examples). In the former, the user is able to express a more precise information need by constructing a query using a given language (such as Boolean Expressions, XPath, XQuery, SQL, etc). While for the interface approach the same is achieved by having the user populate fields, select tabs, select from drop downs, and so on, to apply filters and constraints and so forth to the search. Both are intuitively appealing for QIA, all be it for different reasons, expressiveness and ease of use, respectively. There *appears* to be a trade off between the two. E.g., a highly expressive language would decrease the ease of use because more training and effort is required in issuing an explicitly structured query. Whilst for an interface to provide more expressiveness it becomes more cumbersome. Further, both of these approaches suffer from two major drawbacks, complexity and lack of usage.

The added complexity involved in creating a structured query is time consuming. This is a significant problem, which in the case of formal languages is compounded by the requirement of formulating valid and syntactically correct queries. At INEX,[1] structured XPath queries proved difficult even for competent users[2] to formulate and construct [14]. Even simple query constructions like Boolean Expressions are error prone and infrequently used despite

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

[1]INEX: Initiative for the Evaluation of XML retrieval, see http://index.is.informatik.uni-duisburg.de/
[2]Researchers from computer science and related disciplines.

Information Need:
Retrieve the email that Makici wrote in October to Maillie that was titled, Multimedia.

Query:
Multimedia in October by Makici

INEX XPath Query:

```
//Email[ subject = "Multimedia" and from = "Makici"
          and date="October"]
```

Interface:

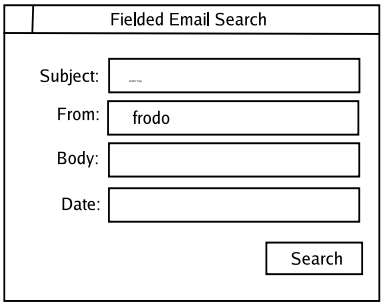

Figure 1: Examples of different ways of expressing an information need.

wide-spread implementation. In terms of usage, when advanced search functionality is provided in digital libraries [9] and on web search engines [3], they are very rarely used. Presumably, this is because the increased time and complexity rarely produces better retrieval performance. The burden and expense in submitting a more precise or better expressed query needs to be mitigated by techniques which are more ubiquitous in the acquisition of query intentions. So instead of performing QIA in such an explicit manner there has been a move towards developing techniques that automatically or implicitly attempt to infer the intents—and this is where this paper's contribution lies. We present a case study on automatically inferring structured queries, thus identifying the semantic types associated with query terms.

The remainder of the paper is organized as follows. In the next section, we outline different types of intentions to provide some background. Section 3 describes the specific query intentions we aim to study and what work has already been done in this area. We then present our case study in the remaining sections: we examine the influence of structure and intentions on the effectiveness of different types of structured retrieval models. We conclude with a discussion of our key findings in Section 7.

2. TYPES OF INTENTIONS

Consider the information needs and queries in Table 1. Each conveys different types of intentions that a user has about their information need. The intentions behind a query vary depending on the information need and stipulate the conditions of relevance. The types of intentions may include, but are not limited to: what type of document format the user wants, what unit of retrieval is sufficient, the type of search required, the pitch of the information desired, and the meaning of a query term in the query. We briefly discuss each of these in turn.

Information Need	Query
I want the homepage of Vandalay Industries	Vandalay Industries homepage
I want to know the syntax for a for loop in c++	for loop in c++
Retrieve the email that Makici wrote in October to Maillie about Multimedia	Multimedia in October by Makici
Find me images of Britney Spears in a School Uniform	Britney Spears jpeg

Table 1: Example Information Needs and Queries.

- **Document type**: Given the variety of documents that are available, users may occasionally be interested in only particular types of document. Common document types often sought after in web, desktop and enterprise search include, emails, minutes, reports particular document formats like PDF, HTML and Microsoft Word file, images, movies, and so forth.
- **Unit of retrieval**: This is related to the document type, but is concerned with the part of a document that should be retrieved. For instance, a user may only want snippets, summaries, sections, passages of documents, citations, the full text, etc. Such intents have been considered as part of INEX.
- **Search task**: The search type (ad hoc, known item, etc) query prediction for retrieval strategy selection; i.e., a web searcher submits a query, where they may be looking for a home page, or many pages relating to a topic. Classifying queries by such intentions has been considered in the late Web track at TREC and more recently in [4].
- **User expertise**: The identification of the user's level of expertise has some influence on what is considered relevant by the user. Identifying and using this information is considered in the HARD track at TREC.
- **The semantics and syntax of the query and query terms**. Examination of the query constituents may provide evidence to suggest many of the above types of intentions. However, it is usually concerned with detecting syntactic and semantic knowledge, such as noun-phrases, term dependencies, what field a query terms refers to, which is used by the retrieval model.

It has been generally posited that knowledge of the above factors can be used to increase the effectiveness of a retrieval system, because the retrieval strategy can be tailored specifically to the retrieval scenario. This area represents a wide cross-section of research performed in contextual Information Retrieval. For the case study we shall examine only the latter type of intention acquisition.

3. INFERRING QUERY TERM INTENTS

There are two main types of intents that are often implicitly inferred by an IR system: ones related to *syntactic* features (like term dependencies), and ones related to *semantic* features (like what field a query term refers to).

One of the initial attempts in structuring queries was by Croft *et al.* [6]. They consider structure to be phrases within

the document. The natural language queries are taken and converted to boolean queries incorporating the extracted phrasal information. They automatically identify phrases by employing three different methods; using a parser based primarily on phrase syntax; a stochastic approach using part of speech information; and, a dictionary of phrases. They show that the retrieval performance using structured queries is more effective than unstructured queries. Further, they showed that automatically structured queries were as effective as structured queries [6].

Several Language Modeling approaches have been proposed over recent years that attempt to exploit dependencies between terms [15, 16, 8]. These focus on the syntactical relationships between query terms defined by co-occurrence data from the collection. Such relationships are usually determined by finding a Maximum Spanning Tree of the query term dependencies and assuming that these terms were dependent accordingly. The probability of producing the query with the specific syntactic structure is used to rank the documents. Such methods have also provided increases in retrieval performance.

The work most relevant to our case study attempts to infer the semantic structure implied by query terms to formulate a structured query automatically. In [5], a set of possible structured queries are inferred from the original natural language query. Their approach assigns the query term to the most likely field. Then a set of structured queries is generated, where the best structured query is the one that maximizes the likelihood of the (query term, field) pairs which is assumed to corresponds to the user's intention. Goncalves *et al.* [9] investigate the effectiveness of automatically structured queries within the context of Digital Libraries containing journal articles. They follow a similar query structuring procedure as in [5] with the constraint of assigning a query term to one and only one field. The selection of the best structured query was vital to their method. Their results show that retrieval effectiveness was as good as or better than a flat query baseline for the majority of queries. However, the comparisons made in these studies do not consider any stronger baselines which utilize document structure in other ways. Nor do they consider what factors influence the structuring and retrieval. In our case study, we investigate different types of structured retrieval models (including ones similar to those used above) and determine how they perform against and with automatically structured queries of varying quality.

Despite the prevalence of structured documents available to users, there has been little other work investigating the benefits and impact that structure plays in the querying process and whether this can be reliably inferred and used effectively. One of the major barriers to such research is that there are no data sets available. In our case study, we build such a data set.

3.1 Benefits

There is a number of reasons why we would like to be able to automatically infer structured queries in a ubiquitous and seamless manner. These include:

- The simplification of the querying interface for the user. There is no requirement for the user to have to use a complicated search interface with multiple boxes or a complicated formal language model expressing structure.
- The formulation of structured queries is often an arduous task—there is limited use of advanced search facilities of my search engines [3].
- The ability to illicit a better understanding of what the user is searching for, so that the retrieval strategy can be tailored to the user's information need.
- This is a move towards bridging the semantic gap between the intent or meaning of the query terms and the information need and by using this knowledge, performance increases could be obtained.

3.2 Evaluation

Invariably, any system that attempts to predict the intent(s) of query terms will need to undergo a more detailed evaluation before being deployed in an operation setting. Specifically, we consider the following criteria:

1. **Reliability:** How accurately can we infer user's intentions from the query?
2. **Robustness:** How robust is the retrieval method with respect to incorrect inferences?
3. **Retrieval Effectiveness:** How effective are the automatically structured queries in terms of retrieval performance?

While other criteria (such as *timeliness*: how quickly can we infer the user's intentions from the query?) are important as well, the three listed above are more fundamental.

4. CASE STUDY: ENTERPRISE TRACK

In our case study we attempt to infer query term intentions in an email forum and consider the difficulty in predicting the intentions of query terms, their ambiguity and the influence this has on different structured retrieval models.

To examine the phenomena of inferring query semantics we have chosen a recent TREC collection, the W3C Public Email Forum from the 2005 Enterprise Track and the task of known-item email searching. We have chosen this collection and task for several reasons. The Email Forum provides a collection which has structure present within the email document (i.e., *subject*, *from*, *to*, etc), which is of a semantic nature and so is suitable for our study. The task is a common search task and the collection provides over 150 example queries from which to build a data set[3] of query intentions. Also, the task is to find a specific email, so reconciling the query terms to the email fields will be possible.

With the collection and task chosen, the case study was broken into the following four steps: (1) The building the data set of query intentions (Section 4.1), (2) An analysis of the query intentions data set (Section 4.2), (3) Automatically structuring queries and classification of query intentions (Section 5), and (4) A study of the influence of ambiguity and the difficulty inferring queries has on various retrieval models, structured and unstructured (Section 6).

4.1 Building a Query Intentions Data Set

The email sub-collection in the W3C corpus (called "lists") contains approximately 170,000 emails posted to the W3C

[3]This data set of query intentions will be in XML and made available from the first author's website.

forums over several years; other non-email documents (such as administrative and navigational pages) in the lists collection were excluded which amounted to the removal of about 30,000 documents. The TREC topics KI1-25 and KI26-KI50 were concatenated to form 150 known item queries.

Each query term for a particular query was manually tagged with the fields in the known email from which the query originates. The possible tags for email fields were: *date*, *from/author*, *subject* and *body*, all other fields were ignored. The assignment of a query term to the email's field was done according to which field was the most salient. By saliency, we refer to how obvious and memorable that field is in an email. The order used was date, author, subject and body, unless surrounding terms indicated otherwise. For instance, if the term 'June' was present in both the date of the email and the author of the email, then it was assigned to the date. Unless some other evidence such as a surname was present or there was an indicator like 'by June' as opposed to 'in June'. If the query term did not occur in the email then it was classed as being "about" one of the possible fields. For instance, referring to persons using a nick name (as shown in Query KI44 in Figure 2, where 'James' is used instead of 'Jim').

Stop words were ignored except for those that seem to indicate topicality ('on', 'in', 'for'), date ('on', 'in'), authorship ('by', 'from'), format of email ('minutes', 'call for papers'); those were tagged as `<T>`, `<A>`, `<B>`, and `<C>`, respectively. Non-text indicators, such as apostrophes, dashes, slashes, and commas were also tagged (`<D>`).

Whilst we have assumed that the query terms have come from a specific field in the email message, this is not necessarily the case in reality. A very frequent term in the email may be in both the body and the subject and chosen by the user because of its overall popularity (or recall-ability) within the email. However, we feel that assigning to the most salient feature of the email is a reasonable approximation under a hard classification. In practice, such hard assignments may not necessarily be employed—this all depends on the retrieval method.

In Figure 2, query KI11 consists of keywords that occur in the fields and their meaning is unambiguous with respect to the known-item email. As there are no other features in the query or extraneous terms there is little loss, except that there is no explicit marking of what fields the query terms refer to. However, the term 'minutes' might possibly suggest an email formatted in such a style or an attachment. The second query KI44 in Figure 2, though, looses several subtle intents in the query through the parsing process. The apostrophe, indicating who wrote the email, the type of email, a question to the forum, the reference to the topic. *Tomcat* with the use of 'in' and indicating what the email is concerned with by the use of 'about' perhaps to denote some vagueness in the description. Further, the query term "Tomcat" is what the known item is about but there is no actual mention of "Tomcat."

Query: KI11
Text: tag minutes 9 june 2003
Parsed:tag minutes 9 june 2003
Marked up in xml:

```
<subject>tag</subject>
<subject><C>minutes</C></subject>
<date>9</date>
<date>june</date>
<date>2003</date>
```

Query: KI44
Text: James' question about the Webdav in Tomcat
Parsed: james question about webdav tomcat
Marked up in xml:

```
<author type="about">James<B>'</B></author>
<body type="about"><C>question</C</body>
<body type="about">about</body>
the
<body>Webdav</body>
<T>in</T>
<body type="about">Tomcat</body>
```

Figure 2: Examples of different ways of expressing an information need.

Table 2: Statistics on the main tags in the query set.

Field	Total Count	Total Queries
Date	19	13
From	41	24
From About	2	1
Subject	323	111
Subject About	28	22
Body	160	62
Body About	61	34
Indicators	105	76
Non word Indicators	28	25

4.2 Query Characteristics

The number of times each field occurred and in how many queries is given in Table 2. Interestingly, the majority of the queries (111 out of 150) contained query terms relating to subject, and over half had some form of indicator, whilst the other features occurred somewhat less often. In the queries there were no references to who the email was to. There were 91 instances where query terms were "about" a field in an email, which indicates that a considerable amount of noise is present within the queries. We refer to this noise as *ambiguity* as the query being expressed to the system contains uncertain information with respect to the target email. We classified queries according to how much ambiguity was present, using three grades: (0) not, (1) somewhat, or (3) very ambiguous. If all the features occurred in the known-item, then there is little or no ambiguity (i.e., we assumed that the query term was put there with specific reference to some field in the email). However, if more than half the query features are present in the known-item email, then there is some ambiguity in the query. If the majority of query terms do not occur in the email then the query is very ambiguous. To some extent this measure reflects the loss of recall that is experienced by a user when formulating the query; assuming that they are trying to select (remember) the exact words and phrases from the email they have in mind. The more vague the user is about their missing email, presumably the less precise and more ambiguous the query will be as a result.

In this collection of emails, we found that there were 20 very ambiguous queries, 33 somewhat ambiguous queries, and the rest were judged as not ambiguous (97).

Intuitively, we would expect that the less ambiguous a query is, the higher the retrieval performance should be as exact matching techniques will have more accurate information for ranking. For more ambiguous queries, then, the classification of such terms will degrade the accuracy in obtaining the correct intent of the query term. The incorrect structuring of a query could then lead to a serious degradation in retrieval performance, if the retrieval method is not robust enough to handle such ambiguity.

5. INFERRING QUERY STRUCTURE

To automatically create structured queries from unstructured queries we used generative language modeling techniques and decision theory to classify each query term with respect to the fields in the email. This combines and formalizes some of the existing work in a more general framework which can be applied to any type of data collection, independent of the retrieval model.

Within a collection of structured documents, let document d be a structured document which is composed of a set of components $x \in X$. The fields may be any feature (semantic, syntactic, layout) which has been indexed as part of the document representation. We assume that each field is a bag of terms and can be defined as a probability distribution over the vocabulary, such that the probability of a term given a field and document is $p(t|x,d)$. Taking the maximum likelihood estimate:

$$p(t|x,d) = \frac{n(t,x,d)}{\sum_{t',x'} n(t',x',d)},$$

where $n(t,x,d)$ is the number of times the term occurs in the field x of document d. By marginalizing over all documents in the collection, the probability of a term given a field $p(t|x)$ is obtained. This serves as a model of the terms that we expect to be generated from that field.

Now, given an unstructured query q which consists of a series of query terms $\{q_1, \ldots, q_k\}$, the aim is to assign each query term to the corresponding fields within documents and thus form a structured query. A structured query is defined in a manner similar to structured documents. The structured query q^s is a set of sets of query terms q^s_x, one set for each query field $x \in X$. For instance, given the email example where X = {subject, from, to, date, body}, the query q = {'Multimedia', 'Bark', 'Maillie', 'Yurat', 'Makici'} is transformed into the structured query $q^s = \{q^s_{subject} =$ {'Multimedia'}, q^s_{from} = {'Bark', 'Maillie'}, q^s_{to} = {'Yurat', 'Makici'}, $q^s_{date} = \{\}$, $q^s_{body} = \{\}$ $\}$.

5.1 Classification

To classify a given query term, we employ the odds ratio [7] to decide whether the query term q_i was from the field x or not, i.e., $\bar{x}$. Formally, we express this as:

$$O(x,q_i) = \frac{p(x|q_i)}{p(\bar{x}|q_i)},$$

where $p(\bar{x}|q_i) = 1 - p(x|q_i)$.

We wish to determine which field each of the query terms belongs to or is associated with. That is, we wish to infer the structure of the query. We consider the problem from two angles, one where each query term is treated independently and one where we treat the query as a sequence of terms where the dependence between terms is considered.

Independence Model. Here, the query terms are assumed to be independent of each other. We wish to determine the probability of the component (or class) given the query term q_i, i.e., $p(x|q_i)$ for each component x.

This can be estimated by invoking Bayes' theorem:

$$p(x|q_i) = \frac{p(q_i|x)p(x)}{p(q_i)},$$

where $p(q_i) = \sum_x p(q_i|x)p(x)$.

Dependence Model. Here, each query term is dependent of the preceding query term (strict and limited). We wish to determine the probability of the element (or class) given the query terms q_i and q_{i+1}, i.e., $p(e|q_i, q_{i+1})$ for each element e.

This can, again, be estimated by invoking Bayes' theorem:

$$p(x|q_{i+1}, q_i) = \frac{p(q_{i+1}, q_i|x)p(x)}{p(q_{i+1}, q_i)}.$$

Applying the chain rule to $p(q_{i+1}, q_i|x)$ gives

$$p(x|q_{i+1}, q_i) = \frac{p(q_{i+1}|q_i, x)p(q_i|x)p(x)}{p(q_{i+1}, q_i)}$$

and

$$p(x|q_{i+1}, q_i) = \frac{p(q_{i+1}|q_i, x)p(q_i|x)p(x)}{\sum_{x' \in X} p(q_{i+1}|q_i, x')p(q_i|x')p(x')}.$$

Since $p(q_{i+1}|q_i, x)$ is very sparse, we opted to weaken the strict order dependence and compute $p(q_{i+1}|q_i, x)$ proportional to the number of times the terms co-occurs in a window of size of two. (See [2] for further details on computing the conditional probabilities.)

Assigning Query Terms to Fields. Finally, structured queries were created by using two strategies: *strict*—where the query term was assigned to the field x, which maximized $O(x, q_i)$—, and *fuzzy*—where the query term was assigned to the fields, where $O(x, q_i)$ for the field x is greater than some threshold ρ. The threshold enables the assignment of a term to multiple elements.

5.2 Classification Results

The inferred semantic type for each query term was compared to the set of 'ground truth' tags from the manual classification process. After parsing the queries the total number of tagged terms was 685. The break down of each class was: date 19, from 42, subject 323, body 145 and unknown 156. The unknowns immediately resulted in classification failure and so are not reported. However, this meant that 22.8% of the query terms were essentially noise in the query. The classification accuracy performance is reported in Table 3 for statistics on a class by class basis. As a baseline, we assumed a naive model that assigns query terms to the most probable class (i.e., subject). The overall classification accuracy for the baseline was 61.1%. The independence and dependence models performed only marginally better, achieving only 62.0% and 62.4 %, respectively.

We further examined each query and classified the query with respect to the difficulty in predicting correctly the query

Table 3: Classification Accuracy shown as a percentage (%) correct per class on the Independence and Dependence Classification models.

	Classified as :				
	Date	**From**	**Subject**	**Body**	Out of:
Date	**84.2** (**84.2**)	5.3 (5.3)	0.0 (0.0)	10.5 (10.5)	19
From	11.9 (9.5)	**78.6** (**78.6**)	0.0 (2.4)	9.5 (9.5)	42
Subject	2.8 (2.5)	2.5 (2.8)	**72.4** (**73.7**)	22.3 (21.1)	323
Body	0.7 (0.7)	2.1 (2.1)	66.2 (67.6)	**31.0** (**29.7**)	145
Total					529

term intents. Three groups (*easy*, *medium* and *hard*) were obtained by assigning all queries that had at most one incorrectly labeled query term as easy. Queries which had at most half their terms labeled incorrectly were classified as medium, and all other queries were classified as hard. This resulted in: 47 easy, 37 medium, and 65 hard queries.

When we compared the difficulty of inferring the structure of a query against the ambiguity of a query, using a χ-squared test we found that the two groups were actually independent ($p < 0.0001$). This appears to be because the number of unambiguous queries were often difficult to predict (either medium or hard). Hence, these are two independent factors which could impact retrieval performance. Note, that the difficulty is (potentially) only a problem when using inferred structured queries—and not for other retrieval models. However, the difficulty could still be indicative of the performance, regardless of the type of retrieval model.

6. LANGUAGE MODELS AND STRUCTURE

Generative Language Models have been applied successfully in a number of tasks in IR, including structured retrieval [10]. They provide several related models that incorporate structure in various ways. Since the models are related the differences are clear from their formulation and enable a fair discussion and comparison over different experimental factors. Below we give an overview of three different Language Modeling approaches. The first is the standard query likelihood approach to retrieval which does not make any structural assumptions about the query or documents [10]. The latter models incorporate the structure of the query in the ranking of documents in distinctly different ways [13, 11, 1]. The first relies on a combination of evidence to produce a better document model and the second form structured queries with which to query the structured emails.

The *Standard Language Modeling approach* computes the probability of a query q being generated from a document model θ_d on behalf of the document d as follows:

$$p(q|\theta_d) = \prod_{t \in q} \left\{ (1-\lambda) p(t|d) + \lambda p(t) \right\}^{n(t,q)}, \qquad (1)$$

where $p(t|d)$ is the maximum likelihood estimate of term t in document d, $p(t)$ is the unconditional probability of t (also using the maximum likelihood estimate), $n(t,q)$ is the number of times term t occurs in query q, and λ is the smoothing parameter.

The *Combination Language Modeling approach* is an extension of the standard approach [13, 11]. It combines the different fields of a document to form one smoothed document model. The document model becomes a combination over each field within the document, and then the document model is further smoothed by the background collection model:

$$p(q|\theta_d) = \prod_{t \in q} \left((1-\lambda) \sum_{x \in X} \left\{ p(t|x,d) p(x|d) \right\} + \lambda p(t) \right)^{n(t,q)}$$

Each field in the document is weighted by $p(x|d)$, which can be interpreted as an indicator of the importance of that field in representing the document. This model has been highly successful for structured retrieval, despite only using structure on the document side.

The *Fielded Language Modeling approach* is the general solution where the joint probability of the components given the structured document needs to be estimated and a structured query [1]. The simplest approach is to assume that each field is independent of the other. However, depending on the structured document and the task, this assumption may be relaxed to account for the relationship between fields. For example, given an email, the subject is dependent on the body, the body dependent on the author and so forth. In reality it may be infeasible to compute such dependencies between the fields; thus we must resort to the independent Fielded Language Model. It is a simple extension of the standard Language Modeling approach as it treats each field of the email document as an independent source of evidence. From each field, query terms are drawn which generated the structured query. Formally, this can be represented as:

$$\begin{aligned} p(q|d) &= p(q_{x_1}, \ldots, p_{x_n} | \theta_d^x) \\ &= \prod_x p(q_x | \theta_d^x), \end{aligned}$$

where $p(q_x|\theta_d^x)$ is the probability of the query field q_x being generated from the model of the document field θ_d^x. This probability is computed as above for standard documents, but for each of the four fields instead.

Each model utilizes structure differently. Whilst the standard model ignores structure all together the others use structure in different ways. The combination LM ignores any structure in the query and focuses on building a better document representation by marginalizing over all the fields in the document to form a robust statistical estimate of that term occurring in the document. The Fielded LM treats each field independently which provides a natural mechanism for issuing structured queries which are matched against the corresponding fields.

This provides three distinct approaches for dealing with and using structure in the IR process and should enable us to study the impact of ambiguity and difficulty given these retrieval models.

6.1 Experiments

The three Language Models were configured as follows: The Standard LM with the smoothing operator λ set to 0.1

Table 4: The retrieval performance of each of the different retrieval models.

	LM	Setting	MRR					
			Overall	Ambiguity		Difficulty		
				None	Some/Very	Easy	Medium	Hard
a	**Standard**		0.466d	0.537	0.337	0.501	0.409	0.470
b	**Combination**	**uniform**	**0.631**ade	**0.719**	**0.469**	0.701	0.636	**0.578**
c		**automatic**	0.627ade	**0.719**	0.458	**0.705**	**0.638**	0.565
d		**strict**	0.355	0.436	0.208	0.516	0.455	0.186
e	**Fielded**	**fuzzy**	0.546ad	0.667	0.325	0.693	0.574	0.425
f		**explicit**	0.581ad	0.679	0.400	0.687	0.425	0.479

as the baseline model.[4] All other models used the same lambda to try and ensure a fair comparison amongst the different language models. The Combination LM was set to *uniform* or *automatic*. *Uniform* refers to when the prior probability of all fields is uniform i.e., $p(x|d) = 0.25$, and *automatic* refers to when the prior probability of a field is set with respect to the number of the query terms that were classified as a particular field $n(t, x)$, i.e., $p(x|d) = n(t, x)/\sum_{x'} n(t, x')$. This latter assignment uses the inferred intents when estimating each fields importance with the document. The Fielded LM was set to *strict*, *fuzzy* or *explicit*, which refer to the query intent information that was provided. *Strict* and *fuzzy* are with respect to the automatically inferred queries, where *strict* assigns one field label, whereas *fuzzy* assigns multiple field labels (with $\rho = 0.1$). The *explicit* setting refers to when the actual query tags are used (i.e., fully explicit labeling of the query terms and their intents).

6.2 Retrieval Results

In Table 4 we show the retrieval performance for each of the different retrieval models in terms of the Mean Reciprocal Rank (MRR), including a breakdown in performance over Ambiguity and Difficulty. Note that the ambiguity scale was reduced to two levels since the number of queries which were somewhat and very ambiguous were few and so were combined. We compared each model's performance using the Wilcoxon Signed Rank Test ($\alpha = 0.05$). In Table 4, if a model significantly outperformed another then the letter identifying the model being outperformed (in the left-most column) is shown to denote this. We found that the worst performing model (significantly so against all other models) was the strict Fielded LM. On the other hand, the combination LM approaches were significantly better than all other models, except when explicit information was used in the Fielded LM. The fuzzy Fielded LM significantly outperformed the standard model and was on par with the explicit Fielded LM.

7. DISCUSSION AND CONCLUSION

Through the course of our study we have examined two factors related to query intention which impact retrieval performance: *ambiguity* and *difficulty*. Our findings confirm the hypothesis that if the level of ambiguity in a query increases then the retrieval performance will degrade. The magnitude of degradation remains relatively consistent across the different retrieval models, on average 0.26 less in MRR.

[4]Other λ parameters were also tried but the results were similar to those reported here.

In general, our findings also show that as it becomes more difficult to infer the query structure, the retrieval performance degrades. However, there are a few notable exceptions. The standard model appears to be relatively robust to how hard it was to predict structure. Presumably, this is because the standard model does not make use of such information and so can not be affected by any misclassifications. When we compare the differences of the Combination LM against the Fielded LMs we also notice that, because the Fielded LMs rely upon structured queries, retrieval performance suffers more so than for the Combination LM. The most pronounced example of this is when the strict Fielded LM is employed, and there is large drop in MRR when more than half of the query terms were incorrectly classified.

The retrieval results confirm previous findings from [5, 9] that automatically structured queries outperform unstructured queries (Standard LM vs Field LM (fuzzy)). However, our message is more subtle. Compared against the standard model, the combination LMs have the advantage that they account for the structure present in the document by averaging over each field. Compared against the Fielded LMs the structured document model again wins out, for the same reason. Structure is accounted for within the document model of the combination LM, so there is no reliance on query side inferences, and hene the prediction difficulty is not a factor.

Our results show how difficult it is to ascertain the user's actual intent and then how to make good use of these intents. More research is needed to develop retrieval techniques that can handle structured queries which also improve performance. Whenever there is ambiguity and/or difficulty within the queries, this uncertainty needs to be accounted for by the retrieval model.

Further, our results suggest that email search facilities provided with email clients need not be field based, like the example in Figure 1 but could be simplified by employing the combination LM approach.

Our study shows that there are predictable habits within user querying behavior but more research needs to be performed in this area. Whilst the brute force approach would be to develop better classification methods so as to improve the structured queries produced, an alternative approach would be aimed at developing a natural language based querying language, which provides natural cues for the prediction of query terms and their intents. In a domain such as known email searching, this appears to be quite feasible. In some of our examples certain 'stop words' appear indicative of intent, such as 'by' indicating who wrote the email, 'in' or 'on' indicating the date. By using a subset of stop words, more infer-able queries could be submitted to the retrieval

system without significantly increasing the burden to the user because it is simply natural language (i.e., 'Multimedia by Mallie from Makici').

In conclusion, QIA represents an important process in any contextual IR system. However, our study has shown that acquiring the intents of query terms with respect structure is a non-trivial task which requires further research to fully develop and utilize such information. Perhaps controversially, our study has shown that whilst automatically structured queries outperform baseline models and are as good as explicitly structured queries, more sophisticated retrieval models still fair better, again suggesting that more research is required to develop models that can handle structured queries and still provide robust and superior retrieval performance.

Now that a data set has been created, future work can be directed in a number of areas, such as improving the classification accuracy using other techniques, estimates and indicators, improving retrieval effectiveness by considering and developing more robust structured retrieval strategies.

Acknowledgments

Maarten de Rijke was supported by grants from the Netherlands Organization for Scientific Research (NWO) under project numbers 017.001.190, 220-80-001, 264-70-050, 354-20-005, 612-13-001, 612.000.106, 612.000.207, 612.066.302, 612.069.006, 640.001.501, and 640.002.501.

8. REFERENCES

[1] L. Azzopardi, K. Balog, and M. de Rijke. Language Modeling Approaches for Enterprise Tasks. In *Proceedings of the 14th TExt Retrieval Conference (TREC 2005)*, 2006.

[2] L. Azzopardi, M. Girolami, and M. Crowe. Probabilistic hyperspace analogue to language. In *Proceedings of the 28th Annual International ACM SIGIR conference on Research and development in information retrieval*, pages 575–576, New York, NY, USA, 2005. ACM Press.

[3] R. Baeza-Yates. Query usage mining in search engines. In A. Scime, editor, *Web Mining Applications and Techniques*. Idea group, 2004.

[4] M. Bomhoff, T. Huibers, and P. van der Vet. User intentions in information retrieval. In *DIR '05: Proceedings of the 5th Dutch Beligian Workshop in information retrieval*, pages 47–54, 2005.

[5] P. Calado, A. S. da Silva, R. C. Vieira, A. H. F. Laender, and B. A. Ribeiro-Neto. Searching web databases by structuring keyword-based queries. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 26–33, New York, NY, USA, 2002. ACM Press.

[6] W. B. Croft, H. R. Turtle, and D. D. Lewis. The use of phrases and structured queries in information retrieval. In *SIGIR '91: Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 32–45, New York, NY, USA, 1991. ACM Press.

[7] M. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill, New York, 1970.

[8] J. Gao, J. Y. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177. ACM Press, 2004.

[9] M. A. Goncalves, E. A. Fox, A. Krowne, P. Calado, A. H. F. Laender, A. S. da Silva, and B. Ribeiro-Neto. The effectiveness of automatically structured queries in digital libraries. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pages 98–107, New York, NY, USA, 2004. ACM Press.

[10] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, Enschede, 2001.

[11] J. Kamps, G. Mishne, and M. de Rijke. Language models for searching in web corpora. In *The Thirteenth Text Retrieval Conference (TREC 2004)*, volume SP 500-261 of *NIST Special Publication*, 2005.

[12] D. Kelly, V. D. Dollu, and X. Fu. The loquacious user: a document-independent source of terms for query expansion. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 457–464. ACM Press, 2005.

[13] P. Ogilvie and J. Callan. Combining document representations for known-search. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 143–150, New York, NY, USA, 2003. ACM Press.

[14] R. A. O'Keefe and A. Trotman. The simplest query language that could possibly work. In *Proceedings of the 2nd INEX Workshop*, 2004.

[15] F. Song and W. B. Croft. A general language model for information retrieval. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 279–280. ACM Press, 1999.

[16] M. Srikanth and R. Srihari. Biterm language models for document retrieval. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–426. ACM Press, 2002.

[17] R. S. Taylor. *Value-added Processes in Information Systems*. Ablex Publishing, Norwood, NJ, 1968.

Vague Element Selection and Query Rewriting for XML Retrieval

Vojkan Mihajlović Djoerd Hiemstra Henk Ernst Blok
CTIT, University of Twente
P.O. Box 217, 7500AE Enschede, The Netherlands
{v.mihajlovic, d.hiemstra, h.e.blok}@utwente.nl

ABSTRACT

In this paper we present the extension of our prototype three-level database system (TIJAH) developed for structured information retrieval. The extension is aimed at modeling vague search on XML elements. All three levels (conceptual, logical, and physical) of the TIJAH system are enhanced to support vague search concepts. The vague search is implemented as vague selection of XML elements using XML element name expansion lists and rewriting techniques. We test the performance of retrieval models using automatically generated expansion lists and compared them with models that use manual ones. The goal is to find the best approach for structured information retrieval with vague structural constraints on element names expressed in the query.

1. INTRODUCTION

Due to the fact that more and more documents on the web come in structured formats such as XML, the information retrieval community begins to realize the importance of document component retrieval and structured querying. To exploit structured retrieval, information retrieval query languages are enriched with the ability to state structural constraints in the query (e.g., [1, 4, 5, 19]). The question is how these constraints should be treated when answering a query: as strict constraints that must be satisfied or just as users' suggestions on where to search for information.

Similarly, as the user gives only a number of terms as hints for searching within a document, XML elements specified within the query need not be considered as a strict requirement but as a hint for structural search. Therefore, when formulating a query the user can state that the search element (support element) or answer element (target element) should be treated as a hint or as a constraint in the retrieval process. We use the term *search element* for elements in which the user would like to find some information and the *target element* for elements that the user would like to see as an answer to his query.

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

The motivation for vague search is found in the discussion raised during the 2004 INitiative for Evaluation of XML retrieval (INEX) workshop and the tasks that are set for the year 2005[1]. The INEX 2005 ad-hoc track introduced a number of subtasks for both the content-and-structure (CAS) task and the content-only (CO) task. For the CAS task, the goal was to test whether the structural constraints should be followed strictly or not, and if not, to what degree they should be freely interpreted. The idea is that the structural constraints should be considered as hints, and different degrees of vagueness for structural constraints should be tested by the following four scenarios:

- SSCAS that assumes strict matching between the structural conditions stated in the query and the path leading to the search and answer elements.
- SVCAS where the structural conditions of search elements need not to be strictly satisfied.
- VSCAS in which the answer elements can be interpreted vaguely, i.e., the answer element need not to be the one specified in the query.
- VVCAS where all the structural constraints can be interpreted vaguely.

On the other hand, for the CO task we address the subtask termed as content-only plus structure (CO+S or COS), formed by adding structural constraints to a set of terms in CO query. Here, structural constraints are also interpreted vaguely, but without explicit separation among different sub-tasks. These queries are used to check whether the structural information can help in the searching process and in what way.

To support these different types of vague search scenarios expressed in user queries:

- we introduced vague element search as a concept
- we allow terms to cross the structural boundaries stated in the query

Vague element search (selection) can be treated in a similar way as query expansion on terms in traditional IR. For example, if a user searches for the term 'conclusion', he might also be satisfied with terms 'decision', 'determination', 'termination', or 'ending' in the answer. In structured documents, if a user asks for 'car' elements, he would probably

[1] http://inex.is.informatik.uni-duisburg.de:2005/.

not mind getting 'auto' or 'vehicle' elements as an answer. Furthermore, he might also agree with the answers: 'van', 'sports-car', or 'convertible'.

While the list of possible synonyms, hypernyms, and hyponyms for terms can be considered as relatively static over time (e.g., WordNet [15]) and the degree of similarity can be pre-specified, in the case of *element name expansion* the problem is more complex and dynamic. Besides the terms that have the same or similar meaning, like the ones given above, it can happen that element names follow different naming patterns. Thus, elements might have complex element names such as: 'sports_car', 'vehicles_list', etc.. Abbreviations could also be used, such as for section elements in the INEX IEEE collection [10]: 'sec', 'ss1', 'ss2', 'ss3'.

Additionally, if a user asks for elements denoting one concept it might not be wrong if the answer is an element from a similar concept. Plenty of such examples can be identified in INEX; e.g., if a user asks for sections as answer elements, like in the INEX 2005 CAS query 235:

```
//article[about(.//abs, "data mining")]
          //sec[about(., "frequent itemsets")]
```

he might be satisfied with paragraphs, abstracts, or even short articles (summaries) given as an answer. Furthermore, the list of element names can be larger in semantically richer and heterogeneous XML collections and it can evolve over time with the introduction of new XML collections.

The problem of element name matching is studied in the research area of schema matching and numerous techniques exist that try to resolve this problem (see [3, 18] for a survey)[2]. However, we decided to simplify the vague element name search task and use the INEX 2004 assessments to find the expanded element names (see the following section for more details).

Throughout the paper we discuss the application of our TIJAH system to vague search in XML documents. Vague search is modeled using the concept of vague XML element selection and rewriting techniques. The TIJAH system [12, 13, 14] is developed as a transparent XML-IR three-level database system for structured information retrieval, consisting of conceptual, logical, and physical layers. The original TIJAH system can handle queries with the strict selection of XML elements, specified in the NEXI query language [19] and can reason about textual information. In this paper we extend the TIJAH system toward handling vague specification of XML elements in the query (similar to [5]).

Additionally, we employ two rewriting techniques at the conceptual level that are used for extending the search on terms not only to the search elements deeper in the XML tree, but also to the higher-level elements that are used in the query formulation. We define two techniques for rewriting the original query as described in Section 2.

In this paper we aim to test whether we can automatically derive expansion lists that can be used to improve vague search and to compare them them with rewriting techniques. For that we need a test collection with a real set of vague queries. Although queries specified in the INEX adhoc CAS and COS tasks are declared as vague, it is not clear whether every structural condition is vague in them. Furthermore, due to the content of the collection, i.e., scientific articles, the XML markup is mostly used to represent document structure (article, title, abstract, sections, paragraphs, etc.). Therefore, the vagueness that can be expressed in INEX queries is mostly 'structural' vagueness (similarity between different structural elements of a document), and not 'semantic' vagueness (similarity between different concepts in different documents), which would be better suitable for our experimental setup. Even though INEX 2005 CAS and COS queries are not ideal for what we are aiming at and due to the lack of other suitable collections, we decided to test our approach on the INEX collection.

The following section explains the extensions introduced in the TIJAH system to model vague XML element specification and rewriting. The experimental setup is presented in Section 3. We conclude with the discussion of experiments performed using the INEX 2004 and 2005 collections in Section 4 and with conclusions and future directions in Section 5.

2. VAGUENESS IN USER QUERIES

This section details the motivation and the implementation of vague search in our three-level database framework. We explain the extensions at each level, conceptual, logical, and physical, aimed for vague search on elements and for rewriting the queries.

2.1 Vague search in NEXI

Instead of extending our conceptual parser for rewriting content-and-structure (CAS) and content-only plus structure (COS) queries into variants with strict or vague specification of target and support elements and as vague element specification should be expressed by the (expert) user we derive our own vague queries. Our vague queries are denoted with SS, VS, SV, VV in front of the CAS and COS query types, e.g., SVCAS, VSCAS, and VVCAS (SSCAS is equal to CAS in our case). Besides the vague selection of elements we model vagueness using query rewriting techniques.

2.1.1 Vague element selection

To express vague element selection we decided to extend the NEXI grammar with one extra symbol '∼'. The 'tilde' symbol is used in front of the element name in the query specification, denoting that the element name does not have to be strictly matched in the query evaluation. As we decided to simplify the vague element name search task, instead of more advanced schema matching techniques used to find the expanded element names, we use the results from the graded INEX 2004 assessments [11]. The list of expanded element names is defined based on the list of element names assessed as relevant in the INEX 2004 assessments process. The lists that we use, termed *element name expansion lists* are the following:

- One manually specified set of lists with the default score 0.55 (based on the 2004 experiments) denoted as *manual*55, given in Table 1[3]. The list is formed by selecting the most frequent highly and fairly exhaustive elements and adding the most frequently used INEX

[2]Note that the schema matching approaches are concerned with matching the exact relations among elements besides element name matching, but due to the complexity of the problem we start our research by trying to understand the elementary problems such as element name matching.

[3]Other elements from the INEX IEEE collection are not included in Table 1 as they were not present as target elements in the 2004 topic set.

Table 1: Manual element name expansion lists based on INEX 2004 assessments.

El. name	Expanded element names
abs	abs, fm, kwd, vt, p, sec, article, bdy, ref
article	article, bdy, sec, abs, fm, bm, bib, bibl, bb, p, ref
atl	atl, st, fgc
bb	bb, bm, bibl, bib, atl, art
bdy	bdy, article, sec, abs, p, ref
bib	bib, bm, bb, atl, art
fig	fig, sec, st, p, fgc, st, atl
fm	fm, sec, abs, kwd, vt, p, article, bdy, ref
kwd	kwd, abs, fm, st, fgc, atl
p	p, vt, abs, sec, fm, article, bdy, st
sec	sec, abs, fm, vt, p, article, bdy, bm, app
st	st, atl, fgc
tig	tig, bb
vt	vt, p, sec, bm, fig

query elements, such as *sec* (section), *p* (paragraph), *abs* (abstract), to the expansion lists of each element where they are not present and for which they seem to be a reasonable expansion element name.

- Seven sets of lists automatically generated out of assessments with exhaustivity (E) and specificity (S) greater or equal to marginally (1), fairly (2), or highly (3) exhaustive or specific: *hh* ($E > 2, S > 2$), *hf* ($E > 2, S > 1$), *fh* ($E > 1, S > 2$), *ff* ($E > 1, S > 1$), *fm* ($E > 1, S > 0$), *mf* ($E > 0, S > 1$), *mm* ($E > 0, S > 0$). The default score is based on a number of relevant elements of that specific name, normalized by a total number of relevant elements[4], for all distinct target elements. The set of lists contained 8 original element names each, and each list in these sets contained on average between 7.38 for *hh* (with median 4.5) and 15.62 for *mm* (with median 16) expanded element names.

2.1.2 Query rewriting techniques

We also model vague node selection using two query rewriting techniques that we used in previous years for INEX [12, 14]. These rewriting techniques treat structural constraints as strict but add new *about* clauses that search for the same terms as in the original query but in different elements. The rewriting is done at conceptual level.

In the first rewriting approach (rw I), all terms that are in different *about* clauses in the same predicate expression, and are not at the top level (i.e., not in `about(., term)`) expression, are added to an extra top-level *about* clause in the same predicate expression.

The second approach (rw II), is an extension of the first one, where not only the terms from non top-level *abouts* are added to the new *about*, but also all the terms from the other predicate[5], if there exists any, are added to the top-level *about* in each predicate.

[4]We had to manually edit the `vt` (vitae) expansion list using the *manual*55 run as the distribution of assessments for `vt` elements significantly degraded the performance of our 2004 runs.

[5]Note that the NEXI syntax allows only two predicates with the *about* clause to be specified in a query [19].

2.2 The complex vague selection operator

The logical level of the TIJAH system is based on Score Region Algebra – SRA (see [13] for more details). The data model consists of a set of regions, each defined by its region start (s) and region end (e) positions, its region type (t), region name (n), and region score (p). The basic operators on regions are given in Table 2, where $r_1 \prec r_2 \equiv r_1.s > r_2.s \wedge r_1.e < r_2.e$. The first four define the selection of regions based on: region name and type – $\sigma_{n=name,t=type}(R)$, numeric value assigned to a region – $\sigma_{\diamond num}(R_1)$, and containment relation among regions – $R_1 \sqsupset R_2$ and $R_1 \sqsubset R_2$. The operator $R_1 \sqsupset R_2$ selects regions from R_1 that contain regions from R_2, and $R_1 \sqsubset R_2$ selects regions from R_1 that are contained in the regions from R_2.

The operator $R_1 \sqsupset_p R_2$ is used for computing scores based on the containment relation among two regions (R_1 and R_2) and the retrieval model specified using the function $f_{\sqsupset}(r_1, R_2)$ (see Section 3). The two operators, $R_1 \blacktriangleright R_2$ and $R_1 \blacktriangleleft R_2$, specify score propagation to the containing or contained regions respectively. The operators $R_1 \sqcap_p R_2$ and $R_1 \sqcup_p R_2$ specify score combination in an AND and OR like combination of regions at the logical level.

Vague node selection at the conceptual level (i.e., in NEXI) is translated into a complex vague node selection operator at the logical level. However, the vague node selection operator in score region algebra has more expressive power than the simple NEXI extension at the conceptual level. It allows much finer specification of search and answer elements than a simple vague '$\sim$' node name specification. The vague node selection operator in SRA is defined as a union of all XML element regions that match the names of the 'expanded name regions' within the element name expansion list. By default all 'expanded regions' are down-weighted by a predefined factor. The definition of the operator is given in the last row of Table 2.

In the definition of $\sigma^{expansion(class)}_{n=name,t=type}(R_1)$, $expansion(class)$ is a set that contains the expansions for all the region names in one expansion class, where expansion list for each region *name* is denoted as $expansion(class, name)$ (with cardinality n):

$$expansion(class, name) := \{(ex\_n_1, ex\_w_1), (ex\_n_2, ex\_w_2), ..., (ex\_n_n, ex\_w_n)\}$$

Here ex_n_i is a expanded element name and ex_w_i is a real number in the range $[0, 1]$ denoting the down-weight factor. The operator $\sigma^{expansion(class)}_{n=name,t=type}(R_1)$ assigns name (ex_n) and score (ex_w) values to the region name (n) and score (p) based on the name and score values in the expansion list $expansion(class, name)$.

The simple selection operator in basic score region algebra operator set $\sigma_{n=name,t=type}(R)$ can be considered as a complex selection operator where the $expansion(class, name)$ set contains only the *name* element with $ex\_w = 1.0$. Note that the complex selection operator can also be expressed using the basic SRA selection operator and scaling operator ($R \circledast w$ denotes that the score of regions in the region set R should be multiplied by w, i.e., $r.p := r.p \cdot w$) as follows:

$$\sigma^{expansion(class)}_{n=name,t=type}(R) :=$$
$$(\sigma_{n=ex\_n_1,t=type}(R) \circledast ex\_w_1) \sqcup_p (\sigma_{n=ex\_n_2,t=type}(R) \circledast ex\_w_2)$$
$$\sqcup_p \ ... \ \sqcup_p (\sigma_{n=ex\_n_n,t=type}(R) \circledast ex\_w_n) \quad (1)$$

Table 2: Score region algebra operators.

Operator	Operator definition
$\sigma_{n=name,t=type}(R)$	$\{r \mid r \in R \wedge r.n = name \wedge r.t = type\}$
$\sigma_{\diamond num}(R_1)$	$\{r_1 \mid r_1 \in R_1 \wedge \exists r_2 \in C \wedge r_2.t = \text{term} \wedge r_2 \prec r_1 \wedge r_2.n \diamond num\}$, where $\diamond \in \{=, <, >, \leq, \geq\}$
$R_1 \sqsupset R_2$	$\{r_1 \mid r_1 \in R_1 \wedge \exists r_2 \in R_2 \wedge r_2 \prec r_1\}$
$R_1 \sqsubset R_2$	$\{r_1 \mid r_1 \in R_1 \wedge \exists r_2 \in R_2 \wedge r_1 \prec r_2\}$
$R_1 \sqsupset_p R_2$	$\{(r_1.s, r_1.e, r_1.n, r_1.t, f_{\sqsupset}(r_1, R_2)) \mid r_1 \in R_1 \wedge r_1.t = \text{node}\}$
$R_1 \blacktriangleright R_2$	$\{(r_1.s, r_1.e, r_1.n, r_1.t, f_{\blacktriangleright}(r_1, R_2)) \mid r_1 \in R_1 \wedge r_1.t = \text{node}\}$
$R_1 \blacktriangleleft R_2$	$\{(r_1.s, r_1.e, r_1.n, r_1.t, f_{\blacktriangleleft}(r_1, R_2)) \mid r_1 \in R_1 \wedge r_1.t = \text{node}\}$
$R_1 \sqcap_p R_2$	$\{(r_1.s, r_1.e, r_1.n, r_1.t, p_1 \otimes p_2) \mid r_1 \in R_1 \wedge r_2 \in R_2 \wedge (r_1.s, r_1.e, r_1.n, r_1.t) = (r_2.s, r_2.e, r_2.n, r_2.t)\}$
$R_1 \sqcup_p R_2$	$\{(r.s, r.e, r.n, r.t, p_1 \oplus p_2) \mid r \in R_1 \vee r \in R_2\}$
$\sigma^{expansion(class)}_{n=name,t=type}(R_1)$	$\{(r_1.s, r_1.e, r_1.n, r_1.t, r.p) \mid r_1 \in R_1 \wedge r_1.t = type \wedge (r_1.n, r.p) \in expansion(class, name)\}$

Table 3: Equivalence classes in INEX IEEE collection.

El. name	Equivalent names
h	h, h1, h1a, h2, h2a, h3, h4
list	list, dl, l1, l2, l3, l4, l5, l6, l7, l8, la, lb, lc, ld, le, numeric-list, numeric-rbrace, bullet-list
p	p, ilrj, p1, p2, p3, ip1, ip2, ip3, ip4, ip5, item-none
sec	sec, ss1, ss2, ss3

As our NEXI extension does not allow explicit specification of the 'expanded regions' list we pre-defined the 'expanded regions' set and pre-specified the default value for *weight*. For such a purpose we used manually predefined lists in Table 1 and seven automatically generated lists from the INEX 2004 assessments and combine them with the INEX equivalence classes given in Table 3 [10]. In this way we also kept the framework fairly simple.

The strict (SS) runs discussed in Section 4 use equivalence classes defined for the INEX IEEE collection [10], depicted in Table 3 and termed *eq_class*, as these represent the default setup in INEX. For the vague selection we used the fusion of equivalence classes and our INEX 2004 expansion element name lists given in Table 1 and in seven automatically extracted lists. This is done in such way that every expanded element name in these lists that has the equivalent name in the *eq_class name* part is also expanded with the *eq_class* equivalent names for *name*. These expansions are termed *manual*55 and *xx* for other seven lists, where $x \in \{h, f, m\}$, as explained in the previous section.

Therefore, the *eq_class* selection on section elements can be expressed as $\sigma^{expansion(eq\_class)}_{n='\text{sec}',t=node}(R)$, and vague node selection $\sim$`sec`, using highly exhaustive and highly specific elements, can be transformed into the next SRA operation $\sigma^{expansion(hh)}_{n='\text{sec}',t=node}(R)$. In such a way we can transparently define the set of expanded nodes and their respective weights and use them for vague node selection in a vague element name selection retrieval scenarios.

2.3 The implementation of vague selection

At the physical level, since we are working with the known INEX IEEE data collection, and as we used static INEX equivalence element name lists and expansion element name lists based on INEX 2004 assessments, we decided to replicate the lists and store them as tables at the physical level, i.e., in MonetDB [2]. Thus, we have eight tables with *(entity_name, expansion_name, expansion_weight)* for *manual*55 and *xx* lists, and one *(entity_name, equivalent_name)*[6] for *eq_class* list. The complex selection operator is then implemented as an additional MIL (MonetDB Interpreter Language) function to functions implementing other operators [14], based on the definition given in the previous section, that uses data from these tables.

For example, the vague *name* selection operator on region table R and the 'expansion regions' table S for the *hh* element names, in the relational algebra can be defined as:

$$\pi_{r.s,r.e,r.n,r.t,s.weight}(\sigma_{s.n=name}(S) \bowtie_{s.n=r.n} (\sigma_{r.t=node}(R)))$$

3. EXPERIMENTAL SETUP

Below, after introducing the retrieval models instantiated in score region algebra operators and metrics reported in the paper, we illustrate our approaches for INEX CAS and COS (sub)tasks.

3.1 Retrieval models

We base the instantiation of retrieval models on language models [6] since they showed good performance in our previous experimental runs [14]. For the relevance score computation on regions we use Equation 2, where *Root* is the root region of the collection and $size(r) := r.e - r.s - 1$.

$$f_{\sqsupset}(r_1, R_2) = r_1.p(\lambda \frac{\sum_{r_2 \in R_2 | r_2 \prec r_1} r_2.p}{size(r_1)} + (1-\lambda)\frac{|R_2|}{size(Root)}) \quad (2)$$

For upwards score propagation and downwards score propagation we employ Equation 3 and Equation 4.

$$f_{\blacktriangleright}(r_1, R_2) = r_1.p \cdot \sum_{r_2 \in R_2 | r_1 \prec r_2} r_2.p \quad (3)$$

$$f_{\blacktriangleleft}(r_1, R_2) = r_1.p \cdot \sum_{r_2 \in R_2 | r_2 \prec r_1} r_2.p \quad (4)$$

The abstract operators $\otimes$ and $\oplus$ in the score combination operators, $\sqcap_p$ and $\sqcup_p$, are implemented as product and sum respectively.

3.2 Metrics

For the evaluation of our 2004 and 2005 runs we use some of the official INEX 2004 and 2005 metrics, and precision at fixed recall points. For the 2004 runs we use the *inex_eval* tool, with both strict and generalized quantization, and we report a set-based overlap (aka O-overlap out of four overlap

[6]In the experiments we do not store weights for equivalent region names as we assume that their default weight is 1.0.

types distinguished in [16]). The *inex_eval* tool is based on the concept termed expected search length [17], and it uses three levels of exhaustivity and specificity: marginal, fair, and high [11]. Additionally, following the idea that simple metrics can give enough evidence for the evaluation of XML retrieval [7, 16], we report precision at three recall points: 10, 25, and 50.

The official INEX metrics for 2005 ad-hoc track are based on the extended Cumulative Gain (xCG) metric [9]. The official metrics are: normalized xCG (nxCG), effort-precision /gain-recall (ep/gr), and extended Q and R [8]. We report the evaluation results of our 2005 runs using nxCG at recall points 10, 25, and 50, as it can be compared to the precision at the low recall points. nxCG actually measures the gain a user has accumulated up to the specific rank, compared to the gain he could have accumulated if the ranking was ideal. The evaluation can be done either with generalized or with strict quantization. We also report the ep/gr MAP. ep/gr measures the user effort in inspecting the retrieved elements with respect to his effort in case the ranking was ideal, which resembles the *inex_eval* measure. We use only VVCAS and COS.Thorough assessments as we wanted to test the approaches on the same assessments set and without going into discussion over the overlap issue [9].

3.3 Vague CAS and COS queries

Since we decided to extend the NEXI syntax with vague selection we had to manually rewrite the queries for each CAS and COS scenario except the SSCAS and SSCOS scenarios. For example, the (SS)CAS query 225:

```
//article[about(.//fm//atl, "digital libraries")]
            //sec[about(.,"information retrieval")]
```

is rewritten into three variants:

- SVCAS:

```
//article[about(.//~fm//~atl, "digital libraries")]
        //sec[about(.,"information retrieval")]
```

- VSCAS:

```
//article[about(.//fm//atl, "digital libraries")]
        //~sec[about(.,"information retrieval")]
```

- VVCAS:

```
//article[about(.//~fm//~atl, "digital libraries")]
        //~sec[about(.,"information retrieval")]
```

We do not consider the 'article' element as a vague element in case it is not the target element or it is not the element in which the *about* search should be performed, as in these cases the 'article' element just serves as a focusing element for deeper search in the XML tree.

3.4 Query rewriting

As we explained in Section 2, we use two techniques for query rewriting. According to the first one we add the terms from the *about* clause that are not in the top-level element to the top-level element. For example, for INEX 2005 topic 240:

```
//article[about(.//(abs|kwd), quality control measure)]
            //sec[about(.//p, software quality)]
```

the rewritten query using rw I is:

```
//article[about(.//(abs|kwd), quality control measure)
            and about(., quality control measure)]
            //sec[about(.//p, software quality) and
            about(., software quality)]
```

Similarly, for the second rewriting technique (rw II), as it is an extension of the rw I technique, we use the rw I rewriting rule and also interchange terms from the other *about* clauses. Thus, for the same topic we have:

```
//article[about(.//(abs|kwd), quality control measure)
            and about(., quality control measure) and
            about(., software quality)]
            //sec[about(.//p, software quality) and
            about(., software quality) and
            about(., quality control measure)]
```

As can be seen in the next section we run these queries in isolation or in combination with the vague element selection queries.

4. DISCUSSION

In this section we discuss the results of experimenting with vague element selection and rewriting techniques on the INEX 2004 and 2005 collections. We start with estimating the best expansion lists and continue with the rw I and rw II experiments and their comparison and combination with vague selection. The result values given in bold in Tables 4 to 12 represent the highest scores (precision or MAP) in the column.

4.1 Estimating the best element name expansion lists

In this set of experiments we test whether automatically generated runs are comparable with the manual run and which of them is the best. As can be seen in Tables 4 to 6, except for the generalized MAP, all automatic runs are comparable and outperform the manual one in many cases, especially when looking at 2004 runs and precision at 25 and 50 for 2005 runs. Although we did not put much effort in specifying the expansion lists for our manual run, we think it is a good representative for what element names the expert user would accept in the results lists

Out of the automatic runs, the one that uses all relevant elements in the INEX 2004 assessments set (*mm*) seems to be the most effective. This is particularly the case for the early precision and overall MAP when using generalized quantization. Furthermore, this run shows constantly good results across different measures. This can be viewed as an indicator that the user really appreciates the wider set of element names in the expansion lists and that the only problem is how to estimate better their importance, i.e., down-weighting factor. Therefore, we selected *mm* and *manual55* runs for our further experiments.

4.2 Comparing vague element selection and query rewriting

Here we test if the effectiveness can be improved when replacing strict queries (*eq_class*) with vague ones and when using rewriting techniques. Furthermore, we compare the structured runs with the runs where structural constraints are removed from the queries (*no structure*). Tables 7 to 9 show that both the rewriting techniques and the vague element search improve the effectiveness with respect to

Table 4: INEX 2004 CAS experiments with different expansion classes evaluated using *inex_eval* and precision at different recall points.

	Strict				Generalized				
Exp. class	**Pr@10**	**Pr@25**	**Pr@50**	**MAP**	**Pr@10**	**Pr@25**	**Pr@50**	**MAP**	**Overlap**
ff	**0.0192**	0.0169	**0.0138**	0.08133	0.0769	0.0615	0.0477	0.05787	44%
fh	**0.0192**	**0.0185**	**0.0138**	0.08200	0.0846	**0.0646**	**0.0485**	0.05573	43%
fm	**0.0192**	**0.0185**	**0.0138**	0.08270	0.0769	0.0631	0.0477	0.05772	45%
hf	**0.0192**	0.0169	**0.0138**	0.08214	**0.0885**	0.0615	**0.0485**	0.05839	44%
hh	**0.0192**	**0.0185**	**0.0138**	0.08064	0.0808	0.0631	**0.0485**	0.05563	43%
mf	**0.0192**	**0.0185**	**0.0138**	0.08157	0.0846	0.0631	0.0469	0.05712	44%
mm	**0.0192**	**0.0185**	**0.0138**	**0.08330**	0.0846	0.0615	0.0462	0.05798	45%
manual55	0.0154	0.0108	0.0100	0.08202	0.0692	0.0462	0.0362	**0.06230**	60%

Table 5: INEX 2005 CAS experiments with different expansion classes evaluated using nxCG at different recall points and ep/gr.

	Strict				Generalized			
Exp. class	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**
ff	0.1333	0.1578	0.1511	0.01066	0.2711	0.2702	0.2534	0.06895
fh	**0.1444**	0.1578	**0.1556**	0.01081	0.2736	0.2736	0.2537	0.06666
fm	0.0778	0.1578	0.1511	0.01037	0.2723	0.2699	**0.2543**	0.07027
hf	0.0889	**0.1622**	0.1511	0.01067	0.2760	**0.2742**	0.2517	0.06870
hh	**0.1444**	**0.1622**	**0.1556**	0.01073	0.2767	0.2726	0.2520	0.06693
mf	**0.1444**	0.1578	0.1533	**0.01094**	0.2687	0.2685	0.2492	0.06824
mm	**0.1444**	0.1578	**0.1556**	0.01085	**0.2811**	0.2728	0.2529	0.07062
manual55	**0.1444**	0.1444	0.1467	0.01056	0.2545	0.2553	0.2428	**0.07296**

Table 6: INEX 2005 COS experiments with different expansion classes evaluated using nxCG at different recall points and ep/gr.

	Strict				Generalized			
Exp. class	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**
ff	0.0765	**0.0877**	0.0865	**0.00219**	0.2822	0.2564	0.2261	0.05907
fh	0.0765	0.0783	0.0830	0.00218	0.2765	0.2492	0.2202	0.05956
fm	0.0765	0.0854	**0.0877**	0.00180	0.2855	0.2541	0.2213	0.05783
hf	0.0765	0.0854	**0.0877**	0.00191	0.2869	0.2522	0.2197	0.05750
hh	0.0765	0.0830	0.0842	0.00218	0.2881	0.2495	0.2230	0.05831
mf	0.0765	0.0759	0.0830	0.00218	0.2849	0.2479	0.2238	0.05998
mm	0.0765	0.0830	0.0854	0.00218	**0.2912**	0.2580	0.2258	0.06060
manual55	**0.0824**	0.0759	0.0689	0.00218	0.2851	**0.2585**	**0.2315**	**0.06872**

Table 7: INEX 2004 CAS experiments with different vague scenarios and rewriting techniques evaluated using *inex_eval* and precision at different recall points.

	Strict				Generalized				
Exp. class	**Pr@10**	**Pr@25**	**Pr@50**	**MAP**	**Pr@10**	**Pr@25**	**Pr@50**	**MAP**	**Overlap**
no strucure	0.0038	0.0031	0.0023	0.07503	0.0115	0.0077	0.0062	**0.06664**	38%
eq_class	0.0192	0.0185	0.0138	0.07117	0.0692	0.0615	0.0462	0.03746	25%
rw I	**0.0269**	**0.0215**	**0.0162**	0.07241	**0.0846**	**0.0692**	0.0469	0.03968	26%
rw II	**0.0269**	**0.0215**	**0.0162**	0.07909	**0.0846**	**0.0692**	0.0469	0.04485	27%
mm, SV	0.0192	0.0185	0.0138	0.07154	0.0808	0.0615	**0.0492**	0.03781	24%
manual55, SV	0.0192	0.0185	0.0138	0.07131	0.0769	0.0615	**0.0492**	0.03716	24%
mm, VS	0.0192	0.0185	0.0138	0.08078	0.0654	0.0554	0.0408	0.05730	45%
manual55, VS	0.0077	0.0077	0.0085	0.07709	0.0385	0.0354	0.0292	0.06233	59%
mm, VV	0.0192	0.0185	0.0138	**0.08330**	**0.0846**	0.0615	0.0462	0.05798	45%
manual55, VV	0.0154	0.0108	0.0100	0.08202	0.0692	0.0462	0.0362	0.06230	60%

Table 8: INEX 2005 CAS experiments with different vague scenarios and rewriting techniques evaluated using nxCG at different recall points and ep/gr.

	Strict				Generalized			
Exp. class	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**
no structure	0.1000	0.1095	0.1137	0.00311	0.1725	0.1762	0.1680	0.05610
eq_class	0.1000	0.1022	0.0867	0.00581	0.2799	0.2851	0.2644	0.05033
rw I	0.1000	0.1200	0.1022	0.00609	0.2687	0.2834	0.2645	0.04670
rw II	**0.1889**	0.1289	0.1022	0.00777	0.3030	**0.2977**	**0.2679**	0.05476
mm, SV	0.0889	0.1067	0.0911	0.00609	0.2865	0.2882	0.2626	0.05219
manual55, SV	0.1000	0.1022	0.0844	0.00609	**0.3066**	0.2853	0.2419	0.05291
mm, VS	0.1333	0.1533	0.1511	0.01012	0.2672	0.2658	0.2524	0.06749
manual55, VS	0.1444	0.1222	0.1400	0.00975	0.2316	0.2417	0.2391	0.06720
mm, VV	0.1444	**0.1578**	**0.1556**	**0.01085**	0.2811	0.2728	0.2529	0.07062
manual55, VV	0.1444	0.1444	0.1467	0.01056	0.2545	0.2553	0.2428	**0.07296**

Table 9: INEX 2005 COS experiments with different vague scenarios and rewriting techniques evaluated using nxCG at different recall points and ep/gr.

	Strict				Generalized			
Exp. class	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**
no structure	0.0667	**0.1489**	**0.1348**	**0.00613**	0.1313	0.2020	0.1958	0.04784
eq_class	0.0471	0.0559	0.0559	0.00153	0.2677	0.2258	0.1787	0.03205
rw I	0.0588	0.0748	0.0595	0.00161	0.2715	0.2430	0.1894	0.03323
rw II	0.0588	0.0677	0.0571	0.00158	0.2872	0.2467	0.1898	0.03409
mm, SV	0.0471	0.0559	0.0559	0.00153	0.2772	0.2333	0.1951	0.03657
manual55, SV	0.0471	0.0559	0.0559	0.00152	0.2727	0.2349	0.1972	0.03650
mm, VS	0.0765	0.0854	0.0854	0.00213	0.2827	0.2499	0.2042	0.04283
manual55, VS	**0.0824**	0.0807	0.0689	0.00215	0.2751	0.2410	0.2060	0.04587
mm, VV	0.0765	0.0830	0.0854	0.00218	**0.2912**	0.2580	0.2258	0.06060
manual55, VV	**0.0824**	0.0759	0.0689	0.00218	0.2851	**0.2585**	**0.2315**	**0.06872**

Table 10: INEX 2004 CAS experiments on combining vague search and rewriting techniques evaluated using *inex_eval* and precision at different recall points.

	Strict				Generalized				
Exp. class	**Pr@10**	**Pr@25**	**Pr@50**	**MAP**	**Pr@10**	**Pr@25**	**Pr@50**	**MAP**	**Overlap**
rw I	**0.0269**	**0.0215**	**0.0162**	0.07241	**0.0846**	**0.0692**	**0.0469**	0.03968	26%
rw II	**0.0269**	**0.0215**	**0.0162**	0.07909	**0.0846**	**0.0692**	**0.0469**	0.04485	27%
mm, VV	0.0192	0.0185	0.0138	0.08330	**0.0846**	0.0615	0.0462	0.05798	45%
manual55, VV	0.0154	0.0108	0.0100	0.08202	0.0692	0.0462	0.0362	0.06230	60%
mm, VV + rw I	**0.0269**	**0.0215**	**0.0162**	0.08062	**0.0846**	0.0677	0.0462	0.05993	46%
manual55, VV + rw I	0.0115	0.0092	0.0092	0.07411	0.0423	0.0308	0.0269	0.06563	61%
mm, VV + rw II	**0.0269**	**0.0215**	**0.0162**	**0.08494**	**0.0846**	0.0677	0.0462	0.06571	52%
manual55, VV + rw II	0.0115	0.0092	0.0092	0.07958	0.0423	0.0308	0.0269	**0.07372**	60%

Table 11: INEX 2005 CAS experiments on combining vague search and rewriting techniques evaluated using nxCG at different recall points and ep/gr.

	Strict				Generalized			
Exp. class	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**
rw I	0.1000	0.1200	0.1022	0.00609	0.2687	0.2834	0.2645	0.04670
rw II	0.1889	0.1289	0.1022	0.00777	0.3030	**0.2977**	**0.2679**	0.05476
mm, VV	0.1444	0.1578	0.1556	0.01085	0.2811	0.2728	0.2529	0.07062
manual55, VV	0.1444	0.1444	0.1467	0.01056	0.2545	0.2553	0.2428	**0.07296**
mm, VV + rw I	0.1778	**0.1711**	**0.1622**	0.00904	0.2734	0.2641	0.2603	0.05899
manual55, VV + rw I	0.1667	0.1622	0.1489	0.00926	0.2427	0.2691	0.2469	0.06872
mm, VV + rw II	**0.2000**	0.1378	0.1089	**0.01129**	**0.3092**	0.2815	0.2366	0.05760
manual55, VV + rw II	0.1889	0.1333	0.1111	0.01113	0.3005	0.2943	0.2556	0.06896

Table 12: INEX 2005 COS experiments on combining vague search and rewriting techniques evaluated using nxCG at different recall points and ep/gr.

	Strict				Generalized			
Exp. class	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**	**nxCG[10]**	**nxCG[25]**	**nxCG[50]**	**MAP**
rw I	0.0588	0.0748	0.0595	0.00161	0.2715	0.2430	0.1894	0.03323
rw II	0.0588	0.0677	0.0571	0.00158	0.2872	0.2467	0.1898	0.03409
mm, VV	0.0765	0.0830	0.0854	0.00218	0.2912	0.2580	0.2258	0.06060
manual55, VV	0.0824	0.0759	0.0689	0.00218	0.2851	0.2585	0.2315	0.06872
mm, VV + rw I	**0.0882**	0.0901	0.0818	0.00215	0.2929	0.2686	0.2262	0.06168
manual55, VV + rw I	0.0824	0.0930	**0.0871**	0.00216	**0.3040**	0.2676	**0.2395**	**0.07168**
mm, VV + rw II	0.0765	0.0759	0.0748	0.00213	0.2873	0.2492	0.2168	0.05878
manual55, VV + rw II	0.0824	**0.0954**	0.0836	**0.00219**	0.2956	**0.2688**	0.2262	0.06891

the strict queries as well as the unstructured queries. The only exceptions are MAP for the 2004 run and nxCG[25], nxCG[50], and ep/gr MAP for the 2005 COS run with strict quantization. The improvements when using vague search are significant and they can go up to more than 100%, e.g., for the MAP in "*manual55*, VV" run using generalized quantization with respect to the *eq_class* run as presented in Table 9. Looking at the rewriting techniques, the rw II shows overall better scores, especially for the early precision as can be seen in Table 8, and it has higher MAP values.

Clearly, the vague element selection has higher MAP values than the rewriting techniques, but in all CAS experiments it has lower precision at low recall points. This can indicate that the rewriting techniques might be used as a precision tool, while the vague element selection can be considered as a recall tool. Looking at different vague scenarios, namely SV, VS, and VV, and except for some early precision scores (see Table 8), VV runs seem to have the best performance. Therefore, "*mm*, VV" and "*manual55*, VV" runs are used in combination with the rewriting techniques for further experiments.

4.3 Combining vague element selection and query rewriting

The third set of experiments confirms our assumption about the rewriting techniques as a precision and vague element search as a recall tool. As can be seen in Tables 10 to 12 in most of the cases the combination of the rw I and rw II rewriting techniques and manual and automatic vague element search improves early precision. However, not in all cases we managed to keep the MAP values, especially for the rw II combinations as can be seen in Table 12.

5. CONCLUSIONS AND FUTURE WORK

Throughout the paper we show that the TIJAH database system is flexible enough to incorporate new advanced search techniques, such as vague element selection and query rewriting. We have shown that rewriting techniques and vague element selection are viable solutions for vague search in XML documents. While query rewriting techniques are more suitable for obtaining higher precision at low recall points, vague element selection yields higher average precision. Furthermore, we show that automatically generated runs give comparable results to manually generated ones. Finally, the combination of vague selection and rewriting technique approaches can boost early precision, but it may also have negative influence on mean average precision.

The continuation of the work presented in this paper include the experimental evaluation of different scenarios for search in structured documents: vague element search with different assignments of non-uniform down-weighting factors, e.g., using the value of exhaustivity and specificity in the assessments to derive more accurate down-weighting factors, or better chosen manual element name expansion lists, and their combination with rewriting techniques. For that, and for a more realistic experimental setup we would need a new large (heterogeneous) collection of structured documents, with more semantical information especially in element names (e.g., the collections provided by Wikipedia or Lonely Planet).

6. ACKNOWLEDGMENTS

Many thanks to the Netherlands Organisation for Scientific Research (NWO) for funding the research described in this paper (grant number 612.061.210).

7. REFERENCES

[1] S. Amer-Yahia, C. Botev, and J. Shanmugasundaram. TeXQuery: A Full-Text Search Extension to XQuery. In *Proceedings of the 13th WWW Conference*, 2004.

[2] P. Boncz. *Monet: a Next Generation Database Kernel for Query Intensive Applications*. PhD thesis, CWI, 2002.

[3] A. Doan and A.Y. Halevy. Semantic Integration Research in the Database Community. *AI Magazine*, 26:83–94, 2005.

[4] D. Florescu and I. Manolescu. Integrating Keyword Search into XML Query Processing. In *Proceedings of the 9th International WWW Conference*, 2000.

[5] N. Fuhr and K. Großjohann. XIRQL: An XML Query Language Based on Information Retrieval Concepts. *ACM Transactions on Information Systems*, 22(2):313–356, 2004.

[6] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, Twente, The Netherlands, 2001.

[7] D. Hiemstra and V. Mihajlović. The Simplest Evaluation Measures for XML Information Retrieval that Could Possibly Work. In *Proceedings of the INEX 2005 Workshop on Element Retrieval Methodology*, 2005.

[8] G. Kazai and M. Lalmas. INEX 2005 Evaluation Metrics. In *Proceedings of the Fourth Initiative on the Evaluation of XML Retrieval (INEX), to appear*, 2006.

[9] G. Kazai, M. Lalmas, and A.P. de Vries. The Overlap Problem in Content-oriented XML Retrieval Evaluation. In *Proceedings of the 27th ACM SIGIR Conference*, 2004.

[10] G. Kazai, M. Lalmas, and S. Malik. INEX'03 Guidelines for Topic Developments. In *Proceedings of the Second INEX Worksop*, ERCIM Workshop Proceedings, 2004.

[11] Gabriella Kazai. Report of the INEX 2003 Metrics Working Group. In *Proceedings of the 2nd INEX Workshop*, ERCIM Workshop Proceedings, 2004.

[12] J. List, V. Mihajlović, A. de Vries, G. Ramirez, and D. Hiemstra. The TIJAH XML-IR System at INEX 2003. In *Proceedings of the 2nd INEX Workshop*, ERCIM Workshop Proceedings, 2004.

[13] V. Mihajlović, H.E. Blok, D. Hiemstra, and P.M.G. Apers. Score Region Algebra: Building a Transparend XML-IR Database. In *Proceedings of the ACM CIKM Conference*, 2005.

[14] V. Mihajlović, G. Ramírez, A.P. de Vries, D. Hiemstra, and H.E. Blok. TIJAH at INEX 2004: Modeling Phrases and Relevance Feedback. In *Proceedings of the Third INEX Workshop*, volume 3493 of *Lecture Notes in Computer Science*, 2005.

[15] G.A. Miller, C. Fellbaum, R. Tengi, S. Wolff, P. Wakefield, H. Langone, and B. Haskell. WordNet: A Lexical Database for the English Language.

[16] J. Pehcevski and A. Thom. HiXEval: Highlighting XML Retrieval Evaluation. In *Proceedings of the Fourth INEX Workshop, to appear*, 2006.

[17] V. Raghavan, P. Bollmann, and G. Jung. A Critical Investigation of Recall and Precision. *ACM Transactions on Information Systems*, 7(3), 1989.

[18] E. Rahm and P.A. Bernstein. A Survey of Approaches to Automatic Schema Matching. *The VLDB Journal - The International Journal on Very Large Databases*, 10:334–350, 2001.

[19] A. Trotman and R. A. O'Keefe. The Simplest Query Language That Could Possibly Work. In *Proceedings of the Second INEX Workshop*, ERCIM Publications, 2004.

Archival metadata for durable data sets

E.H. Dürr
Euformatics
Scriveriusmate 28
8014 JW Zwolle
The Netherlands
E.H.Durr@phys.uu.nl

R. Dekker
Delft University of Technol.
Library
P.O. Box 98
2600 MG Delft
The Netherlands
R.Dekker@TUDelft.nl

K. van der Meer
Delft University of Technol.
Computer Science
P.O. Box 5031
2600 GA Delft
The Netherlands
K.vanderMeer@TUDelft.nl

ABSTRACT
The DARELUX project envisages long-term storage of hydrology measurement data in a permanent archive. In order to make the DARELUX hydrology data sets accessible in an unknown future, a future-proof metadata element set is wanted. An analysis of current standards and best practices for metadata lead to the choice of the metadata of the Dublin Core Metadata Element Set, with an addition of archival metadata elements, based on the OAIS Preservation Description Information (PDI) metadata.
A second problem is related to the encapsulation of an OAIS AIP package into an XML container file. This is the best we can do to let containers with data travel through time. Metadata and content are entangled together as an indivisible unit. The problem is that the PDI fixity/checksum value stores the value of the entire XML container, inside of the preservation metadata section of the container. However, storage of the value of the container means that the value of the container changes. Our solution to this, a procedure to store a valid checksum value of the XML container in the XML container is described.
A third problem is the geographical coverage. The Dublin Core prescription for the geographical representation is ambiguous; a choice was made for an encoding scheme.

Categories and Subject Descriptors
H.3.6 [Information storage and retrieval]: Library automation - *large text archives*

General terms
Documentation

Keywords
Metadata, Data sets, Longevity, OAIS, Preservation Description Information, Dublin Core, fixity, MD5

Proceedings of the sixth Dutch-Belgian Information Retrieval Workshop DIR 2006

1. PRESERVATION OF HYDROLOGY DATA SETS
The DARELUX (Data Archiving River Environment Luxemburg) project concentrates on preservation of measurement data sets of precipitation (rain, snow), water levels in currents, water discharge etc.: hydrology data sets (1).
Hydrology data sets like these are needed for models of water management, and they will continue to be needed for future models; water management is an important issue for the Low Countries. To make certain the usability of these data in the future, long-term storage of the data is crucial.
Based upon our experience with archiving digital information objects in a durable way, we are currently building the DARELUX repository; for the present state see (2).
There is a growing experience with preservation of digital publications. Preservation of data sets is in some aspects different. Data sets are often deleted after their use, but there are data sets that have to be preserved, as they will remain valuable a long time after they have been generated, and they are unique, they cannot be reconstructed once they would be lost. Hydrology data sets are an example of this type of data sets.
Especially for their use in an unknown future, it is essential to make the data sets retrievable. Without the prospect of future retrieval and access, preservation efforts simply do not make sense. The demand for future retrieval and access leads to the demand of a future-proof metadata element set. Determination of a suitable metadata element set, then, is the first problem.

2. METADATA ELEMENT SETS
The DARELUX project is one of the SURF / DARE projects. DARE puts conditions to the metadata accompanying the data sets: the DARE projects have been prescribed to use the Dublin Core Metadata Element Set, "DC", ISO standard 15836:2003 (3). Even more, DARE Guidelines to the DC have been issued (4), prescribing how to use the DC MES.
The prescription of a metadata element set and the edition of Guidelines for its actual use should be applauded. The use of standardised metadata assignment is the best we can do to make digital information objects retrievable. This is even more valid for future retrieval. However, the DC is not without drawbacks.
The DC is meant to enable retrieval and access to digital publications. Although the website (3) states, that there are no fundamental restrictions to the types of resources to which Dublin Core metadata can be assigned, the DC is not tailored to the description of data sets. Also, the DC offers no built-in support for, e.g., authenticity and provenance, i.e. archival

aspects needed to determine the value of the content in a distant future.
The fact that DARELUX deals with data sets could be overcome, but the archival aspect should be adapted. The two options for the DARELUX project are:

- Accept some other metadata element set, which gives rise to a difficult explanation to the project board that their conditions cannot be met
- Add metadata elements originating from records management or archival practice to the DC.

3. ARCHIVAL METADATA

3.1 Archival metadata element sets

There are several well-known archival metadata element sets.
1. ISAD(G) (General International Standard Archival Description, second edition, (5)) is an international description standard for cataloguing archival materials. It is based on older, national standards. It is used a lot and has a good reputation for the description of these sources.
2. EAD, (Encoded Archival Description, (6)) is a standard for encoding archival finding aids using XML. EAD is used a lot and a valuable tool, too.
3. The ISO standard for metadata for records ISO 23081 is a recent development (7). ISO 23081 is connected to the ISO standard for records management, ISO 15489 (8), and ISO 15489 is a most important standard for records management (it stresses, in a way, the same need for records management as the famous or infamous US Sarbanes-Oxley law (9)). Records management is not the same as a durable archive, but as the two are closely related, ISO 23081 must be considered, too.
4. The OAIS (Reference Model for an Open Archival Information System, ISO standard 14721:2002, (10)) metadata set. This set includes Preservation Description Information, PDI metadata elements. The OAIS PDI is a limited set that describes the use of four metadata elements: Reference information, Provenance information, Context information and Fixity information. Fixity information is described as "documents authentication mechanisms used to ensure that the Content information has not been altered in an undocumented manner (e.g. checksum, digital signature)". OAIS has the advantage that the developers of the OAIS metadata element set obviously included in their requirements that data sets would have to be described.
Due to the specific material, data sets, OAIS representation information is probably not needed.

González et al. compare these metadata element sets (for another divergent data type, viz. software components) based on granularity, suitability and compatibility for the data type, and simplicity for users (11). That is a basis to judge the DARELUX case, too.

3.2 Combination

If one metadata element set, with its own purpose and 'designated community', will not do, metadata element sets can be combined.
There are various crosswalks that can be used as start for a combination; see e.g. Day (12).
Additions to and combinations with DC have been proposed several times, especially before Qualified Dublin Core with its refinement had been established. For the data type of software components, González (11) added archival elements to the DC. Another example of DC extension to non-document-type information objects is the proposal of Bird for an extensible XDC scheme for a type of language vocabularies (13).

3.3 Experiences

Searle and Thompson describe a pragmatic approach at the National Library of New Zealand (14). Referring to experiences at the national Library of Australia, Cedars, the OCLC/RLG Working Group, it is emphasized that a balance should be found between the principles expressed in the OAIS Information Model and the practicalities of implementing a working set of preservation metadata. Unfortunately, there is no recipe for preservation metadata assignment.
The Dutch "Koninklijke Bibliotheek", the Dutch National Library, preserves digital publications in its Digital Information Archiving System (DIAS). The experiences at the DIAS are useful, despite that the DIAS is not meant for data sets and the DIAS is far larger than the DARELUX data base. DIAS is one of the largest repositories in the world with at present over 4 million publications. It is based upon Dublin Core-like XML-based metadata and compatible with the OAIS reference model. The Koninklijke Bibliotheek is considering the (rather extended) PREMIS model to preserve the publications in her DIAS. PREMIS (ref. 15, page 4-5: fixity, integrity, authenticity) states that these characteristics of a [digital] object have to be verified, but again the recipe is still under construction. The PREMIS working group does not seem to be operational yet.
In order to ensure encapsulation the use of digital containers as the basis of OAIS AIP's (Archival Information Packages) is defended. The use of XML containers stems from a previous project called EArchive, where the unity of metadata and content was developed (16). In the AIP's, the containers contain both the metadata and all representations of the digital archive information object.
Evidently, there is not much experience yet with metadata element sets for preserved data sets.

4. DESIGN

4.1 Choice of the element set

The demand for simplicity and flexibility (criteria also used by González) of DC, compared to ISAD(G) and EAD, would induce a choice for DC even if it had not been prescribed. ISAD(G) is broad and general and cannot be tailored easily to properties of types of documents. ISAD(G) does not primarily seem to think of retrieval. ISAD(G) and EAD both have more to offer for archival aspects than DC but it is a bit overdone. By the way, Boudrez (17) states that ISAD(G) is of limited use for digital archive documents, and the DARELUX data sets are surely digital.
An advantage of the DC is the mass of users in the present "designated community" for the DARELUX data sets. Also in that respect ISAD(G) and EAD are not superior to the DC.
There is overlap between the ISAD(G) and DC and the EAD and DC metadata element sets. Upon inspection it is clear that

their structure is different from the DC structure. It is not easy to make a consistent set of ISAD(G) or EAD combined with DC.
A disadvantage to the ISO standard 23081 for metadata for records is that only Guidelines to the metadata for records standard have been published, there is no accepted metadata element set that is ready to be used or can be tailored for this purpose and that is broadly accepted.
The OAIS metadata structure has the advantage, that the OAIS PDI metadata types are a suitable complement to the DC: simple, concise, no overlap, just an addition.
Next to the conceptual connection, cost aspects are a factor. For the preservation of hydrology data sets, DARELUX must aim at minimal effort to make the data sets accessible. Hydrology data sets do not rank high in the list of cultural heritage materials. So, the keeping of this type of raw material is faced with a second problem, a financial one, leading to a second (less important, but not to be neglected) selection criterion.
Probably, the extent of external financial support is restricted; it is not even sure whether financial support from a government body will be granted, although we (of course) feel that the need to preserve these hydrology data sets is beyond doubt. In this respect, the situation for hydrology data sets may be different from, e.g., archeological data sets and is surely different from normal archive documents.
ISAD(G) and EAD describe extensive tag libraries, their implementation would require a lot of effort. The costs to assign the small set of OAIS PDI metadata are relatively low.
So, the combination DC + OAIS PDI was chosen.
Finally: it has been brought to our attention that the DC has been extended with (among others) a provenance element. It is not part of the Simple DC, and different from the refinement and encoding scheme that is the base of the Qualified DC. We cannot judge the acceptance among users of this element; that is essential for the DARELUX users. So, we can not choose for DC alone. But the use of just one set would obviously be preferred over two; and if the DC would be extended with a dedicated, recognizable, concise set of archival metadata elements (and why not the current OAIS PDI), that could well be preferred.

4.2 Structure for long-term preservation

In the DARELUX project, metadata and content are stored together in self-descriptive containers in XML format. An XML container, an indivisible unit in which metadata and content are stored together, has a good chance to travel unchanged through time. It continues to be the base for automated processing. Relying on any linking mechanism (e.g. via a data base) between metadata and content requires that archiving organisations are obliged to maintain the technical linking infrastructure (data base software) over a very long time. A linking mechanism is a considerable risk for long-term preservation. That is the background of the original idea of Lourens et al. (16). Boudrez (17) evidently follows the same reasoning.
The purpose of OAIS is an archival information system. In the DARELUX repository, or in any repository for that sake, digital information objects have to be able to travel through time. Therefore, the main focus is the longevity of digital information objects. The solution that is being used to manage these digital objects (e.g. databases or linking mechanisms) is subordinate to accomplishing that goal. As the purpose of the DARELUX project is different from the OAIS system, the way of working is different.

4.3 Fixity

The OAIS PDI metadata structure is suitable as an addition to the DC for the DARELUX repository.
As stated, OAIS PDI describes four metadata element types. Of these four, the fixity item poses an extra problem. The metadata is stored in an XML container together with the bit stream. So, the fixity information is stored in the container.
The fixity describes the value of an information object, e.g. by stating the calculated checksum.
In DARELUX containers, there are several possibilities.
One could calculate the checksum on the content part only, but then the metadata could be changed; that is not to be preferred.
One could calculate the checksum of the complete container, but then a problem arises. When the value of the container is calculated, and filled in in the fixity field (in the container), the value of the container changes.
One could calculate the value of the checksum for the content and the metadata separately. This again leads to the problem that the value, in this case of the metadata, changes upon completion of the field.
We propose the next, probably unambiguous way to deal with this problem. The checksum tag will be completed before calculation with XXXXXXXXXXX (12 times X). This XXXXXXXXXXX will most likely draw attention. It is evidentially not a hexadecimal value, so one kind of misunderstanding is omitted. And we hope that mistakes with a value ‘0’ by a future data archeologist are avoided. Next, the checksum is calculated and the checksum value is stored in the checksum tag. Control of the checksum value in the future has to follow the same procedure: the checksum tag has to be saved, overwritten by XXXXXXXXXXX, and then the checksum value can be calculated (and the value has to correspond to the original value in the checksum tag).
This procedure has to be stored in the XML container, too, of course; in the fixity field.
OAIS prescribes a separation of the bit stream and the metadata. In our case, the bit stream and the metadata are not separated. This requires an explanation.

4.4 OAIS PDI overview

As a result, the following metadata element types are designed to be added to the DC:
- The OAIS Reference information field is omitted. Reference information is covered sufficiently in the DC metadata.
- The OAIS Provenance information contains e.g. information on the data set provenance and the equipment, and any data on refreshment, i.e. bit stream preservation and eventually on migration.
- The OAIS Context information contains any contextual data related to the measurement data
- The OAIS Fixity information contains several tags: the choice of the way of calculation, in the DARELUX case MD5; the recipe to calculate the fixity value (including XXXXXXXXXXX); and the MD5 value of the container. The MD5 algorithm is probably stable, it is widely used and well-known; moreover MD5 is as near a standard as can be for a subject in the field of internet information, as MD5 has been published as RFC 1321 (18). The publication of an RFC is the

best guarantee for the future use of a digital information structure.

4.5 DC geographic coverage

In the Dublin Core metadata set an item called <DCMI:Coverage> is included. This item is meant to include time period and spatial information on the geographic location for the document. In the DCMI part two options are offered: either a (geolocation) Point <DCMI:Point> or a geographic rectangular area <DCMI:Box>. For a point the following components are defined: east, north, elevation, units, zunits, projection and a name. For a box the components are northlimit, eastlimit, southlimit, westlimit, uplimit, downlimit, units, zunits, projection and a name.

Several encoding schemes are proposed in the documentation (19) like DCSV (semicolon separated), and XML either with sub-elements or with attributes. This document states: "Given the flexibility of XML many alternatives are possible. One possible form is: ...". Next, an example XML element definition with corresponding DTD is given.

In the published schema for the DC elements (20) currently the coverage item is defined as a generic element type. This is a string with optionally a language attribute. The consequence is that no sub-elements are allowed by this schema.

The consequence of this choice is that inside the DC metadata no uniform encoding scheme is defined (chosen) for geographical information items. All kinds of indexing schemes using DC metadata are difficult to develop as they have to deal with a wide variety of encoding schemes or deliver ambiguous results.

This issue may become very relevant in the near future where "mobile information retrieval" based upon the geographic (from GPS or Galileo systems) location of the user is needed. Examples are public transport, tourism, traffic information etc. Also in our hydrologic data sets geographic location of the sensors evidently plays a vital role. It is a major parameter in building models based upon these data sets now and in the future.

A choice is necessary. In the DARELUX project it was chosen to use an encoding with elements for the components and an attribute for the name. We had to introduce our own name space (dl) because the DC schema only allowed text strings here.

Like the example from the DCMI Box the result is:

```
<dl:spatial name="Maisbich" >
   <northlimit>49.8942</northlimit>
   <eastlimit>6.0506</eastlimit>
   <southlimit>49.8812</southlimit>
   <westlimit>6.0303</westlimit>
 </dl:spatial>
```

4.6 The archival macro level

Finally, we find that data sets are different from more "normal" documents. A single data set consists of the measurements for one geographic location for a certain time interval (e.g. a month). The DC metadata for each of them is identical apart from the time period indication field. Indexing based on DC delivers many "hits". That may be unwanted.

The choice for metadata assignment may be analogous to the different levels of description of archives: micro (individual information items are considered), meso (folders are considered) and macro (complete collections are considered). More research is needed to the question, whether new retrieval methods are needed in such situations, where "classic" indexing based on meta-data fields for content consisting of individual data sets are less suited, and how to use these methods for a digital repository.

5. CONCLUSIONS

In the DARELUX project metadata must be assigned to make data sets in the hydrology data repository retrievable and accessible, also in a distant future. The use of the Dublin Core Metadata Element Set was prescribed to make these data accessible, a Guideline was available, and Dublin Core is a valuable start, but it is not enough. In addition to the DC, metadata on archival aspects are necessary to make the DARELUX repository useable in the distant future. Moreover, due to budgetary constraints, metadata assignment must be cheap (a condition that may be applicable to other data sets as well).

Based on over a year of experiences with the DARELUX repository, our findings are:

1. The combination of DC plus OAIS Preservation Description Information metadata elements seems to be the best fit for retrieval of and access to the DARELUX hydrology data sets in the long term. This, by the way, is not strikingly different from the solution of González for software components.
2. The data set and its corresponding metadata are saved together in XML containers.
3. A conjuring trick enables to deal with the recording of the checksum value of the XML container inside of the container it describes.
4. An overview of the contents of the OAIS PDI elements has been given.
5. A choice was made as to the description of the geographical information by DC in the DCMI Point item.
6. More research is needed to the possibility for repositories to describe collections of data sets.

The current and foreseen implementation promise metadata that are sufficient for future use. It should enable a "data archeologist" in the far future to use the current DARELUX hydrology data sets.

We acknowledge the valuable comments of an unknown referee.

6. REFERENCES

All references have been checked 23rd February 2006

[1] DARELUX Data Archiving River Environment LUXemburg http://www.library.tudelft.nl/darelux/

[2] DARELUX archieftoegang (in Dutch) http://www.library.tudelft.nl/darelux/3872/f_EN.html

[3] Dublin Core Metadata Element set. ISO standard 15836:2003. http://dublincore.org/documents/dces/

[4] DARE use of Dublin Core version 2.0, December 2004 http://www.surf.nl/download/DARE%20use%20of%20DC%20v.%202.0.pdf

[5] General International Standard Archival Description. Second edition, 1999 http://www.ica.org/biblio/cds/isad_g_2e.pdf

[6] Encoded Archival Description, 2002 http://www.loc.gov/ead/

[7] Metadata for records. ISO/Technical Standard 23081-1:2004.
[8] Records management. ISO Standard 15489:2001.
[9] R. Kahn & B.T. Blair: The Sarbanes-Oxley act: understanding the implications for information and records management. http://www.bitpipe.com/detail/RES/1089741697_942.html
[10] Reference model for an Open Archival Information System (OAIS). Also: ISO standard 14721:2002 http://www.ccsds.org/documents/650x0b1.pdf
[11] R. González and K. van der Meer: Standard metadata applied to software retrieval. Journal of Information Science 30(4), (2004), 300-309.
[12] M. Day: Metadata: mapping between metadata formats. http://www.ukoln.ac.uk/metadata/interoperability/
[13] S. Bird: A simpler format for OLAC vocabularies and schemes. http://listserv.linguistlist.org/cgi-bin/wa?A2=ind0209&L=olac-implementers&D=1&F=&S=&P=192
[14] S. Searle and D. Thompson: Preservation Metadata. D-Lib Magazine 9(4), (April 2003) http://www.dlib.org/dlib/april03/thompson/04thompson.html
[15] PREMIS http://www.oclc.org/research/projects/pmwg/premis-final.pdf
[16] W. Lourens and E. Dürr: Programs for Ever. ICEIS 2001 NDDL Workshop November 2001. http://durr.dhs.org/ =>earchive/publications
[17] F. Boudrez: Digitale archiefcontainers voor het digitaal archiefdepot (in Dutch) http://www.expertisecentrumdavid.be/docs/digitale_containers.pdf
[18] R. Rivest: The MD5 Message-digest algorithm http://www.ietf.org/rfc/rfc1321.txt
[19] S. Cox: DCMI encoding scheme http://dublincore.org/documents/dcmi-box/
[20] Schema for DC elements http://dublincore.org/schemas/xmls/simpledc20021212.xsd

Dictionary-independent translation in CLIR between closely related languages

Anni Järvelin
+46-480-411662
anni.jarvelin@uta.fi

Sanna Kumpulainen
+358-505298901
sanna.kumpulainen@uta.fi

Ari Pirkola
+358-14-762278
pirkola@cc.jyu.fi

Eero Sormunen
+358-3-35516972
eero.sormunen@uta.fi

Department of Information Studies, FIN 33014, University of Tampere, Finland

ABSTRACT
This paper presents results from a study, where fuzzy string matching techniques were used as the sole query translation technique in Cross Language Information Retrieval (CLIR) between the closely related languages Swedish and Norwegian. It is a novel research idea to apply only fuzzy string matching techniques in query translation. Closely related languages share a number of words that are cross-lingual spelling variants of each other. These spelling variants can be translated by means of fuzzy matching. When cross-lingual spelling variants form a high enough share of the vocabulary of related languages, the fuzzy matching techniques can perform well enough to replace the conventional dictionary-based query translation. Different fuzzy matching techniques were tested in CLIR between Norwegian and Swedish and it was found that queries translated using skipgram matching and a combined technique of transformation rule based translation (TRT) and n-grams perform well. For the best fuzzy matching query types performance difference with respect to dictionary translation queries was not statistically significant.

Categories and Subject Descriptors
H.3.3 [**Information Systems**]: Information Search and Retrieval

General Terms
Performance, Experimentation

Keywords
Cross-language retrieval, Fuzzy matching

1. INTRODUCTION
Information retrieval methods are based on comparing the words in requests with the words in documents. Cross Language Information Retrieval (CLIR) refers to the retrieval of documents in other languages than the language of the request. For an overview of different approaches to CLIR, see [6]. Fuzzy string matching methods are used for finding matches between words that are similar but not identical. In CLIR fuzzy string matching has been used for handling proper names and technical terms, as well as other cross-lingual spelling variants not found in translation dictionaries [5, 12, 17]. McNamee and Mayfield [7] have used n-grams in corpus-based query translation.

Closely related languages have not been considered as a separate line of research in CLIR. The dominating approach, dictionary-based translation of queries, is a fairly effective technique, but has its problems in the limited coverage of dictionaries and the constant need for updating, which can make it an expensive technique. Closely related languages typically share a high number of spelling variants, i.e., equivalent words that share the same origin and are similar (but not identical). If the number of the shared cross-lingual variants is high enough, query translation can be handled by much cheaper and simpler fuzzy techniques.

Among fuzzy techniques n-gram and skipgram matching have been found to be effective in monolingual proper name [10] and cross-lingual spelling variant matching [5, 12] and transformation rule based translation technique (TRT) has been found to be an effective method for translating cross-lingual spelling variants [17]. N-grams and skipgrams are language independent techniques and the TRT technique can be easily adjusted for new language pairs. The methods are therefore easily applicable for new languages and thus ideal translation methods in CLIR. They are not dependent on expensive linguistic resources. In this study we used these dictionary-independent fuzzy string matching techniques as a query translation technique between closely related languages. The techniques were tested with the Scandinavian language pair Norwegian and Swedish, with Norwegian as the source language and Swedish as the target language.

Scandinavian languages have not been studied extensively from the information retrieval point of view. Hedlund et al. [3] is an exception. In their study characteristics of Swedish in information retrieval were analyzed. Swedish and Norwegian together with Danish form a language group where the speakers of one language can quite easily understand the other languages, especially in written form. Both the grammar and vocabulary of the languages are similar as they have developed in a close historical and cultural relation to one another. Some 50% of the Swedish and Norwegian (Bokmål) vocabulary is identical and around 40% similar, when inflected word forms and orthographical differences of using æ/ø instead of ä/ö are not considered [1]. There are also consistent and frequently occurring differences in the orthographies of Swedish and Norwegian. For

Proceedings of the sixth Dutch-Belgian Information Retrieval Workshop (DIR 2006)

example, Norwegian avoids the use of letters c, z, and x (*center (Swe) – senter (No)*) and the letter d is often left out of words where Swedish has it (*kunde (Swe) – kunne (No)*), the Danish letters æ/ø are used in Norwegian instead of Swedish ä/ö and the Swedish word endings -sion, -ssion and –tion are written –sjon in Norwegian. These similar features suggest that the use of fuzzy string matching techniques and the statistical transformation rules might be efficient in query translation from Norwegian to Swedish.

The research problems investigated in this paper are as follows:

1. Are fuzzy string matching methods as effective as the dictionary-based translation techniques in CLIR between closely related languages like Norwegian and Swedish?

2. Which of the fuzzy string matching methods tested is the most suitable translation technique for CLIR between closely related languages?

To the best of our knowledge, attempting to solve the query translation problem in CLIR between closely related languages with fuzzy string matching techniques without dictionary translation is a novel research idea not tried before.

The rest of the paper is organized as follows. The fuzzy string matching techniques used in this study are introduced in Section 2. Section 3 presents the test environment, methods and data. The similarity between Norwegian and Swedish is discussed in Section 4. Section 5 presents the findings and Section 6 discussion and conclusions for the study.

2. TRANSLATION TECHNIQUES

2.1 N-grams and Skipgrams

N-gram matching is a language independent method for matching words whose character strings are similar [13, 14]. Query keys and words in documents are decomposed into n-grams, i.e. into substrings of length *n*. The degree of similarity between the query keys and index terms can then be computed by comparing their n-gram sets. For a description of the applications of the technique, see [14]. N-gram matching has been reported to be an effective technique among fuzzy string matching techniques in name searching [10] and in cross-lingual spelling variant matching [5]. McNamee and Mayfield [7] used a direct corpus-based n-gram query translation technique, where the source language n-grams were directly translated to the target language n-grams using aligned corpora. The translation technique using 4- and 5-grams was found feasible. They also found n-grams an effective technique in tokenization, as it outperformed the stemmer used. Also Adafre et al. [1] have used 4-grams combined to a parallel corpus in query translation.

N-grams can consist both of adjacent characters or non-adjacent characters of the original words. Pirkola et al. [12] devised a novel matching technique for n-grams formed of non-adjacent characters, called the classified skipgram matching technique. In this technique digrams are divided into categories (classes) on the basis of the number of the skipped characters and only the digrams belonging to the same class are compared with each other. *Gram class* indicates the number of skipped characters when digrams are formed from a string *S*. *Character combination index* (CCI) then indicates a set of gram classes enumerating all the digram sets to be produced from the string *S*. For example *CCI* = {{0},{1,2}} means that two gram classes are formed from the string: one with conventional digrams formed of adjacent characters and one with skip-digrams formed both by skipping one and two characters [5]. The classified skipgrams have performed better than the traditional n-grams in the empirical tests examining the matching of cross-lingual spelling variants [5, 12].

It is common to use padding spaces in the beginning and in the end of the strings when forming n- and skipgrams. If the padding spaces are not used, the characters at the front and at the end of the strings will be under-represented in the gram set that is generated. Keskustalo et al. [5] tested different types of padding spaces for conventional digrams, trigrams and skipgrams, and found that using padding spaces both in the beginning and the end of the words gave the best results. However, the use of end padding spaces has been found unsuitable for inflectionally complex suffix languages, such as Finnish, where the use of the beginning padding only has been found beneficial [12]. This way of down-weighting the word ends – the inflectional suffixes – was assumed to be useful also when handling Swedish and Norwegian. For n-grams it is common to use a padding of n-1 characters [14]. For skipgrams a padding that varies according to the number of the skipped characters can be used.

The similarity values for n-grams are computed with a string similarity scheme [10]:

$$SIM(w_1, w_2) = \frac{|N_1 \cap N_2|}{|N_1 \cup N_2|}, \text{ where}$$

N_i is a digram set of a string w_i, $|N_1 \cap N_2|$ denotes the number of intersecting n-grams in strings w_1 and w_2, i.e. n-grams that the strings have in common, and $|N_1 \cup N_2|$ denotes the number of unique n-grams in the union of N_1 and N_2. The similarity measure for skipgrams is then defined between two strings *S* and *T* with respect to the given CCI as follows [5]:

$$SIM_{\mathrm{CCI}}(S,T) = \frac{\Sigma i_{\mathrm{CCI}} |DS_{\mathrm{i}}(S) \cap DS_{\mathrm{i}}(T)|}{\Sigma i_{\mathrm{CCI}} |DS_{\mathrm{i}}(S) \cup DS_{\mathrm{i}}(T)|}, \text{ where}$$

DS_i is the digram set of a string, *i* denoting the gram class, $|DS_i(S) \cap DS_i(T)|$ denotes the number of intersecting n-grams and $|DS_i(S) \cup DS_i(T)|$ the number of unique n-grams in the union of the strings *S* and *T*.

2.2 The TRT Technique

Transformation rule based translation (TRT) is a fuzzy translation technique based on the use of statistically generated rules of regular character correspondences in cross-lingual spelling variants within a language pair. The technique resembles transliteration, phonetic translation across languages with different writing systems, but no phonetic elements are included and the technique is meant for processing languages sharing the same writing system. It is applied in two-steps: the transformation rules are combined to n-gram matching. The idea of the TRT and the generation of the transformation rules are described in more detail in [17].

A *transformation rule* contains source and target language characters and their context characters [17]. In addition the frequency and the confidence factor of the rule are recorded. *Frequency* refers to the number of the occurrences of the rule in the data used for generating the rules. *Confidence factor* is the frequency of a rule divided by the number of source words where the source substring of the rule occurs. They are important threshold factors that can be used for selecting the most reliable rules for the translation. An example of a Norwegian to Swedish rule is:

for för beginning 132 147 89.80

The rule can be read: the letter o, prior to r and after f, is transformed into the letter ö in the beginning of words, with the confidence factor being 89.80. The confidence factor is calculated from the frequency of the rule (132) and the number of source words where the string occurs (147).

In this study we used the thresholds of confidence factor = 50% and frequency = 2.

3. METHODS AND DATA

3.1 Test Topics and Collection

The performance of the fuzzy translation methods was tested by running CLIR tests with a set of 60 topics used in the CLEF evaluation forum in the year 2003 [9]. Norwegian and Swedish topics were used, of which Swedish topics were included in the collection of the CLEF topics. To get the Norwegian topics, English topics were translated by a native Norwegian speaker. Of the two official Norwegian languages the more common Bokmål was used. In ten of the topics, queries failed in preliminary test runs for technical reasons. These topics were removed from all of the queries and the final tests were run with the remaining 50 topics. The target document collection was the Swedish CLEF document collection containing 142819 newspaper articles obtained from the Swedish news agency TT (Tidningarnas Telegrambyrå) published in 1994-1995 [9]. The document collection was lemmatized using Swetwol morphological analyzer by Lingsoft Inc. Compounds were split into their constituents and both the original word and the constituents were lemmatized and indexed. Words not recognized by the morphological analyzer were indexed as such to a separate index of unrecognized words. We used the InQuery Retrieval System as the search engine. InQuery is a probabilistic information retrieval system based on the Bayesian inference net model, where queries can be presented as unstructured bag-of-words queries or they can be structured with a variety of operators [2].

3.2 Creating TRT Rules

To create the word-pair list used for generating the Norwegian to Swedish transformation rules a part of the Swedish document collection's index was translated to Norwegian with the Global Dix dictionary by Kielikone plc. Words not recognized by the morphological analyzer were removed and, as the index was too large to use as a whole, every sixth word of the index was chosen. This list contained 6714 word-pairs. Word-pairs with an edit distance value bigger than half of the length of the longer word in the word-pair or including a word shorter than four characters were removed. The final word-pair list included 3058 unique word-pairs. This list seemed to be insufficient for generating enough high frequency transformation rules. This lack of high quality rules may have affected negatively the TRT technique's translation results.

3.3 N- and Skipgram Matching

The n- and skipgram translations were done by matching the n- or skipgrams of the topic words against the normalized index words of the Swedish test collection. The index was divided into two: the index of the words recognized by the morphological analyzer and the index of unrecognized words. Dividing the index is helpful when matching proper names [4]. For n-digram translation we used beginning weighted n-digrams with the padding of 1. Leaving out the padding at ends of words gives more weight to the beginnings of words, which can be useful when the words are inflected. For skipgram translation, a padding of the number of the skipped characters + 1 was used. For example for gram class 1, the skipgrams were formed with two padding spaces.

3.4 Queries

We used five sets of test queries, which were compared to three sets of baseline queries. The five translation methods tested were n-digrams, classified skipgrams with CCI = {{0}{1}} (*Skip1*) and CCI = {{0}{1,2}} (*Skip2*), plain TRT translation and the combined TRT and n-digram technique. The set of baseline queries consisted of a monolingual Swedish query set (*Swebase*), a monolingual Norwegian query set (*Nobase*) and a dictionary translated Norwegian to Swedish query set (*Dicbase*). The Global Dix dictionary was used for the translations. The Swebase and Dicbase gave high performing baselines, while the Nobase was used for testing how high performance is achieved without any translation and how much the fuzzy methods can improve this result.

The test query types were as follows. The query operators used in a query are presented in parentheses and examples of the queries are presented in Appendix 1.

1) Swedish monolingual baseline (#sum)

2) Norwegian monolingual baseline (#sum)

3) Dictionary baseline (#sum, #syn, #uw7)

4) N-digram query (#sum, #syn)

5) Skip1 query (#sum, #syn)

6) Skip2 query (#sum, #syn)

7) Plain TRT query (#sum, #syn)

8) Combined TRT and n-digram query (#sum, #syn)

The queries were formed from the title- and description fields of the CLEF topics. The topic words were lemmatized with the morphological analyzer Twol. For the dictionary translation, compound words were split into constituents that were then translated separately. This is because compound components are more often found in dictionaries than the whole compounds. For other query types, no compound splitting was done, as we assumed the compounds in Norwegian and Swedish to be similar. The lemmatized source words were translated and stop words were removed both before and after the translation.

The queries were formulated by grouping the query keys with InQuery's operators *sum*, *syn* and *uwn*. The sum-operator computes an average of query key weights for keys grouped by the operator. It is used for grouping the whole query and can include either the query keys without any structure or query key sets structured with the other operators. The syn-synonym operator treats its operand query keys as synonyms. The unordered proximity operator with a window size n (uwn) allows free word-order and combines the translations equivalents of the constituents of a source language compound [13].

The Swedish and Norwegian monolingual baseline queries were formed directly from the Swedish and Norwegian topic words as bag-of-words queries without any structure. The rest of the queries were structured with the syn-structure (*Pirkola's method*), which has been found effective in CLIR [11, 13, 16]. For the Dicbase queries all the translation equivalents of a source word were selected to the query and were grouped together with the syn-operator. When the translation was a noun phrase, its words were combined with a proximity operator of uwn, where we set the value of *n* to seven. Words not found in the dictionary were added to the query as such.

All the five test query types were structured queries, where the translation equivalents selected for a source word were grouped together with the syn-operator. For the n-gram and skipgram queries we selected for each source word the four highest ranked keys from the result list of n-gram matching. This selection was based on the findings by Hedlund et al. (2004), who showed that the best retrieval performance is achieved using just a few n-gram keys in queries [4]. These keys included two keys from the index of words recognized by the morphological analyzer and two from the index of unrecognized words.

For plain TRT-queries all the translated keys from the TRT result list were selected for each of the source word for the final queries. The combined TRT and n-digram queries were formed by selecting the first word form of each of the original source words from the TRT result list, which was then matched to the Swedish database index using n-digram matching. The word forms created with a rule combination with the highest confidence factor and frequency values get the highest position in the TRT result list. The four highest ranked keys from the result list of n-gram matching were then selected for the final queries like in other n-gram techniques.

3.5 Performance Measures

The effectiveness of the test queries was measured by Mean Average Precision (MAP) i.e., the average non-interpolated precision calculated over all relevant documents, and by interpolated recall precision averages at standard recall levels of 10 and 50, averaged over all queries. The test queries' precision-recall graphs were created using the eleven standard recall levels and the test queries' graphs were compared. The statistical significance of the results was tested using the Friedman two-way analysis of variance by ranks. The statistical significance levels are indicated in the tables.

4. SIMILARITY BETWEEN NORWEGIAN AND SWEDISH

To get an insight to how close two languages should be for the fuzzy matching to be practicable, the similarity of Swedish and Norwegian language was measured. A measure based on the Longest Common Subsequence (LCS) [8] was used, and German and English were used as a baseline language pair. They belong to the same language group but are not so closely related to make fuzzy matching alone a sufficient translation technique. The average similarity values measured for Swedish and Norwegian and for English and German were 0,815 and 0,556 respectively.

LCS is a measure that counts the maximum amount of letters that two words share and have in the same order, for example for a English - German word pair *motivation - motivierung* the longest common subsequence *motivin* has length 7. The data used for measuring the similarities between the languages included 167 word pairs for both language pairs. The vocabulary was selected from two sources: 71 words were chosen from the CLEF'03 topics and 96 words from a word list containing work environment vocabulary in all four languages (from the TNC-termbank by the Swedish national centre for terminology, TNC). The similarities were measured by first measuring the LCS values pair wise for all the words. Then each of these LCS values was divided by the length of the longer word of the word pair. Finally a mean value was calculated of these pair wise word similarity values for both language pairs. The similarity values range between 0-1. For example for the Swedish-Norwegian word pair *brevbomb - brevbombe* the LCS value is 8 and the similarity is counted by dividing it with the length of the longer of the words (here 9), with the similarity value being $8/9 \approx 0{,}889$.

Swedish, Norwegian and German are *compound languages* [4], i.e. languages where the components of multi-word expressions are written together, whereas English is a *non-compound language* where multi-word expressions are written as phrases (*fackförening, fagforening, gewerkschaft,* but *trade union*). The way the multi-word expressions are written is an important feature when measuring the orthographical similarity of languages. Therefore the test data included multi-word expressions. Phrases were written together by using a '_' to mark the space between the components (*trade_union*).

The similarity value of 0,815 measured for Swedish and Norwegian can be illustrated with examples: For a pair of short words such as *skola – skole* one character substitution results in a similarity value of 0,8. A longer word pair with a similarity value of 0,818 is *ioniserende – joniserande*, where two character substitutions happen. The orthographical differences in Swedish and Norwegian words are typically at this level. The mean similarity value of 0,556 measured for English and German corresponds to changes like *north_sea – nordsee*, which share five out of nine letters and get the similarity value of 0,556. The short word pairs *night – nacht* and *level - pegel,* where three letters out of five are common, get a similarity value of 0,6. The source of the vocabulary affected the similarity values slightly: the Swedish-Norwegian values for CLEF and TNC vocabularies were 0,829 and 0,805, respectively. English-German values were 0,582 for CLEF words and 0,536 for TNC words.

5. FINDINGS

5.1 The Performance of Fuzzy String Matching in Comparison to Baselines

Table 1 summarizes the Mean Average Precision values for all query types, and the performance differences between the test queries and the baseline queries. As the performances of the n-digram, skipgram and the combined TRT-n-gram queries were quite close to each other, they are referred together as the *n-gram queries* in the following. The performance differences between these queries are considered in Section 5.2.

The MAP is a measure that rewards techniques that retrieve relevant documents quickly [18]. When comparing the MAP values, the dictionary translation gives the best results, the monolingual Swedish baseline being second. The n-gram queries perform well: differences to the Dicbase and Swebase results are not statistically significant for any of the queries. The practical differences to the Dicbase are nevertheless noticeable (according to [15]) being over 5% for all fuzzy queries. All these techniques performed both statistically significantly and practically noticeably better than the Norwegian monolingual baseline. The plain TRT query's performance was better than the Nobase's, the difference not being statistically significant. The TRT query's performance was statistically significantly weaker than the Dicbase and Swebase baselines' performance.

Tables 2 and 3 present the recall precision averages at standard recall levels of 10 and 50. The Precision-Recall curves for all query types are shown in Figure 1. As can be seen from the P-R curves, the dictionary baseline and the Swedish monolingual baseline perform best on the high precision levels (0-20) and middle recall levels (20-80). For the high recall levels (80-100) the differences even up and the two skipgram queries perform as well as the Swebase baseline. Nobase and plain TRT queries still perform worse than the other queries.

At the recall level of 10 (Table 2), the dictionary baseline gets the highest precision average. The Swedish baseline is again the second best query type. The n-gram queries perform well, the differences to Dicbase and Swebase not being statistically significant. All the n-gram queries perform markedly better than the Nobase. Plain TRT query type is clearly worse than the Dicbase and Swebase baselines.

At the recall level of 50 (Table 3), the differences between different techniques diminish but the trend is still clear: Dictionary translation gives the best result followed by the monolingual Swedish query. The n-gram queries perform also well, the difference to Dicbase and Swebase not having statistical significance, although the practical differences between n-gram queries and Dicbase are noticeable. The plain TRT queries and Nobase are clearly the two weakest query types; their differences to the other query types are statistically significant.

5.2 Best Fuzzy String Matching Technique

The fuzzy queries were also compared to each other to determine the most suitable technique for CLIR between closely related languages. As can be seen from Figure 1, the plain TRT queries' P-R curve is consistently clearly below the other curves. The difference to the other fuzzy queries is most of the time statistically significant or highly significant, and the practical difference is always noticeable. Therefore it can be concluded that, when used alone, it is not an adequate translation technique in CLIR between closely related languages. Earlier research results from [17] support this conclusion. In this research, the TRT queries' performance may have been negatively affected by the lack of high frequency transformation rules. This may also have affected the results of the combined TRT-n-gram queries.

The findings do not suggest one fuzzy string matching technique as being the best translation method in CLIR between closely related languages. The differences between the different n-gram queries were small and statistically insignificant. The combined TRT-n-gram queries performed best on the high precision levels and the practical difference to the plain n-gram queries was noticeable at the recall level of 10. On the middle recall levels all the n-gram queries were quite even and their differences had no statistical or practical significance at the recall level of 50. Here the skipgram queries gave the best results, the Skip2 -grams with CCI={{0}{1,2}} being the best query type. From the Figure 1 it can be seen that the P-R curves of skipgram queries are above the others fuzzy queries' curves at the high recall levels.

Even if the differences are small, the Skip2 queries and the combined TRT-n-gram queries performed slightly better than the other queries. At the same time, the combined TRT-n-gram queries outperformed the plain n-gram queries indicating that the transformation rules do improve n-gram results in CLIR between closely related languages.

Table 1. The MAP values (%) for the test queries and their difference to the baselines (%) (* statistically significant difference, ** statistically highly significant difference)

	Baseline queries			**Test queries**				
	Nobase	Swebase	Dicbase	Skip1	Skip2	N-gram	Plain TRT	TRT-n-gram
Precision	12,64	31,76	34,13	28,34	28,63	26,53	16,88	27,74
Difference to Nobase	0	19,12	21,49	15,7*	*15,99**	13,89**	4,24	15,1**
Difference to Swebase		0	2,37	-3,42	-3,13	-5,23	*-14,88***	-4,02
Difference to Dicbase			0	-5,79	*-5,5*	-7,6	*-17,25***	-6,39

Table 2. The interpolated recall precision averages (%) at standard recall level 10 for the test queries, and their difference to the baselines. (* statistically significant difference, ** statistically highly significant difference)

	Baseline queries			Test queries				
	Nobase	Swebase	Dicbase	Skip1	Skip2	N-gram	Plain TRT	TRT-n-gram
Precision	21,85	50,65	54,91	44,39	43,95	41,44	28,17	46,54
Difference to Nobase	0	28,8	33,06	22,54**	22,1*	19,59**	6,32	24,69**
Difference to Swebase		0	4,26	-6,26	-6,7	-9,21	-22,48**	-4,11
Difference to Dicbase			0	-10,52	-10,96	-13,47	-26,74**	-8,37

Table 3. The interpolated recall precision averages (%) at standard recall level 50 for the test queries, and their difference to the baselines. (* statistically significant difference, ** statistically highly significant difference)

	Baseline queries			Test queries				
	Nobase	Swebase	Dicbase	Skip1	Skip2	N-gram	Plain TRT	TRT-n-gram
Precision	13,1	31,03	35,64	28,81	29,58	27,02	15,78	28,77
Difference to Nobase	0	17,93	22,54	15,71	16,48	13,92*	2,68	15,67**
Difference to Swebase		0	4,61	-2,22	-1,45	-4,01	-15,25**	-2,26
Difference to Dicbase			0	-6,83	-6,06	-8,62	-19,86**	-6,87

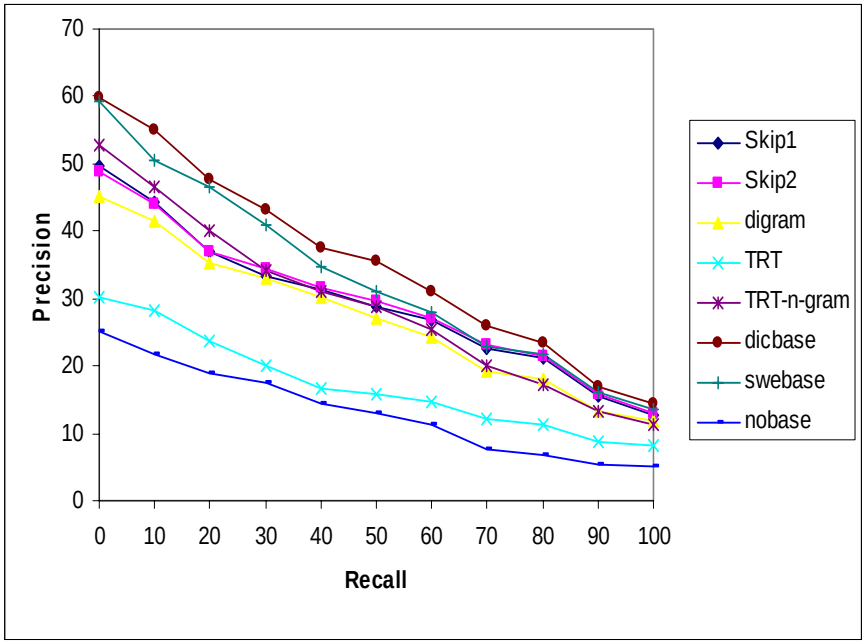

Figure 1. Recall-precision curves for all queries.

6. DISCUSSION AND CONCLUSIONS

The aim of this research was to find out (1) if fuzzy matching techniques are as effective as the dictionary-based translation techniques in CLIR between closely related languages like Norwegian and Swedish, and (2) the most suitable fuzzy string matching technique for query translation in CLIR between closely related languages. The effectiveness of five fuzzy string matching techniques was tested for Norwegian to Swedish query translation with CLEF search topics from the year 2003. The fuzzy techniques were compared to three baseline techniques, which

were a dictionary translation baseline, a monolingual Swedish baseline and a monolingual Norwegian baseline.

Our main findings were:

- The fuzzy (n-gram) matching techniques are effective and applicable translation techniques in CLIR between closely related languages. For the best fuzzy matching query types performance difference with respect to dictionary translation queries was not statistically significant.
- The results do not suggest one best fuzzy matching technique for CLIR between closely related languages.
- The TRT technique alone is not a good approach (however, see below for the generation of transformation rules).

The results were encouraging giving support to our hypothesis that dictionary-based translation could be replaced by fuzzy string matching techniques in CLIR between closely related languages. The n-gram based techniques performed well, skipgrams being slightly better than conventional n-grams. This is in line with earlier research, where skipgrams have been found to be better than n-grams in matching cross-lingual spelling variants [5, 12]. Combining n-grams to the TRT techniques' statistical transformation rules improved results, the practical difference being of noticeable (5,1%) at the recall level 10. The TRT-n-grams also outperformed the best skipgrams at low recall levels. This suggests that the combined technique is useful in CLIR, as also found in earlier research [17]. The results also give reason to assume that combining the transformation rules to skipgram matching would be a good approach. This combination can be assumed to perform well, as the skipgrams have been shown to outperform the conventional n-grams in cross-lingual spelling variant matching [5, 12].

The results suggests that the transformation rules should be formed on a basis of a larger term pair list than was done in this study, or the list should be formed from technical terms instead of general vocabulary. The performance of the TRT queries might improve if the transformation rules were thereby improved. Better transformation rules might also further improve the performance of the combined TRT and n-gram queries.

In the present research, all the query words were lemmatized because the transformation rules in their current state can only handle base forms. Creating transformation rule collection capable of handling inflected word forms will be one of the next steps in our research. Our future research will also include testing the combination of TRT and skipgrams, and extending the research to concern Danish language.

7. REFERENCES

[1] Adafre, S., van Hage, W., Kamps, J., de Melo, G. & de Rijke, M. 2004. The University of Amsterdam at CLEF 2004. CLEF 2004 Working Notes. Available at: Available at: http://clef.iei.pi.cnr.it/

[2] Barðdal, J., Jörgensen, N., Larsen, G., & Martinussen B. 1997. Nordiska: Våra språk förr och nu. Lund, Studentlitteratur.

[3] Broglio, J., Callan, J. & Croft B. 1993. Inquery system overwiev. In Proceedings of the TIPSTER text program, 47-67. Available: http://acl.ldc.upenn.edu/X/X93/X93-1008.pdf

[4] Hedlund, T., Pirkola, A. & Järvelin, K. 2001. Aspects of Swedish morphology and semantics from the perspective of mono- and cross-language information retrieval. Information Processing & Management, 37, 147-161.

[5] Hedlund, T., Airio, E., Keskustalo, H., Lehtokangas, R., Pirkola, A. & Järvelin, K. 2004. Dictionary-based Cross-Language Information Retrieval: Learning Experiences from CLEF 2000-2002. Information Retrieval – Special Issue on CLEF Cross-Language IR, 7, 99-119.

[6] Keskustalo, H. & Pirkola, A. & Visala, K. & Leppänen, Erkka & Järvelin, K. 2003. Non-adjacent Digrams Improve Matching of Cross-Lingual Spelling Variants. In: Nascimento, M.A., de Moura, E.S., Oliveira, A.L, (Eds.). Proceedings of the 10th International Symposium, SPIRE 2003. Manaus, Brazil, October 2003. Berlin: Springer, Lecture Notes in Computer Science 2857, pp. 252 - 265. ISSN 0302-9743, ISBN 3-540-20177-7.

[7] Kraaij, W. 2004. Variations on language modeling for information retrieval. PhD thesis, University of Twente.

[8] McNamee, P. & Mayfield, J. 2003. JHU/APL Experiments in Tokenization and Non-Words Translation. CLEF 2003 Working Notes. Available at: http://clef.iei.pi.cnr.it/

[9] Navarro, G. 2001. A Guided tour to approximate string matching. ACM Computing surveys (CSUR) (33)1.

[10] Peters, C. 2003. Introduction to the CLEF 2003 Working Notes. Available at: http://clef.iei.pi.cnr.it/

[11] Pfeiffer, U., Poersch, T. & Fuhr, N. 1996. Retrieval effectiveness of proper name search methods. Information Processing & Management, 32(6), 667-679.

[12] Pirkola, A. 1998. The effects of query structure and dictionary setups in dictionary-based cross-language information retrieval. In: Proceedings of the 21st Annual International ACM Sigir Conference on Research and Development in Information Retrieval, Melbourne, August 24-28. New York: ACM, 55-63.

[13] Pirkola, A., Keskustalo H., Leppänen, E., Känsälä, A.P. & Järvelin, K. 2002. Targeted s-gram matching: a novel n-gram matching technique for cross- and monolingual word form variants. Information research, 7(2) [http://InformationR.net/ir/7-2/paper126.html]

[14] Pirkola, A., Puolamäki, D. & Järvelin, K. 2003. Applying query structuring in Cross-Language Retrieval. Information Processing & Management 39(3), 391-402.

[15] Robertson, A.M. & Willet, P. 1998. Applications of n-grams in textual information systems. Journal of Documentation, 54(1), 48-69.

[16] Spark Jones, K. 1974. Automatic indexing. Journal of Documentation (30) 4, 393-432.

[17] Sperer, R. & Oard, D. W. 2000. Structured Translation for Cross-Langiage Information Retrieval. In Belkin, N. & Ingwersen, P. & Leong, M-K. (Eds.), Proceedings of the 23rd Annual International SIGIR Conference on Research and Development in Information Retrieval, 120-127. Athens, Greece.

[18] Toivonen, J., Pirkola, A., Keskustalo, H., Visala, K., & Järvelin, K. 2005. Translating cross-lingual spelling variants

using transformation rules. Information Processing & Management, 41, 859-872.

[19] Voorhees, E. M. 2002. Overview of TREC 2002, Appendix 1. Common Evaluation Measures. The Proceedings of the eleventh Text REtrieval Conference. Gaithersburg, Maryland. National Institute of Standards and Technology. [http://trec.nist.gov/pubs.htm

Appendix 1. Examples for query types

Swebase

#sum(christo packeterar tyska riksdagshus konstnär christo inslagning tyska riksdagshus)

Nobase

#sum(christo pakke tysk riksdagsbygning innpakking tysk riksdag berlin kunstner christo)

Dicbase

#sum(christo #syn(paket packe bunt ask packa) #syn(tysk tyska) #syn(regerings stats stat statlig) dag #syn(byggnadsverk byggnad konstruktion hus) #syn(packning) #syn(tysk tyska) #syn(regerings stats stat statlig) dag berlin konstnär christo)

N-digram query

#sum(#syn(mchistori chefshistorik @christo @christos) #syn(paket pakets @paker @pak) #syn(tysk tysktysk @tyskl @tysklan) #syn(riksdagsbyggnad riksdagsbevakning @riksdagsoch @landsbygsriksdagen) #syn(skinnpaj inpassning @pakkinen @iakkinen) tysk #syn(tysk tysktysk @tyskl @tysklan) #syn(riksdag riksdagsdag @riksdagsoch @riksdagsrupp) #syn(berliner berlinsk @berlin @berlins) #syn(kungstiger kungakonst @kunst @kunstler) #syn(mchistori chefshistorik @christo @christos))

Skip1 query (CCI = {{0},{1}})

#sum(#syn(chefjurist charterturistort @christo @christos) #syn(packe paket @takke @pakue) #syn(tysk tysktysk @tyskl @otysk) #syn(riksdagsbevakning riksdagsordning @riksdagsoch @riksdagsrupp) #syn(inpackning inpassning @ing @king) #syn(tysk tysktysk @tyskl @otysk) #syn(riksdag riksdagsdag @riksdagsoch @riksdagsrupp) #syn(berglin merlin @berlin @berlins) #syn(konstnär konstnummer @kunstler @köstner) #syn(chefjurist charterturistort @christo @christos))

Skip2 query (CCI = {{0},{1,2}})

#sum(#syn(tyristor mchistori @christo @christos) #syn(paket packe @pakue @takke) #syn(tysk tysktysk @tyskl @otysk) #syn(riksdagsbyggnad riksdagsbevakning @riksdagsebatten @riksdagsrupp) #syn(inpackning inpassning @king @parking) #syn (tysk tysktysk @tyskl @otysk) #syn(riksdag riksdagsdag @riksdagsoch @riksdagsrupp) #syn(berglin merlin @berlin @berlins) #syn(konstnär konstcenter @kunstler @kunstlers) #syn(tyristor mchistori @christo @christos))

TRT query

#sum(#syn(christo) #syn(packa pakka packe pakke) #syn(tysk) #syn(riksdagsbygning) #syn(innpacking innpakking) #syn(tysk) #syn(riksdag) #syn(berlin) #syn(kunstner) #syn(christo))

Combined TRT and n-digram

#sum(#syn(mchistori chefshistorik @christo @christos) #syn(packa packad @packard @packalén) #syn(tysk tysktysk @tyskl @tysklan) #syn(riksdagsbyggnad riksdagsbevakning @riksdagsoch @landsbygsriksdagen) #syn(inpackning inpacka @inpac @racking) #syn(tysk tysktysk @tyskl @tysklan) #syn(riksdag riksdagsdag @riksdagsoch @riksdagsrupp) #syn(berliner berlinsk @berlin @berlins) #syn(kungstiger kungakonst @kunst @kunstler) #syn(mchistori chefshistorik @christo @christos))

Automatic Extraction of Knowledge from Greek Web Documents

Fotis Lazarinis
Technological Educational Institute of Mesolonghi
30200 Mesolonghi, Greece
0030-26310-58148
lazarinf@teimes.gr

ABSTRACT
Extracting textual data from Greek corpuses poses additional difficulties than in English texts as inclinations and intonation differentiate terms of equal information weight. Pre-processing and normalization of text is an important step before the extraction procedure as it leads to fewer rules and lexicon entries, thus to less execution time and greater success of the mining process. This paper presents a system accessible via the Web which automatically extracts data from Greek texts. The domain of conference announcements is utilized for experimentation purposes. The success of the extraction procedure is discussed on the basis of an evaluative study. The conclusions and the techniques discussed are applicable to other domains as well.

Categories and Subject Descriptors
H.2.8 [**Database Application**]: Data mining

H.3 [**Information Systems**]: Information Search and Retrieval

Keywords
Web mining, information extraction, XML storage, multilingual retrieval

1. INTRODUCTION
Some recent studies showed that common search engines supporting Greek do not actually understand specific characteristics of the language [7, 8] so utilizing a general purpose search engine to discover specific information such as dates, keywords or even general purpose terms demand more effort by the user resulting also to lower success. This is mainly due to differences in Greek terms caused by inclinations, intonation and lower and upper case forms.

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

In this paper we present a tool for extracting the title, keywords, event date, submission deadline and location of conference announcements. This tool is based on the identification of patterns and on knowledge lexicons (dictionaries) for extracting the previously mentioned data. Pre-processing and normalization of text is an important step before the extraction procedure as it leads to fewer rules and lexicon entries and to greater success of the mining process. Our main aim is not simply to build a system with extraction capabilities but to explore additional inconveniences and present solutions applicable in mining data from Greek corpuses which show considerable grammatical diversity although they carry the same information weight. The conclusions of this work could be applied to other spoken languages with similar characteristics to the Greek language.

2. EXTRACTING TEXTUAL DATA
Information extraction systems analyze unrestricted text in order to extract specific kind of information. They process documents trying to identify pre-defined entities and the relationships between them, filling a structured template with the mined information. Such systems have been implemented to extract data such as names and scientific terms from chemistry papers [2, 12]. Gaizauskas and Robertson [4] used the output of a search engine as input to a text extraction system. Their domain was management succession events and their scenario was designed to track changes in company management.

More contemporary work uses co-occurrence measurement in order to identify relationships and to extract specific data from Web pages [9]. Han et al [5] extract personal information from affiliation, such as emails and addresses, based on document structure. Efforts on Greek information extraction are recorded as well. In [11] a rule based approach to classify words from Greek texts was adapted. Rydberg-Cox [14] describes a prototype multilingual keyword extraction and information browsing system for texts written in Classical Greek. This system automatically extracts keywords from Greek texts using term frequency.

Our approach differs from the ones described in the previous paragraphs in that it tries to identify specific information based on rules and on vocabularies of rule activation terms. Also a technique for recognizing term relationships is explored. Additionally classic IR techniques such as suffix and stopword removal [1] are utilized and evaluated in Greek texts.

3. SYSTEM OVERVIEW

The relevant work done so far, focus mainly on English text neglecting other languages, which are more demanding and challenging in terms of recognition of patterns. In languages like Greek the same information may appear in many different forms, e.g. 11 Μαΐου 2005 or 11 ΜΑΙΟΥ 2005 or Μάιο 11 or 11 Μάη 2005 (11 May 2005), and still convey exactly the same meaning.

In our system, information extracting relies on rule formalisms for each identified entity. Each extraction sub-procedure ends up with one of four alternative results:

(i) identified (IDN)
(ii) possibly identified (PDN)
(iii) not identified (NDN)
(iv) not applicable (NA)

Strong rule paths produce IDN results while weak rule paths end up in PDN. Strong rules are those which definitely identify the information that accurately falls into one of the known and well defined patterns. Weak rules are those who rely on probability and heuristic methods to infer the data.

Failing to identify some entity may be due to one of two reasons: *i.* A rule activates but it fails to complete, so the data is not identified because of our system's inability. These cases, denoted as NDN, could be used for retraining the system and eventually improve mining of data.

ii. The detection of an entity is not possible because it does not exist in the announcement. For example in preliminary announcements the exact conference's date is not yet decided. So NA, adopted by Morrisey's work [10], denotes nonappearance of the hunted piece of information. NDN and NA are preferred over null as they provide the system with different semantics which could be utilized for improving the system's functionality and the searching capabilities.

The extracted data form an XML file based on a short DTD. That way data can be presented in many different forms and utilized by other applications. In order to construct rules that will enable the successful extraction of the desired facts, we examined 25 text files, a small part of our collection consisting of 145 meeting announcements. This analysis allowed us to realize the different patterns the desired data follow and construct the rules. The remaining 120 call for papers were used in the evaluation.

3.1 Text Normalization

From the analysis of the textual data it was considered necessary to normalize the data first. Words are capitalized and accents or other marks are removed. In addition, simple suffix removal techniques were applied. The primitive Greek stemmer, which is analytically described in [8] removes final Greek sigma and transforms some endings such as "ει" and "ηκε" to "ω" among other mild transformations. It has been proved that the factors described in the previous paragraph influence searching of the Greek Web space as well [6, 7].

Abbreviations were automatically replaced by their full form. For example, month names appear abbreviated quite often, e.g Jun (Ιουν) stands for June (Ιούνιος). As a final normalization point, multiple spaces, html tags and other elements, which are not useful at this first version of the system, are removed. We should indicate though that html tags could prove significant especially in correctly identifying the title and the thematic area, as they provide structure to the information.

The normalization procedure leads to fewer rules and vocabulary entries, thus to less execution time and greater success in the mining process. In English text normalization procedure is simpler as there are no differences between upper and lower case forms, there are no inclinations of verbs and nouns (apart from minor differences between singular and plural forms) and accent marks are absent unlike in Greek.

3.2 Title extraction

Extraction of the title of a conference is based on heuristic rules. The basic idea is that titles appear on the top part of an announcement and they follow a "title" format, i.e. words are in capital letters or start with a capital letter, etc. Obviously normalization should be done after the identification of title as the form of words plays an important role here. Another rule employed is based on the surrounding text and in keywords, like conference, symposium, congress and meeting. As we will see in the evaluation section title identification is quite successful, though some extracted titles are truncated.

3.3 Keyword extraction

Correct identification of the title is also important for classifying the meeting. Classification means the detection of some keywords which describe the meeting. At the moment we base the classification on two techniques. We try to identify sort list of terms by discovering terms such as "conference topics".

Furthermore we explored a technique for constructing pairs of terms describing the conference. This technique is based on co-occurring terms [9]. We define co-occurrence of two terms as terms appearing in the same Web page. If two terms co-occur in many pages, we can say that those two have a strong relation and the one term is relevant to the other. Using words from the top part of an announcement we construct a list of pairs of neighboring terms. Then we try to measure the co-occurrence of these pairs. This co-occurrence information is acquired by the number of retrieved results of a search engine using the coefficient measure $r(a, b) = |a \cup b| / (|a| + |b| - |a \cup b|)$. With $|a|$ we symbolize the number of documents retrieved when we search using term a. Similarly $|b|$ is the number of documents relevant to term b and $|a \cup b|$ is the number of pages containing both terms. The co-occurrence is measured for every pair of terms and the top results are kept, based on a fixed cut off value. So if a conference is about New Technologies in Adult Education "in" is removed and the pairs "New Technologies", "Technologies Adult", "Adult Education" are formed. Then these pairs along with the terms "New", "Technologies", "Adult", "Education" are searched in the Web and the coefficient measure of the term pairs is decided.

Although our first heuristic approach performed well the second technique produced several "bad" instances among some useful two-term keywords. For example in a conference about "Educational Software" the keywords "Educational Games" were produced, which is acceptable and was not stated explicitly in the announcement, but the bizarre keyword "Adult Software" was also produced. Clearly this technique, although promising, needs certain refinements so as to be useful.

3.4 Extraction of dates

3.4.1 Conference's date

The first step in the identification of dates is the construction of a suitable vocabulary containing the normalized month terms that will activate the rules for the extraction of the conference's date. The identification of the date is based on a simple observation. The latest dates, appearing in a call for papers, are most probably the event's start and end dates. Our purpose is to recognize both start and end dates. For example from a date 11-13 June 2005 we extract 11 June 2005 as the start date and 13 June 2005 as the end date.

The date detection procedure initiates when a month or a full date (e.g. 12/05/2006) is found in the text. In that case we first check the succeeding words until the end of the sentence and then the preceding words until the beginning of the sentence. This search aims at identifying the day and the year of the conference and keywords which verify that it is actually the meeting's date. Thus the system needs to be able to keep information preceding and succeeding the rule activation keyword. If more than one date or date range is discovered then the system searches for appropriate keywords.

Rules are a set of *If then else* and *sub ifs*. Document is processed line by line and term by term. At the end of the rule formalism the result is stored in the XML repository. A simplified part of the date extraction procedure in pseudo code is shown below.

```
While not eof and date not identified do
    Separate current line to terms
    While not eof term set do
        Look up Vocabulary
        If month name is found then
            Scan Previous Terms
            Scan Next Terms
            If ... then
                ...
            Else if ... then
                ...
            End
        End
    End
End
Update conference XML Repository accordingly
```

3.4.2 Submission date

Submission date is trickier than the event's date as is absent in many cases, especially in short announcements. This procedure is complimentary to the previous one as dates which are denoted as meeting's start and end dates should not be checked again. After the extraction of a proper date the surrounding text is scanned for words like deadline (υποβολή), or other synonyms. Clearly these rules are domain dependant and have a high error probability. This procedure ends up mostly with one of the codes PDN, NDN, NA.

3.5 Location extraction

For extracting the location we constructed and utilized an ontology with the major Greek cities and the prefecture in which they belong. This listing also models bordering city and county relations. A city's name will trigger off the rules for the identification of the desired information. It was proved that normalization of locations names is absolutely essential as they appear in many different forms, e.g. Αθήνα, Αθηνών, Αθήνας (Athens). One problem in the identification of the location arises when a conference is co-organized by more than one institutions. In this case many locations co-exist. Mining is then based on the surrounding context or on the location's tf (term frequency) measured in the whole announcement. If a strong decision is made then the procedure ends up, whereas when a weak decision is made the procedure initiates again when new activation terms appear up.

4. SYSTEM ARCHITECTURE

The system is implemented in Java using JSP and Servlets. For processing the textual information a version of the jflex utility (http://jflex.de) is used. A flowchart of the system is shown in figure 1. The conference announcement is submitted either as a url pointing to an html file or it pasted in a text box on the system's web page.

The extracted information is stored in an XML file which is then accessible by the retrieval component of the system. This component, which is currently under development, dynamically forms an index of the processed conferences based on the information found in the XML repository. When projected to the client's browser conferences are classified as open or past and they are categorized based on their date. This tool will also allow multiriteria retrieval of conferences, such as "show me conferences in Athens or near Athens which are about Web mining and will take place this summer". Supporting these queries will be based on the location knowledge base and on the month dictionary.

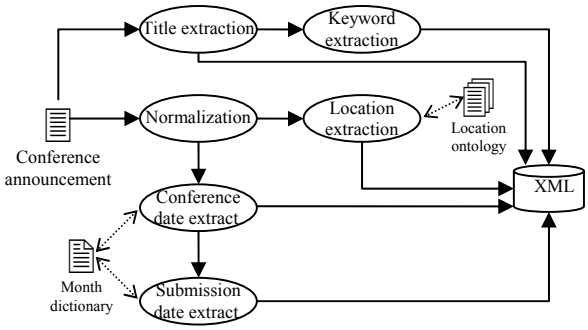

Figure 1. Flowchart of the extraction procedure.

5. EVALUATION

The performance of an information extraction system can be measured using Precision (P) and Recall (R) [13], as in Information Retrieval systems. Precision measures the ratio of the correctly extracted information against all the extracted information. Recall measures the ratio of the correct information extracted from the texts against all the available information. Despite the diversity of the collection the system works adequately well and the employed rules achieve high rates of precision and recall, especially in the attributes where a dictionary is used.

Table 1. Precision and Recall of the extraction procedure

	Title	Keyword	Conf date	Subm date	Location
Correct	77	39	107	89	110
Wrong	29	65	8	19	7
Not extra	14	16	5	12	3
Precision	72,64%	37,50%	93,04%	82,41%	94,02%
Recall	64,17%	32,50%	89,17%	74,17%	91,67%

The results of the evaluation are summarized in table 1. As expected, title and keywords show a higher error percentage. Clearly more sophisticated rules are needed. A possible solution would be the exploitation of tagging information and the usage of lexicons which model domain relationships as well. It should be noted that partially extracted titles, even those with only one not identified word, were accounted as erroneously extracted. So with slight improvements we can achieve higher precision and recall. Date and location rules achieve high precision and recall scores. Their extraction is relying on specific word lists and they follow better structured patterns.

In order to realize the effects of normalization and to get an indication of the additional difficulties posed in Greek we evaluated the system's performance, on date, submission date and location extraction, without extensive normalization. That is words were only capitalized and short forms replaced by their full forms. The evaluation showed that system's precision reduced by more than 30%. It could be argued that in this case more rules should be employed in order to achieve higher precision. While this could be partially true, we need to take into account that more rules means increased execution time as more searches are needed and a higher error probability as more heuristics and weak rules will be employed.

A final evaluation task was performed utilizing Google. A set of five queries concerning specific locations and a second set concerning dates consisting of months and years were run in our collection using Google. Then we evaluated the precision of each query (tables 2 and 3). Clearly Google retrieves many irrelevant files which diminish precision and recall. This is because every file containing the query terms or one of them is retrieved. Furthermore, announcements where terms appear in different forms than the requested ones are not retrieved. In our tool vocabularies act as thesauri as well allowing retrieval of meetings where locations or month names appear in another form or inclination. Of course tables 2 and 3 show an initial estimation. A more thoroughly designed evaluation is needed with more queries to safely reach useful conclusions.

Table 2. Precision and Recall of location queries in Google

Location	Precision	Recall
Query 1	57,50%	76,00%
Query 2	42,86%	83,33%
Query 3	77,78%	83,33%
Query 4	55,88%	64,29%
Query 5	50,00%	65,71%

Table 3. Precision and Recall of date queries in Google

Date	Precision	Recall
Query 1	42,31%	60,00%
Query 2	32,14%	52,38%
Query 3	43,75%	75,00%
Query 4	40,63%	50,00%
Query 5	37,50%	54,29%

6. SYNOPSIS AND FUTURE WORK

This paper presents an under development system which automatically extracts data from Greek conference announcements. Five categories of data are mined utilizing various techniques and approaches. For the first two categories rules are based on text's position, on context surrounding the information and on a coefficient measure. The last three types of data are mined with the utilization of lexicons which contain rule initiation terms. Then the surrounding text is again exploited. It was shown that simple removal of endings and accents and other adjustments, specific to Greek language, improve the extraction procedure and lead to increased Precision and Recall and to less elaborate rules. Vocabularies act as thesauri permitting retrieval of text where terms appear in different forms than the requested ones.

However more work needs to be done in order to achieve high rates of precision. Tagging and formatting information should be utilized in the identification of complex textual information. Metadata and link tracking, in the case of html or xml files, could be utilized. Links usually point to more detailed announcements in which all the data are applicable. Domain vocabularies are necessary in order to identify classification terms. Also, when fully developed, the system should be evaluated against the existing manual or semi automatic conference engines so as to realize all the advantages of our automated system.

Ultimately we aim at building a more complicate system which continually scans the Web to find future conferences, symposiums and congresses. From this combined system XML descriptions of the events could be produced which in turn could be utilized in automatically constructing conference announcement indices. These Web pages will be thematically sorted and automatically and regularly updated, with advanced searching capabilities thus enabling users to find everything in one place. Many issues related to information retrieval are open in the intended system, from categorization of events to summarization and to multicriteria and multilingual retrieval.

7. REFERENCES

[1] Baeza-Yates, R., Ribeiro-Neto, B. *Modern Information Retrieval*. Addison Wesley, ACM Press, New York, 1999.

[2] Chowdhury, G. G., Lynch, M. F. Automatic interpretation of the texts of chemical patent abstracts, part 1: lexical analysis and categorisation. *Journal of Chemical Information and Computer Science*, 32, (1992), 463-467.

[3] Cowie, J, Lehnert, W. Information extraction. Communications of the ACM, 39, (1996), 80-91.

[4] Gaizauskas, R., Robertson, A. Coupling information retrieval and information extraction: a new text technology for gathering information from the web. In *Proceedings of the RIAO'97 Conference*, (Canada), 1997, 356-370.

[5] Han, H., Giles, L. C., Manavoglu, E., Zha, H., Zhang, Z., Fox, E.. Automatic document metadata extraction using support vector machines. In *Proceedings of the ACM IEEE Joint Conference on Digital Libraries*, 2003, 37-48.

[6] Lazarinis, F. Do search engines understand Greek or user requests "sound Greek" to them? In *Open Source Web Information Retrieval Workshop* (in conjunction with IEEE/WIC/ACM International Conference on Web Intelligence & Intelligent Agent Technology, France), 2005, 43-46.

[7] Lazarinis, F. Evaluating user effort in Greek web searching. In *Proceedings of the 10th PanHellenic Conference in Informatics* (University of Thessaly, Greece), 2005, 99-109.

[8] Lazarinis, F. Old information retrieval techniques meet modern Greek Web searching. In *Data Mining and Information Engineering Proceedings, 2006* (accepted)

[9] Mori, J., Matsuo, Y., Ishizuka, M., Faltings, B. Keyword extraction from the web for foaf metadata. In *1st Workshop on Friend of a Friend, Social Networking and the Semantic Web* (1-2 September 2004, Galway, Ireland), 2004.

[10] Morrissey, M. J. *A treatment of imprecise data and uncertainty in information systems*. PhD Thesis, Department of Computer Science, University College, Dublin, Ireland, 1987.

[11] Petasis, G., Paliouras, G., Karkaletsis, V., Spyropoulos, C. Resolving part-of-speech ambiguity in the Greek language using learning techniques, In *Proceedings of the ECCAI Advanced Course on Artificial Intelligence* (ACAI, Chania, Greece), 1999.

[12] Postma, G. J., Van der Linden, J. R., Smits, J. R. M., Kateman, G. TICA: a system for the extraction of analytical chemical information from texts. In Karjalainen E J (ed) *Scientific Computing and Automation*. Elsevier, Amsterdam, 1990, 176-181.

[13] Robertson, S. E. The parameter description of retrieval systems: overall measures. Journal of Documentation, 25, 1969, 93-107.

[14] Rydberg-Cox, A. J. A prototype multilingual document browser for ancient Greek texts. *The New Review of Hypermedia and Multimedia*, 7(1), 2002, 103-113.

Google-based Information Extraction

Finding John Lennon and Being John Malkovich

Gijs Geleijnse Jan Korst Verus Pronk
Philips Research
Prof. Holstlaan 4
5656 AA, Eindhoven, the Netherlands
{gijs.geleijnse,jan.korst,verus.pronk}@philips.com

ABSTRACT

We discuss a method to extract information from text fragments found with a search engine. We populate an ontology using hand-crafted domain-specific relation patterns and a class-dependent rules to recognize instances of the classes. The algorithm uses the instances for one class found in the Google excerpts to find instances of other classes. The work is illustrated by two case studies. The first involves the population of an ontology in the movie domain. The second is a search for famous people and the collection of their biographical entries such as nationality and profession.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: Linguistic Processing; H.3.3 [**Information Search and Retrieval**]: Query formulation

General Terms

Information extraction, Search engines, World Wide Web

1. INTRODUCTION

Suppose we are interested in *the countries where Burger King can be found, the Dutch cities with a technical university* or perhaps *the way to Amarillo.* For such diverse information needs, the World Wide Web in general and a search engine in particular can provide a solution. However, current search engines retrieve web pages, not the information itself[1]. We have to search within the search results in order to acquire the information. Moreover, we make implicit use of our knowledge (e.g. of the language and the domain), to interpret the web pages.

In this paper, we present an algorithm that – given a domain of interest – extracts, structures and combines information obtained from an internet search engine.

[1]The question-answering services of `http://www.google.com` and `http://www.askjeeves.com` do not provide answers to these (simple) questions.

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

The extracted information can, e.g. be used by recommender systems to acquire additional metadata. This metadata can be used to make meaningful recommendations for music or TV programs. For example, suppose a user has expressed a preference for TV programs relating to Italy. The recommender system will be able to recognize regions as Tuscany and Veneto and cities as Milano and Florence using extracted information. Occurrences of such terms in a program guide will mark a program as relevant. Likewise, if the user has expressed a preference for TV programs relating to photography the system will be able to recognize the names of famous photographers as Cartier-Bresson and Moholy-Nagy.

This paper is organized as follows. After defining the problem and discussing related work in the next parts of this section, we present an algorithm to populate a given ontology in Section 2. Section 3 handles a case study on populating a movie ontology. In Section 4 we present work on finding famous people and their biographical data. Finally, Section 5 handles the conclusions and future work.

1.1 Problem definition

The semantic web community [2] is providing standards for machine readable information on the web. The languages RDF(S) and OWL are developed for this purpose by the World Wide Web Consortium[2]. Dedicated reasoners are created for ontology-based question-answering services. As such, these reasoners are able to provide answers to information demands like the above, given a sufficiently populated ontology.

For our purposes we define an ontology as follows:

Definitions. Reference ontology O is a 4-tuple (C, I, P, T), where

$C = (c_0, c_1, ..., c_{N-1})$, an ordered set of N classes,
$I = (I_0,\ I_1,\ ...,\ I_{N-1})$, with I_j, $0 \leq j < N$,
the set of instances of class $c_j \in C$,
$P = (p_0, p_1, ..., p_{M-1})$, a set of M binary relations
on the classes, with $p_i : c_{i,0} \times c_{i,1}$,
$0 \leq i < M$, and $c_{i,0}, c_{i,1} \in C$, and
$T = (T_0,\ T_1,\ ...,\ T_{M-1})$, is a set of instances of
the relations in P, with $T_j = \{(s, o) \mid p_j(s, o)\}$
for each $j,\ 0 \leq j < M$
and $s \in I_{j,0}$ (an instance of $c_{j,0}$)
and $o \in I_{j,1}$ (instance of $c_{j,1}$).

A *partial ontology* of O is defined as $O' = (C, I', P, T')$, where

[2]`http://w3c.org/`

$I'_j \subseteq I_j$ for all j, $0 \leq j < N$,
$T'_j \subseteq T_j$ for all j, $0 \leq j < M$ and
$(s, o) \in T'_k \Rightarrow s \in I'_i \wedge o \in I'_j$ for some i, j, k. □

Popular search engines currently only give access to a list of possibly interesting webpages. A user can get an idea of relevance of the pages presented by analyzing the title and an excerpt presented. When a user has send an accurate query to the search engine, the actual information required by the user can already be contained in the excerpt.

We are interested whether the data in the excerpts presented by a search engine is sufficient to extract information. With the definitions presented above, we formulate the information extraction problem as an ontology population problem:

Problem. Given a partial ontology O', extend O' to some O'' that maximizes the precision and/or recall using Google excerpts only. □

We define *precision* and *recall* as measures of a class $c_i \in C$: $precision(c_i) = \frac{|I_i \cap I''_i|}{|I''_i|}$ and $recall(c_i) = \frac{|I_i \cap I''_i|}{|I_i|}$. Similar measures can be formulated for relations p_j.

1.2 Related work

Information extraction and ontologies are two closely related fields. For reliable information extraction, we need background information, e.g. an ontology. On the other hand, we need information extraction to generate broad and highly usable ontologies. An overview on ontology construction and usage and ontology learning from structured and unstructured sources can be found in [16, 6].

In the nineties, the Message Understanding Conferences focussed on the recognition of named entities (such as names of persons and organizations) in a text [7]. This work is mostly based on rules on the syntax and context of such named entities. For example, two capitalized words preceded by *mr.* will denote the name of a male person. Research on named entity recognition is continued for example in [22]. Hearst [14] propagated another application of the use of patterns, viz. to identify relations between instances.

The growth of the Web as well as the matureness of current search engines have given pattern-based research a new incarnation. On the one hand since the use of such simple techniques has proven to be successful a approach in information extraction from large corpora [10]. On the other hand since the use of patterns in queries to a search engine are a convenient method to access relevant documents and identify phrases containing the required information [21].

In early work, Brin identifies the use of patterns in the discovery of relations on the web [4]. He describes a website-dependent approach to identify hypertext patterns that express some relation. For each web site, such patterns are learned and explored to identify instances that are similarly related. In [1], the system described in [4] is combined with a named-entity recognizer. This Snowball-system also identifies instances with the use of the named-entity recognizer.

In [19], a method is described to extract information from the web. Ravichandran identifies patterns that express some relation. Alike our approach, he uses patterns in queries to a search engine to find information. A difference is that Ravichandran performs a constant number of queries, where we dynamically construct queries with the instances found.

KnowItAll is a hybrid named-entity extraction system [11] that finds lists of instances of some class from the web using a search engine. The focus is rather on the identification of instances of classes than on populating instance-pairs of relations between the classes. It combines hyponym patterns [14] and learned patterns for instances of some class to identify and extract named-entities. Moreover, it uses adaptive wrapper algorithms [9] to extract information from html markup such as tables.

Cimiano and Staab [8] use Google to identify relations between concepts. Contrary to our method, they test a hypothesis rather than to extract new knowledge. For example, they test whether the phrase *the nile is a river* returns enough Google hits to accept the relation *is a(Nile, river)*.

Automated part of speech tagging [3] is a useful technique in term extraction [12]. Here, terms are extracted with a predefined part-of-speech structure, e.g. an adjective-noun combination. In [18], methods are discussed to extract information from natural language texts with the use of both term identification and Hearst patterns. The use of such techniques can be integrated in the information extraction algorithm proposed in this paper.

2. SOLUTION APPROACH

Information extraction from the web can be separated into two concerns. On the one hand we need some method to index and/or retrieve relevant webpages. On the other hand we need techniques to process the retrieved web content.

We focus on the second concern and use a state-of-the-art search engine to retrieve relevant web data. We choose the currently popular search engine Google[3] [5] and base our work on the excerpts returned by Google after submitting a query.

When using a search engine, we have to deal with the following restrictions.

1. The current search engines return a maximum of 1,000 results per query.
2. We want to perform as few queries to a search engine as possible to limit the use of its services.

We therefore need accurate queries, for which we can expect the search engine to return relevant excerpts.

For example, if we are interested in the father of *Christaan Huygens*, we can simple query "*Christiaan Huygens*". However, the excerpts to this query are less likely to contain the specific information of interest (*Constantijn Huygens*). Moreover if these excerpts do contain this information, we need elaborate techniques to identify both the instance and the relation.

We can also query the expression "*was the father of Christiaan Huygens*". The excerpt results for this query are likely to contain the information needed. Moreover, we only need techniques

[3] `http://www.google.com`

to identify the instance (i.e. *Constantijn Huygens*). The instance pair of the relation we then get on a bargain.

We thus choose to use patterns expressing the relations in O' as part of the queries. When we combine a relation pattern with an instance into a query, we expect the search engine to return relevant excerpts. Moreover, using the relation pattern we can extract instance-pairs of the relation instantly from the excerpts.

Our method assumes a partial ontology O' of an arbitrary knowledge domain. Since we use an instance each time we query Google, initially at least one of the sets I'_j must be non-empty. We do not consider this a disadvantage, since the creator of the ontology is expected to have some knowledge of the field.

In Section 2.1 we focus on the identification of relation patterns. Section 2.2 handles the identification of instances of a class from the excerpts. The process of combining instances and patterns into queries is discussed in 2.3. We combine these strategies into the ontology population algorithm as found in Section 2.4.

2.1 Identifying relation patterns

For relation p_k, defined on $(c_{k,0}, c_{k,1})$, in the partial ontology O', we have to identify explicit natural language formulations of this relation. We are thus interested in patterns $\mathcal{P}_k$ of the form "[$c_{k,0}$] expression [$c_{k,1}$]" [4], that express the relation p_k in natural language. Such patterns have to meet two criteria:

(Precision.) Preferably, the phrase is unambiguous, i.e. the probability that the terms found do not belong to the intended class must be small. For example, consider the relation *place of birth(Person, City)*. The pattern [*Person*] *was born in* [*City*] is not an unambiguous representation of this relation, since [*Person*] *was born in* can precede a date or the name of a country as well.
(Recall.) The pattern must frequently occur on the Web. Rare patterns are not likely to give much search results when querying such a pattern in combination with an instance.

Suitable formulations can be found by observing how instances of the related classes are connected in natural language texts. For example, if we are interested in populating *plays for(player, team)*, we can identify this set of patterns: $\mathcal{P}_{plays\_for}$ = { "[*team*]-*player* [*player*]", "[*player*] ([*team*])", "[*player*] *signed for* [*team*]", "[*team*] *substituted* [*player*] *for* [*player*]" } .

In this work, we select the relation patterns manually. Our current work however involves the automatic identification of effective patterns [13], which are patterns that are likely to give useful results when using them as queries.

2.2 Instance identification

A separate problem is the identification of terms in the text. An advantage is that we know the place in the text by construction (i.e. either preceding or following the queried expression). A disadvantage is that each class requires a different technique to identify its instances. Especially terms with a less determined format, such as movie titles, are hard to identify. We therefore design recognition functions f_i for each class.

For these functions f_i, we can adopt various techniques from the fields of (statistical) natural language processing, information retrieval and information extraction. A regular expression that describes the instances of class c_i can be a part of the function f_i. The user may also think of the use of part of speech tagging [3]. We note that the HTML-markup can be of use as well, since terms tend to be emphasized, or made 'clickable'.

After extracting a term, we can perform a *check* to find out whether the extracted term is really an instance of the concerning class. We perform this check with the use of Google. We google phrases that express the term-class relation. Again, these phrases can be constructed semi-automatically. Hyponym patterns are candidates as well for this purpose. A term is to be accepted as instance, when the number of hits of the queried phrase is at least a certain threshold.

When we use such a check function, we can allow ourselves to formulate less strict recognition functions f_i. That is, false instances that are accepted by f_i, are still rejected as an instance by the use of the check function.

2.3 Formulation of Google-queries

When we have chosen the sets of relation p_k, we can use these to create Google queries. For each "[$c_{k,0}$] expression [$c_{k,1}$]" pattern, we can formulate two Google queries: "[$c_{k,0}$] expression" and "expression [$c_{k,1}$]". For example, with the relation *was born in* and instances *Amsterdam* and *Spinoza*, we can formulate the queries "*Spinoza was born in*" and "*was born in Amsterdam*".

This technique thus allows us to formulate queries with instances that have been found in results of prior queries.

2.4 Sketch of algorithm

Per relation, we maintain a list of instances that already have been used in a query in combination with the patterns expressing this relation. Initially, these lists are thus empty.

The following steps of the algorithm are performed until either some stop criterion is reached, or until new instances and instance-pairs no longer can be found.

- **Step 1:** Select a relation p_k on $c_i \times c_j$, and an instance v from either I_i or I_j we have not yet used in a query.
- **Step 2:** Combine the patterns expressing $\mathcal{P}_k$ with v and send these queries to Google.
- **Step 3:** Extract instances from the excerpts using the instance identification rules for the class of v.
- **Step 4:** Add the newly found instances to the corresponding instance set and add the instance-pairs found (thus with v) to $T'_{(i,j)}$.
- **Step 5:** If there exists an instance that we can use to formulate new queries, then repeat the procedure.

Note that instances of class c_i learned using the algorithm applied on relation p_k on $c_i \times c_j$ can be used as input for the algorithm applied to some relation p_l on $c_i \times c_h$ to populate the sets I'_h and $T'_{(i,h)}$.

[4] We use the [c_i] notation to denote a variable instance of class c_i

3. POPULATING A MOVIE ONTOLOGY

For our first case study, we have constructed a small partial ontology on the movie domain. It is defined as

$O'_{movie} = ($ (*Director, Actor, Movie*) ,
({ *Steven Spielberg ,*
Francis Ford Coppola }, $\emptyset, \emptyset$) ,
(*acts in(Movie,Actor),*
director of(Movie,Director)) ,
($\emptyset, \emptyset$)).

We thus only identify three classes, of which only the class *Director* has instances. Using our method, we want to find movies directed by these directors. The movies found are used to find starring actors, where those actors are the basis of the search for other movies in which they played, etc. The process continues until no new instances can be found.

Relation patterns. This small ontology contains two relations, *acts in* and *director of*. For these relations, we have manually selected the sets of patterns:
$\mathcal{P}_{acts\_in}$ = {"[*Movie*] *starring* [*Actor*],[*Actor*] *and* [*Actor*] " } and $\mathcal{P}_{director\_of}$ = {"[*Director*]*'s* [*Movie*] " , "[*Movie*]*, director:* [*Director*]" }.

Instance identification. We identify a term as a $Movie$ title, if it is placed in a text between quotation marks. Although this may seem a severe restriction, in practice we can permit to loose information contained in other formulations since each Google query-result gives much redundant information. So, if a movie title is placed between quotation marks just once in the Google results, we are able to recognize it.

A person's name (instances of the classes $Director$ and $Actor$) is to be recognized as either two or three words each starting with a capital.

Another feature of the recognition function is the use of lists with tabu words. If a tabu word is contained in an expression, we ignore it. We use a list of about 90 tabu words for the person names (containing words like 'DVD' and 'Biography'). For the movie titles we use a much shorter list, since movie titles can be much more diverse. We have constructed the tabu word lists based on the output of a first run of the algorithm.

We *check* each of the extracted candidate instances with the use of one of the following Google-queries: "The movie [*Movie*]", "[*Actor*] plays", or "[*Director*] directed". A candidate is accepted, if the number of Google-results to the query exceeds a threshold. After some tests we choose 5 as a threshold value, since this threshold filtered out not only false instances but most of the common spelling errors in true instances as well.

Formulation of Google-queries. The relation patterns lead to the following set of Google-queries: {"[*Director*]*'s*", "[*Movie*] *starring*", "[*Movie*] *director*" , "*starring* [*Actor*]"}. We have analyzed the first 100 excerpts returned by Google after querying a pattern in combination with an instance.

3.1 Results

We first ran the algorithm with the names of two (well-known) directors as input: *Francis Ford Coppola* and *Steven Spielberg*. Afterwards, we experimented with larger sets of directors and small sets of beginning directors as input.

An interesting observation is that the outputs are independent of the input sets. That is, when we take a subset of the output of an experiment as the input of another experiment, the outputs are the same, modulo some small differences due to the changes in the Google query results over time.

We have found 7,000 instances of the class Actor, 3,300 of Director and 12,000 of Movie. The number of retrieved instances increases, about 7%, when 500 query results are used instead of 100.

Precision. When we analyze the precision of the results, we use the data from the Internet Movie Database (IMDb)[5] as a reference. An entry in our ontology is accepted as a correct one, if it can be found in IMDb. We have manually checked three sequences of 100 instances (at the beginning, middle and end of the generated file) of each class. We estimate a precision of 78 %. Most misclassified instances were misspellings or different formulations of the same entity (e.g. "Leo DiCaprio" and "Leonardo DiCaprio"). In the future, we plan to add postprocessing to recognize these flaws. We can analyze the context (e.g. when 2 actors act in the same set of movies) and use approximate string matching techniques to match these cases.

Likewise, we have also analyzed the precision of the relations, we estimate the precision of the relation between movie and director around 85 %, and between movie and actor around 90%.

Recall. The number of entries in IMDb exceeds our ontology by far. Although our algorithm performs especially well on recent productions, we are interested how well it performs on classic movies, actors and directors. First, we made lists of all Academy Award winners (1927-2005) in a number of relevant categories, and checked the recall (Table 1).

CATEGORY	RECALL
Best Actor	96%
Best Actress	94%
Best Director	98%
Best Picture	87%

Table 1: Recall of Academy Award Winners

IMDb has a top 250 of best movies ever. The algorithm found 85% of them. We observe that results are strongly oriented towards Hollywood productions. We also made a list of all winners of the Cannes Film Festival, the 'Palme d'Or'. Alas, our algorithm only found 26 of the 58 winning movies in this category.

4. EXTRACTING INFORMATION ON FAMOUS PEOPLE

The second case study aims at extracting a long list of famous persons and in addition extracting for each of them biographical information such as nationality, period of life, and profession. Using this additional information, we can create sublists of e.g. 17th-century Dutch painters. The information extraction is carried out in two phases. First a long list of famous persons is extracted, and secondly, additional information on these persons is gathered.

[5] `http://www.imdb.com`

4.1 Relation patterns and query formulation

It has been observed by e.g. [20] that a surface pattern as "*Wolfgang Amadeus Mozart (*" is very successful to determine the year of birth of in this case Mozart, as the open bracket will be often followed by the period of life of the person (in this case: 1756-1791). We decided to use this observation but in a different fashion. Instead of looking for the year of birth of a given person, we use year intervals that possibly relate to the lifetime of a person to find famous persons. More precisely, we issued all year intervals "(y_1 - y_2)" as queries to Google, with $y_1 \in [1000..1990]$, $y_2 - y_1 \in [15..110]$ and $y_2 \leq 2005$. In other words, we search for persons who were born during the last millenium and who died at an age between 15 and 110. Note that, in this way, we will only find persons that already passed away.

4.2 Instance identification

For each of these issued queries, we scanned the at most 1000 excerpts that Google returned. In each of these excerpts, we determined the first occurence of the queried pair of numbers. Since Google ignores non-alphanumeric characters, the queried pair of numbers may also occur as y_1, y_2 or as y_1/y_2. If the queried pair of numbers is in the intended context $(y_1 - y_2)$, i.e. if they are surrounded by brackets and seperated by a hyphen, then the words directly preceding this first occurrence are stored for later analysis, to a maximum of six words. In this way, we obtain for each queried pair of numbers up to 1000 short text fragments that potentially contain person names. In addition, for each of the stored text fragments, we remove potential pre- and suffixes that normally cannot be part of a name. For example, we delete all words that precede a full stop (except when preceded by a single capital letter), a colon, or a semicolon. In addition, of words consisting of upper-case letters only we transform the upper-case into lower-case letters, except for the first one (with some specific exceptions concerning ordinal numbers of kings, queens, etc., composite names including hyphens or apostrophes, and Scottish and Irish names). This results in a set of candidate names.

The *check* phase consists of two filtering steps: one to filter out non-person names and one to filter out multiple variants of a single person name. These steps are next discussed in more detail.

Not all text fragments we have found in the extraction phase will be person names. Typically, historic periods, art styles, geographic names, etc. can also directly precede a time interval. Table 2 illustrates the difficulties in discriminating between person names and other text fragments. We note that *West Mae* is an inversion of the person name *Mae West* and that *Napoleon Hill* refers to a person as well as to a geographic location in the state Idaho (USA).

PERSON NAME	NON-PERSON NAMES
Art Blakey	*Art Deco*
West Mae	*West Virginia*
Amy Beach	*Miami Beach*
HP Lovecraft	*HP Inkjet*
Napoleon Hill	*Napoleon Hill*

Table 2: Some examples to illustrate the difficulties in discriminating between persons names and other text fragments.

To filter out non-person names, we first constructed from dedicated websites a long list of the most common first names (boy's and girl's names). If a text fragment starts with such a name, then this is a strong indication that the text fragment is a person name. In addition, we constructed a long list of suspect words that typically do not occur in person names, as follows. From the many excerpts that we gathered with the year interval queries we extracted all words, counting how often they occur with a capital and without a capital. If a word occurs most often without a capital, and it is not a special word as 'van', 'de', or 'la', then it is added to the long list of suspect words. We next apply a rule-based approach using these lists of first names and suspect words to filter out text fragments that probably do not relate to person names.

In addition to filtering out non-person names, we also want to filter out multiple occurrences of the same person name. These occurrences are caused by variations in spelling of names and errors in the lifetimes. To this end, we carried out the following filtering steps.

1. *Keeping only the last name/lifetime variants that occur most often*. For each last name/lifetime combination, we often find different variants of first names preceding it. For example, *Bach (1685 - 1750)* is preceded by, e.g., *Johann Sebastian*, *JS*, and *Johann S*. Of all these variants we only store the one that is found most often, i.e., the variant that occurs most often in the text fragments we found in the 1000 excerpts that Google returned on query *"(1685 - 1750)"*.

2. *Filtering out small variations in name*. If two names have exactly the same lifetime and the edit distance [17] between these full names is less than a given threshold, then only the variant that is found most often is kept. As threshold we use an edit distance of two.

3. *Filtering out single errors in lifetimes*. If two names are completely identical but their lifetimes differ in only the year of birth *or* the year of death, then only the variant that is found most often is kept.

Experiments indicate that in this step we reduce the candidate set of names by approximately 25%.

4.3 Ordering persons by fame

To order the persons by fame, we use Google page count (GPC) as our measure of fame. Now, the question is which query we should issue to Google to determine the GPC of a person. The query should be neither too general nor too specific.

A single person is often identified in different ways, e.g. *Johann Sebastian Bach*, *JS Bach*, *JOHANN SEBASTIAN BACH* and *Bach, Johann Sebastian* all refer to the same person. The last variant is called an *inversion*. The latter two variants can be transformed into the first variant by substituting upper-case characters by lower-case ones and by adjusting the order of first and last names. Complicating factors in the identification of inversions are (i) that a comma between last name and first names is sometimes omitted and (ii) that many first names also occur as last names. An additional complication is that the first names sometimes vary per language (e.g. Charles vs. Karel). To achieve that we are less sensitive to these variants, we use the following query to determine the GPC:

"[*last name*] ([*year of birth*] - [*year of death*])"

For kings, queens, popes, etc., we use the Latin ordinal number as last name. In this way *Charles V (1500 - 1558)*, *Carlos V (1500 - 1558)*, and *Karel V (1500 - 1558)* are all covered by query *"V (1500 - 1558)"*. Note that we assume the combination of last name and lifetime to be specific enough to uniquely identify famous persons.

4.4 Extracting additional information

The first phase, described above, resulted in a large list of famous persons that was ordered using GPC as measure of fame. For further details on this list we refer to [15]. In the next phase, we extracted additional information, such as gender, nationality, and professions. Also, we tried to retrieve related images and a few one-liners that already give a brief impression of how the person gathered fame. We extracted additional information for the top 10,000 of the list of famous persons that we obtained in the first phase. We next briefly describe how we gathered this additional material.

To acquire additional information, we again issued queries to Google of the type "*Albert Einstein was*", i.e., we used the full name of a person followed by the word *was*, where we restrict ourselves to English language pages. From the excerpts that Google returns, we extracted complete sentences that contain the query. Hence, if only a fraction of a sentence was given in an excerpt, then this fraction was simply ignored. These sentences were next used to identify specific words that indicate gender, nationality and professions.

Determining gender. We simply counted words that refer to the male gender, namely the words *he*, *his*, *son of*, *brother of*, *father of*, *man* and *men*. Likewise, we counted words that refer to the female gender, namely the words *she*, *her*, *daughter of*, *sister of*, *mother of*, *woman*, and *women*. We simply assigned the gender with the highest count.

Determining nationality. We extracted for each country from the CIA World Factbook website the country name (in conventional short form) and the corresponding adjective that indicates nationality, e.g. 'Belgium' and 'Belgian'. In addition, for some countries we added a number of additional terms relating to parts of the country, such as 'Flemish' for Belgium and 'English', 'Scottish', and 'Welsh' for the United Kingdom. To determine the nationality, we count for each country the number of word occurrences in the set of sentences, and simply assign the nationality with the highest count. So far, we did not consider country names of countries that do no longer exist, such as Prussia.

Determining professions. To determine in a similar fashion the professions of a given person, we first have to construct a list of potential professions. This list is generated as follows. We started with a hand-made list of 40 professions, that we extended automatically as follows. Using the nationalities and short list of professions we constructed queries of the form

"[*nationality*] [*profession*] and"

such as e.g. "*Dutch astronomer and*". These were issued to Google, and the resulting excerpts were analysed for additional profession names. More precisely, if in a resulting excerpt the query was succeeded by up to three words without a capital followed by a word with a capital, then these one to three words without a capital are added to the list of potential professions. This resulted in a list of approximately 2500 potential professions. This list includes qualifications that are not professions in the strict sense but are used to characterize persons, such as *free thinker*, *sex symbol*, and *feminist*. Table 3 give the top-40 of the professions found, ranked by the number of times that these professions were found in the excerpts.

PROFESSIONS			
philosopher	1275	designer	222
composer	804	scientist	215
mathematician	773	musician	213
poet	668	historian	210
physicist	501	inventor	208
writer	478	essayist	201
playwright	469	engineer	199
novelist	429	singer	198
sculptor	362	dramatist	186
author	352	theorist	175
critic	346	illustrator	171
astronomer	343	journalist	166
painter	329	statesman	138
politician	323	teacher	138
artist	286	mystic	133
architect	284	educator	132
director	270	theologian	127
conductor	267	physician	125
actor	261	printmaker	124
pianist	224	scholar	112

Table 3: The professions that were found most often.

As for gender and nationality, we now simply count how often each of these profession names occur in the sentences. However, instead of only selecting the one with the highest count, we here want to be able to retain multiple professions. For that reason, we select the ones that have at least a count of $0.5 \cdot c_{\max}$, where $c_{\max}$ is the score of the highest scoring profession, ordered by decreasing count.

4.5 Results

To give an impression of the results that we obtained in this case study, we present three tables. Table 4 gives the top of the persons born in the period $[1880..1889]$, Table 5 gives the top of the persons that has as their highest scoring profession either artist or painter, and Table 6 gives the top of the persons that were identified as Dutch.

Recall. To get an impression of the performance of our algorithm, we estimate the recall by choosing a diverse set of six books containing short biographies of persons whom we would expect to find in our list. For each of these books, we determined for the persons that could potentially be found by our algorithm (i.e., the persons who are born in the intended time period and have died). Of these 1049 persons, 1033 were present in our list, which is a fraction of 0.98. For further details on the chosen books we refer to [15]. We observe that the recall is close to one, for each of the six books, even for a more specialized topic as 17th century Dutch painters. Of the total 108 of these painters mentioned in one of the books, 106 were found. We note that of the 16 persons that did not appear in our list, there were 4 persons for which the books could not provide the lifetime.

For the recall of the additional information, we observe that for the 10,000 persons that we considered all were given a gender, 77% were given a nationality, and 95% were given one or more professions.

Precision. All kinds of imperfections can still be observed in our list of famous persons, such as remaining inversions, missing parts

BORN IN [1880, 1889]		
James Joyce (1882-1941)	Ireland	author
Bela Bartok (1881-1945)	Hungary	composer
Pablo Picasso (1881-1973)	Spain	artist
Anton Webern (1883-1945)	Austria	musician, composer
HL Mencken (1880-1956)	United States	author, journalist
Niels Bohr (1885-1962)	Denmark	scientist, physicist
Adolf Hitler (1889-1945)	Germany	leader
Amedeo Modigliani (1884-1920)	Italy	artist, painter
Agustin Barrios (1885-1944)	Paraguay	musician, composer
Le Corbusier (1887-1965)	Switzerland	architect
John Maynard Keynes (1883-1946)	United Kingdom	economist
Ludwig Wittgenstein (1889-1951)	Austria	philosopher
Igor Stravinsky (1882-1971)	Russia	composer
TS Eliot (1888-1965)	United Kingdom	poet
Franz Kafka (1883-1924)	Czech Republic	author
Franklin D. Roosevelt (1882-1945)	United States	president
Marc Chagall (1887-1985)	Russia	painter, artist
Martin Heidegger (1889-1976)	Germany	philosopher
Kahlil Gibran (1883-1931)	Lebanon	poet, philosopher,...
Heitor Villa-Lobos (1887-1959)	Brazil	composer

Table 4: The 20 persons born between 1880 and 1889 with the highest GPC**.**

ARTISTS/PAINTERS		
Leonardo da Vinci (1452 - 1519)	Italy	artist, scientist,...
Pablo Picasso (1881 - 1973)	Spain	artist
Vincent van Gogh (1853 - 1890)	Netherlands	artist, painter
Claude Monet (1840 - 1926)	France	artist, painter,...
Pierre-Auguste Renoir (1841 - 1919)	France	painter
Paul Gauguin (1848 - 1903)	France	painter
Edgar Degas (1834 - 1917)	France	artist, painter,...
Paul Cezanne (1839 - 1906)	France	painter, artist
Salvador Dali (1904 - 1989)	Spain	artist
Henri Michaux (1899 - 1984)	Belgium	artist, poet
Gustav Klimt (1862 - 1918)	Austria	painter, artist
Peter Paul Rubens (1577 - 1640)	Belgium	artist, painter
Katsushika Hokusai (1760 - 1849)	Japan	painter
Amedeo Modigliani (1884 - 1920)	Italy	artist, painter
JMW Turner (1775 - 1851)	United Kingdom	artist, painter
James Mcneill Whistler (1834 - 1903)	United States	artist
Rene Magritte (1898 - 1967)	Belgium	artist, painter
Henri Matisse (1869 - 1954)	France	artist
Rembrandt van Rijn (1606 - 1669)	Netherlands	artist, painter
Edouard Manet (1832 - 1883)	France	artist, painter
Herm Albright (1876 - 1944)	-	artist, engraver,...
Marc Chagall (1887 - 1985)	Russia	painter, artist
Edvard Munch (1863 - 1944)	Norway	painter, artist
Wassily Kandinsky (1866 - 1944)	Russia	artist, painter
Francisco Goya (1746 - 1828)	Spain	artist, painter

Table 5: The 25 artists/painters with the highest GPC**.**

of a name, and errors in lifetimes, although each of these occurs relatively infrequently. We concentrate on estimating the fraction of names that do not relate to persons. The corresponding precision that is obtained by the algorithm has been estimated as follows. We selected three decennia, namely 1220-1229, 1550-1559 and 1880-1889, and analyzed for each the candidate persons that were 'born' in this decennium. For the first two decennia we analyzed the complete list, for decennium 1880-1889 we analyzed only the first 1000 as well as the last 1000 names. This resulted in a precision of 0.94, 0.95, and 0.98, respectively. As the decennium of 1880-1889 resulted in considerably more names, we take a weighted average of these results. This yields an estimated precision for the complete list of 0.98 [15].

Regarding the precision of the additional information, we make the following observations. The algorithm will find at most one nationality. For persons that migrated during their lives this poses a problem. Very often, sentences with the pattern "*was born in*" occur so frequent that the country of birth determines the nationality found, such as for Henri Michaux.

Regarding the professions found, we observe that the results are usually quite accurate, even if persons have performed diverse things in life. For example, Leonardo da Vinci has been given the professions artist, scientist, and inventor, and Benjamin Franklin the professions inventor, scientist, statesman, and author. Some of the profession are ambiguous words such as *general*, *director* and *judge*. Such professions lead to less precise results. Also, *king*, *queen*, and *saint* are often used in a metaphorical sense. Errors also occur due to the mentioning of the parents' professions. For example, Edgar Degas was given *banker* as his third profession, because he was born to a banking family. Some persons remain difficult the characterize, however. Calamity Jane was given a long list of professions: *actress*, *horsewoman*, *prostitute*, *musician*, *entertainer*, *dancer*, and *hunter*. As wife of Franklin Delano Roosevelt, Eleanor Roosevelt is given the profession *president*.

Biographical entries. To get a better impression of the quality of the biographical entries, we manually checked 50 persons, evenly distributed in the top-2500. Of these 50 persons, we observed that gender, nationality and professions were all correct for 38 persons. No errors in gender were detected in any of the 50 persons. For three persons the nationality was not found. All nationalities found proved to be correct. For two persons, all given professions were wrong. For eight others, one or more given professions were incorrect, but usually the professions with the highest count were correct. In the final paper, results of a more extensive analysis will be provided

5. CONCLUSION AND FUTURE WORK

We have presented a framework algorithm for ontology population using Googled expressions. We combine patterns expressing relations and an instance of a class into queries to generate highly usable Google excerpts. From these excerpts we simultaneously extract instances of the classes and instance pairs of the relations.

The method is based on hand-crafted patterns which are tailormade for the classes and relations considered. These patterns are queried to Google, where the results are scanned for new instances. Instances found can be used within these patterns as well, so the algorithm can populate an ontology based on a few instances in a given partial ontology.

The results of the experiments are encouraging. We used simple patterns, recognition functions and checks that proved to be successful. When submitting accurate queries, the Google excerpts provide enough data to populate an ontology with good recall and precision.

The current algorithms contain two hand-crafted elements: the construction of the patterns and the identification of the instances. In current work, we are investigating methods to (semi-) automate these steps [13].

6. REFERENCES

BELGIAN/DUTCH	
Cesar Franck (1822 - 1890, B)	organist, composer, pianist
Vincent van Gogh (1853 - 1890, NL)	artist, painter
Roland de Lassus (1532 - 1594, B)	composer
Abraham Kuyper (1837 - 1920, NL)	theologian, politician
Henri Michaux (1899 - 1984, B)	artist, poet
Peter Paul Rubens (1577 - 1640, B)	artist, painter
Baruch Spinoza (1632 - 1677, NL)	philosopher
Rene Magritte (1898 - 1967, B)	artist, painter
Christiaan Huygens (1629 - 1695, NL)	astronomer, scientist,...
Rembrandt van Rijn (1606 - 1669, NL)	artist, painter
Johannes Vermeer (1632 - 1675, NL)	painter, artist
Edsger Wybe Dijkstra (1930 - 2002, NL)	computer scientist
Anthony van Dyck (1599 - 1641, B)	painter
MC Escher (1898 - 1972, NL)	artist
Antony van Leeuwenhoek (1632 - 1723, NL)	scientist
Piet Mondrian (1872 - 1944, NL)	artist, painter
Hugo Grotius (1583 - 1645, NL)	lawyer, philosopher,...
Jan Pieterszoon Sweelinck (1562 - 1621, NL)	composer, organist,...
Andreas Vesalius (1514 - 1564, B)	physician
Hieronymus Bosch (1450 - 1516, NL)	painter
Audrey Hepburn (1929 - 1993, B)	actress, princess
Ferdinand Verbiest (1623 - 1688, B)	astronomer
Desiderius Erasmus (1466 - 1536, NL)	philosopher, reformer,...
Theo van Gogh (1957 - 2004, NL)	judge, artist
Gerard Dou (1613 - 1675, NL)	painter, artist
Nicolaas Beets (1814 - 1903, NL)	king, poet, writer
Carel Fabritius (1622 - 1654, NL)	painter
Georges Simenon (1903 - 1989, B)	author
Kees van Dongen (1877 - 1968, NL)	painter
Gerardus Mercator (1512 - 1594, B)	cartographer

Table 6: The 30 Belgian/Dutch persons with the highest GPC**.**

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, 2000.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. In *Scientific American*, May 2001.

[3] E. Brill. A simple rule-based part-of-speech tagger. In *Proceedings of the third Conference on Applied Natural Language Processing (ANLP'92)*, pages 152–155, Trento, Italy, 1992.

[4] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at sixth International Conference on Extending Database Technology (EDBT'98)*, 1998.

[5] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[6] P. Buitelaar, P. Cimiano, and B. Magnini, editors. *Ontology Learning from Text: Methods, Evaluation and Applications*, volume 123 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2005.

[7] N. A. Chinchor, editor. *Proceedings of the Seventh Message Understanding Conference (MUC-7)*. Morgan Kaufmann, Fairfax, Virginia, 1998.

[8] P. Cimiano and S. Staab. Learning by googling. *SIGKDD Explorations Newsletter*, 6(2):24–33, 2004.

[9] V. Crescenzi and G. Mecca. Automatic information extraction from large websites. *Journal of the ACM*, 51(5):731–779, 2004.

[10] J. R. Curran and M. Moens. Scaling context space. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 231–238, Philadelphia, PA, 2002.

[11] O. Etzioni, M. J. Cafarella, D., A. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165(1):91–134, 2005.

[12] K. Frantzi, S. Ananiado, and H. Mima. Automatic recognition of multi-word terms: the c-value/nc-value method. *International Journal on Digital Libraries*, 3:115–130, 2000.

[13] G. Geleijnse and J. Korst. Learning effective surface text patterns for information extraction. In *Proceedings of the EACL 2006 workshop on Adaptive Text Extraction and Mining (ATEM 2006)*, Trento, Italy, April 2006.

[14] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 539–545, Morristown, NJ, USA, 1992.

[15] J. Korst, G. Geleijnse, N. de Jong, and M. Verschoor. Ontology-based extraction of information from the World Wide Web. In *Intelligent Algorithms*, Philips Research Book Series. Springer, 2006 (to appear).

[16] N. Kushmerick, F. Ciravegna, A. Doan, C. Knoblock, and S. Staab, editors. *Proceedings of the Dagstuhl Seminar on Machine Learning for the Semantic Web*. Dagstuhl, Germany, February 2005.

[17] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics*, 10:707–710, 1966.

[18] G. Nenadić, I. Spasić, and S. Ananiadou. Automatic discovery of term similarities using pattern mining. In *Proceedings of the second international workshop on Computational Terminology (CompuTerm'02)*, Taipei, Taiwan, 2002.

[19] D. Ravichandran. *Terascale Knowledge Acquisition*. PhD thesis, University of Southern California, 2005.

[20] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 41–47, Philadelphia, PA, 2002.

[21] W. R. van Hage, S. Katrenko, and A. T. Schreiber. A method to combine linguistic ontology-mapping techniques. In *Proceedings of the fourth International Semantic Web Conference*, Galway, Ireland, 2005.

[22] G. Zhou and J. Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 473 – 480, Philadelphia, PA, 2002.

Facing restrictions in questions answering

Keynote

Maarten de Rijke
Informatics Institute, University of Amsterdam
mdr@science.uva.nl

ABSTRACT

Restricted domain question answering systems attempt to answer question subject to restrictions on the document collection or on the type of questions. These restrictions provide an interesting mix of challenges and opportunities that are not fully understood. For example, when restrictions are imposed on the data, it may be impossible to apply redundancy-based answering techniques, and more and increasingly deep levels of analysis may be needed. How do we manage, organize and exploit multiple levels of annotation? And if restrictions are imposed on the types of questions to be answered, how can we exploit the restrictions for mining the documents so as to arrive at more richly structured answers? How do we present such answers?

In the talk, ongoing research efforts aimed at answering these and other questions will be presented. Specifically, I will discuss work in progress on multidimensional mark-up, medical, biographical and temporal question answering.

Authoritative Re-Ranking in Fusing Authorship-Based Subcollection Search Results

Toine Bogers Antal van den Bosch
ILK / Language and Information Science
Tilburg University, P.O. Box 90153
NL-5000 LE Tilburg, The Netherlands
{A.M.Bogers,Antal.vdnBosch}@uvt.nl

ABSTRACT

We examine the use of authorship information to divide IR test collections into subcollections and we apply techniques from the field of distributed information retrieval to enhance the baseline search results. We base an estimate of an author's expertise on the content of his documents and use this knowledge to construct rankings of the different author subcollections for each query. We go on to demonstrate that these rankings can then be used to re-rank baseline search results and improve performance significantly. We also perform experiments in which we base expertise ratings only on first authors or on all except the final authors and find that these limitations do not further improve our re-ranking method.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms

Keywords

Information retrieval, re-ranking, collection fusion, expertise, user modeling

1. INTRODUCTION

Nowadays, many retrieval systems can access a variety of sources and collections to help fulfill a user's information need. Prime examples of such systems are the meta search engines for the Web such as Dogpile [9] and Vivísimo [16], which combine the search results of other search engines and present a merged list of results to the user. Meta search engines do not attempt to perform any actual search themselves. Instead, they respond to user queries by using different search engines to increase their coverage, making use of the fact that not all search engines cover the same parts of the Web. Another type of meta retrieval system that references different sources is an information management assistant that attempts to aid a user in his or her daily writing activities, such as Watson [5] or Syskill & Webert [13]. An important step in any such meta search process is combining the results from the different collections and sources so that they can be presented to the user as if the retrieved documents were all present in one big collection. The entire process, from selecting search engines and submitting the original query to combining and presenting the results, is called *collection fusion*.

In this paper we present a novel method of improving search results within a single collection inspired by collection fusion on a larger scale. The fusion approach we use results in scores for each subcollection that we subsequently use to perform *authoritative re-ranking* of the original search results. For us to be able to apply collection fusion to enhance search in such a way, we need to identify sensible and distinct subcollections within a single collection. The obvious distinction is to equate subcollections with subtopics within the collection, identifying which is a well-researched subfield of information retrieval [15, 18]. Another, more novel way is regarding the sets of papers written by the same author as the subcollections present in the collection. The latter approach utilizes the differences in expertise between authors on certain topics to guide the selection and fusion of the different subcollections. Authors with a lot of documents about a certain topic are more likely to be an expert on that topic and also more likely to have written documents that are relevant for queries about that topic. How we determine this topical expertise is the subject of section 3.

Using author information to identify subcollections requires the parent collection to contain author labels for each document. A typical situation involves, for instance, a research lab where the workgroup is composed of around ten to fifty people and has specific interests. Workgroup members are bound by the common research focus of the workgroup, but each member also has separate interests and may be the group's expert on certain topics. Workgroup collections are also a good testcase because publications of colleagues are often considered to be more trustworthy than random books and articles found in libraries and on the WWW [1]. By adopting a wider perspective and by disregarding institutional or geographical proximity, our method can be ex-

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

tended to scientific communities, e.g. loosely knit groups of people publishing in the same journal or conference proceedings. In general, authoritative fusion can be applied to any collection of documents that represents the research output of a community of sorts, where all the author labels have been preserved. In the remainder of this paper we will refer to such a collection as a *community collection*.

2. BACKGROUND

A major challenge for meta search engines in fulfilling the user's information need is referencing the disparate sources in such a way that it approximates the performance of the hypothetical scenario if all the documents covered by the collections were *all* in a *single* collection [12]. The entire process involves not only selecting the search engines and submitting the original query to these engines but also combining and presenting the results. According to Voorhees, each of these fusion steps has its own peculiar subproblems [17]:

- *Database selection* is concerned with which subcollections to use in responding to an information need. Some collections may charge fees and searching every available collection may be too expensive in terms of resources.
- *Query translation* involves translating the original query to the different formats required by the other search engines used by the meta engine. The utilized search engines may be very different from each other, not only in the retrieval model they use, but also in the type of stemming algorithm used, the use of different stopword lists, or the query processing techniques [6].
- *Document selection* focuses on the question of what kind and how many documents the meta engine should select from the results of every search engine. One problem might be that certain documents may occur in more than one collection but are ranked differently by the search engines. Multiple occurrences of a documents need to be de-duplicated.
- *Results merging* deals with the combining the results into a coherent set to be presented to the user. Not every search engine may return the numerical values used in that specific engine's ranking and some systems might even return results that are not ranked at all.

Different solutions to the collection fusion problem have been proposed over the years. Voorhees et al. [17] propose two different approaches that both use a set of training queries. Their first solution uses relevance feedback information from these training queries to model the distributions of relevant documents over the different collections. They use these distributions to calculate the number of top-ranked documents to be selected from each collection and interleave these ranked result lists. In their second approach they cluster the set of training queries on topic, based on the overlap in relevant documents they retrieve. The new query vector is matched to the cluster centroids and the training weights of the best matching cluster are then retrieved for all collections. These weights are used to determine the number of documents to retrieve from each collection. Callan et al. [6] use a probabilistic approach in the form of an inference network to rank the different collections. They combine these collection-specific weights with the ranking scores assigned to the documents by the retrieval engines of each collection. Documents from collections with high collection weights are favored, but good documents from poor collections can also be ranked higher. Baumgarten [2] also proposes a probabilistic framework for distribution information retrieval, but one that relies less on heuristics and is better motivated theoretically.

In this paper we present a novel method of improving search results where we apply fusion techniques not on disparate collections but on a single collection. We identify different subcollections *within* the parent collection based on the sets of documents written by authors. These documents indirectly represent a subset of the expertise of each author. For each query we derive a ranking of these subcollections based on expertise and use these to re-rank the baseline search results, an approach we call authoritative re-ranking.

This type of *intra-collection* fusion lacks some of the characteristic problems of inter-collection fusion. For instance, searching all the subcollections is not very resource-intensive and since all authors within the parent collection should be considered, the problem of database selection is nonexistent. Query translation is also not an issue in our approach since we use one approach for one collection: the same stemming algorithm and stoplist is used for the baseline retrieval and for the ranking of the subcollections. However, our approach does inherit the issues of document selection and results merging; we describe the solutions to these issues within our approach in Section 3.

Constructing rankings of member expertise is a relatively new subfield of information retrieval research. TREC 2005 marked the introduction of the 'Expert Search Task', aimed at solving the problem of identifying employees who are the experts on a certain topic or in a certain situation [14]. Campbell et al. [7] performed similar experiments on a corpus of e-mail messages sent between people in the same company. Neither approach uses these expertise rankings to enhance any kind of information retrieval.

A considerable amount of research has been devoted to improving the search results of information retrieval systems. Among the more successful approaches are query expansion [19] and using cluster analysis [11] or citation analysis for re-ranking purposes [10].

3. AUTHORITATIVE RE-RANKING

As mentioned in the previous sections we try to identify subcollections within a single community collection based on the sets of documents written by an author and the expertise they implicitly represent. We assume that the aggregated content of an author's publications represents his or her expertise. Based on this assumption, we estimate how well a term or phrase points to a certain experts, by calculating the author-term co-occurrence weights in the community collection. We describe a method to create expertise rankings of the members for a query, and use these rankings to re-rank the search results produced by a baseline system. This is similar to producing collection rankings in distributed information retrieval where the collection weights signify the relevance of each collection for a specific query. In our case we combine the original document similarities

with subcollection-specific weights: the documents of authors who would be well suited to answer the query will be ranked higher in the final results list.

In addition to this, we also performed some experiments to determine which author rank contributes most to expertise re-ranking. We created special versions of each of our community collections where only the primary authors were included, and versions where the last author was removed from the author listings. Our hypothesis was that, on average, the first author has contributed the most to a paper and the final author the least. This is, in essence, a mild case of database selection by disregarding specific subcollections in the re-ranking process.

We do not use a probabilistic approach, but our approach has much in common with the collection fusion approach of Callan et al. [6]. They too combine the collection-specific weights with the baseline scores assigned to the documents. As in their approach, documents from 'good' collections and good documents from poor collections are favored in the end ranking.

3.1 Baseline approach

Our re-ranking approach was designed to be used on top of a basic vector space model of information retrieval. In our experiments, we used the following formulas for document weights (1) and query weights (2) as proposed by Chisholm et al. [8]:

$$dw_{ij} = \left(\sqrt{f_{ij} - 0.5} + 1\right)\left(\sqrt{\frac{F_i}{n_i} - 0.9}\right) \quad (1)$$

$$qw_{ij} = (1 + \log(f_{ij}))\left(\log\left(\frac{N}{n_i}\right)\right) \quad (2)$$

Here, f_{ij} is the frequency of term i in document j, n_i is the number of documents term i appears in, F_i is the frequency of term i throughout the entire collection, and N is the number of documents in the collection. Document-query similarity was calculated by using the cosine measure.

We incorporated some of the tried and tested low-level NLP-techniques in our baseline system, such as stopword filtering and stemming. One-word terms that occurred in the stopword list or in more than a certain percentage of documents were filtered from the documents, and all words were stemmed using the Porter stemming algorithm.

We also experimented with other higher-level techniques such as statistical phrases and using POS tagging and chunking to extract and index syntactic phrases. According to Brants [3], these processing techniques do not always yield improvements and may even result in a decrease in accuracy. Therefore we tested the utility of statistical phrases of different sizes, using syntactic phrases[1], and reweighting based on POS tags. We optimized the use of these techniques for every test collection, as recommended by Brants. We intentionally did not include other techniques such as query expansion in our baseline approach, nor did we distinguish in weighting between the text in the title or the abstract. We intended to measure the effect of our approach as clearly as possible without interference of other possible improvements.

[1]We used the Memory-Based Shallow Parser to obtain the POS and chunk tags. See [4] for more information.

3.2 Test collections

Investigating the merits of authoritative re-ranking required testing our approach on test collections that (a) contain information about the authors of each document, and (b) are a realistic representation of a community, such as a workgroup or a scientific community. We used two well-known test collections, **CACM** and **CISI**, that both represent scientific communities. **CACM** is a reference collection composed of all the 3204 article abstracts published in the Communications of the ACM journal from 1958 to 1979, and **CISI** is made up of 1460 document abstracts selected from a previous collection assembled at ISI [15].

We know of no publicly available IR test collections that represent the body of work published by a workgroup operating in a single institution, which prompted us to create our own: the **ILK** test collection[2]. **ILK** contains 147 document titles and abstracts of publications of current and ex-members of the **ILK** workgroup[3]. The topics of the papers are in the area of machine learning for language engineering and linguistics with subtopics ranging from speech synthesis, morphological analysis, and text analysis & processing to information extraction, text categorization, and information retrieval. We asked the current group members to provide us with queries and the corresponding binary relevance assignments, which resulted in 80 natural language queries.

Table 1: Characteristics of the three main test collections used in the experiments. The total author count ('# total authors') is the sum of the author count over all documents; the total number of unique authors ('# unique authors') is the sum of the author count over all documents with each author counted only once.

	CACM	CISI	ILK
# documents	3204	1460	147
# queries	52	76	80
# total authors	4392	1971	395
# unique authors	2963	1486	89
avg. # authors per document	1.371	1.350	2.687
avg. # unique authors per doc	0.925	1.018	0.605

Table 1 shows some numeric data characteristics of the three test collections. The four last features listed in the table seem to indicate the type of community collection. **ILK** has a high average number of authors per document but a low average number of unique authors per document, indicating a fairly high degree of cooperation within the community. The distribution of authors in **CACM** is similar to that of **ILK**. This in contrast to, say **CISI**, where these values are lower and higher respectively—it has more cases of solo authorship, and cooperation between the same authors rarely occurs more than once.

[2]Publicly available at http://ilk.uvt.nl/~tbogers/ilk-collection/.

[3]The Induction of Linguistic Knowledge (ILK) workgroup is part of the Department of Language and Information Science of the Faculty of Arts of Tilburg University. It focuses mainly on machine learning for language engineering and linguistics.

Table 2: Author-related characteristics of the six special test collections.

	CACM–first	CISI–first	ILK–first	CACM–m1	CISI–m1	ILK–m1
# total authors	3204	1460	147	3491	1637	278
# unique authors	2155	1112	43	2383	1250	74
avg. # authors per document	1	0.999	0.993	1.090	1.121	1.891
avg. # unique authors per doc	0.673	0.762	0.293	0.744	0.856	0.503

We also performed some experiments to determine which author rank contributes most to expertise re-ranking and created special versions of each collection for this. We created versions where only the primary authors were included (**CACM–first**, **CISI–first**, and **ILK–first**), and versions where the last author was removed from the author listings (**CACM–m1**, **CISI–m1** and **ILK–m1**). This means that, for each community collection, the special versions have the same number of documents and queries. Table 2 lists some characteristics of the six special test collections. The fact that special versions with only the first author have the same number of total authors as documents is not a coincidence. For instance, for **CACM** 3204 documents · 1 author = 3204 total authors.

3.3 Identifying subcollections

Identifying the subcollections in each community collection was a straightforward step. We equate subcollections with the documents written by a member of the community. A document can have multiple authors and can therefore belong to more than one collection—a situation no different from regular distributed information retrieval.

3.4 Determining subcollection weights

Our goal was to determine the expertise of each author to calculate the weights of the different author subcollections: authors with a lot of expertise on a certain query topic were assigned a higher weight. We partitioned the documents into one-vs-all data sets for each author, with each feature vector consisting of the term frequency counts f_{ij} for that document-author combination. In other words, we extracted author-term pairs based on the authorship of a document and the terms appearing in that document, but also the terms appearing in the other documents. We then calculated the co-occurrence weights of each author-term pair for each term (words and phrases) that occurred in the collection. This is similar to Callan's approach, who relies on, among other things, the term occurrence in different collections to calculate collection weights. We examine the co-occurrence of the terms with authors which also involves looking at the occurrence (or lack thereof) in the different author subcollections.

The weights were determined using the following feature selection metrics from text categorization: Information Gain, Chi-Square, and Mutual Information [20]. We also tested using the average TF·IDF value as a measure of term informativeness; collection terms that did not occur in the author's document were assigned a score of zero.

Combining these term weights for each author yielded a matrix of term-author weights which was used to extract the expertise rankings. For each query-author combination an expert score was calculated that signified the expertise of that author on the query topic. Calculating the expert scores was based on the straightforward assumption that if terms characteristic for author X occur in query Q, X is likely to be more of an expert on Q. For each author separately, the informativeness weights were collected for each of the query terms and combined into an expert score. We experimented with taking an unweighted average of the weights and an average weighted by the TF·IDF values of the query terms, so that the differences in the importance of the terms in the query were taken into account. However, there was no appreciable difference between the two, so we chose the intuitively more appealing TF·IDF-weighted average. The end result of this step was ranking of the different subcollections based on the expertise scores[4] for each query. This ranking effectively shows which authors are the biggest experts on the query topic, based on the documents they have authored.

3.5 Document selection & results merging

Document selection and results merging are two issues in collection fusion that are also important for our approach. One issue in document selection is that certain documents may have multiple authors and have different expertise scores. Since our approach works on a single collection and the baseline retrieval also returns a single similarity score for each document-query combination, these documents with multiple expertise scores need to be resolved into a single document score for that query. Merging results involves combining the results into a coherent set to be presented to the user and involves combining the original similarity scores with the expertise weights into a single ranking score. We therefore address both fusion issues simultaneously by re-ranking based on authority.

Our re-ranking is based on the premise that the documents authored by the experts on the current query topic are more likely to be relevant to the query, i.e. more *suitable* to resolve the query. Early experimentation with combining the different expertise scores showed that weighting the scores with the total number of publications of each author gave the best performance. We also investigated abating the influence of high numbers of publications with the square root and the natural logarithm of these counts as weighting factors, which, in general, worked slightly better, but not significantly. After computing this 'suitability' score, which is computed for each query-document combination, it is combined with the original baseline similarity score to form a new score on the basis of which the authoritative re-ranking is performed.

We also performed experiments to determine the optimal way of combining these two scores in order to re-rank the

[4]We will use 'subcollection weights' and 'expertise scores' interchangeably in this paper.

Table 3: Comparison of the re-ranking approaches on R-precision scores. The underlined scores are statistically significant improvements over the baseline.

community collection	re-ranked	baseline	% increase
CACM	0.313	0.233	(+34.3%)
CACM–first	0.302		(+20.2%)
CACM–m1	0.304		(+30.5%)
CISI	0.206	0.203	(+1.5%)
CISI–first	0.206		(+1.5%)
CISI–m1	0.206		(+1.5%)
ILK	0.649	0.647	(+0.3%)
ILK–first	0.650		(+0.5%)
ILK–m1	0.656		(+1.4%)

search results. The most successful combinations involved multiplying the original similarity score with the suitability score (*suit*) and transforming the original similarity score by multiplying it with $1 + suit$ (resulting in a number between 1 and 2). Experiments showed that the optimal re-ranking settings were collection-dependent, so the settings were optimized for each collection, similar to the NLP techniques [3].

4. EVALUATION

We evaluated the performance of our approach using R-precision, the precision at the cut-off rank of the number of relevant documents for a query. R-precision emphasizes the importance of returning more relevant documents earlier. The reliability of the comparisons between our baseline system and the re-ranking approach was determined by performing paired t-tests.

Table 3 shows the results of our experiments. The improvements seem to be very dependent on the community collection used, but improvements were present in each of the nine test collections. Authoritative re-ranking using author-based subcollections produced statistically significant performance improvements on the standard **CACM** test collection and the special versions, ranging from +20.2% to +34.3%. Statistically significant performance improvements were also present in the three versions of the **CISI** test collection, albeit much smaller at +1.5%. Optimal performance on the **ILK** collection yielded very small improvements, but these were not significant. Figures 1–3 show the precision plotted against recall for each cut-off point, both before and after re-ranking, and for each collection. The data points in the lower right half of each graph correspond to the lowest cut-offs. The graphs show that the biggest improvements were made in the top sections of the search results.

A possible reason for these differences in performance might be the topical diversity of the test collections: **CACM** has a much more diverse range of topics than **CISI** and **ILK**, which is likely to make it easier for different areas of expertise to be recognized. Our approach relies on terms that are specific for a certain topic area. This means that our approach has a harder time distinguishing between topics in collections where the different documents are closer together topic-wise.

The experiments with different author selections do not confirm our initial hypothesis: using the expertise of all authors associated with a document yields the best results

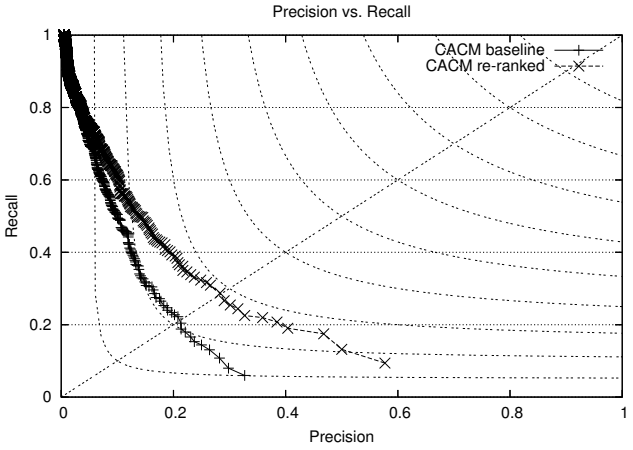

Figure 1: Precision vs. Recall for CACM.

and using less authors did not increase performance significantly. The difference between the type of community in **CACM** and **CISI** vs. **ILK** might offer an explanation for this, but we have not conducted a more extensive investigation into this matter. These findings suggest that more work is needed to determine the exact influence of author rank.

5. CONCLUSIONS AND FUTURE WORK

In this paper we presented a novel method of improving search results where we apply fusion techniques on a single collection instead of on disparate collections. We distinguish subcollections based on the sets of documents written by authors and use the content of their documents to produce expertise weights for each query. We use these weights to perform authoritative re-ranking of the baseline search results. Under optimized settings, authoritative re-ranking is able to significantly boost R-precision, especially improving the top search results, with the exact performance increase dependent on the document collection. Therefore, one issue for future research is comparing different ways of constructing expertise rankings such as using clustering, which could also be used to better determine the topical diversity of the three test collections. Another improvement might be the use of citation analysis to improve the expertise scores, similar to the approach taken in [10].

In theory, our approach is equally applicable to the search

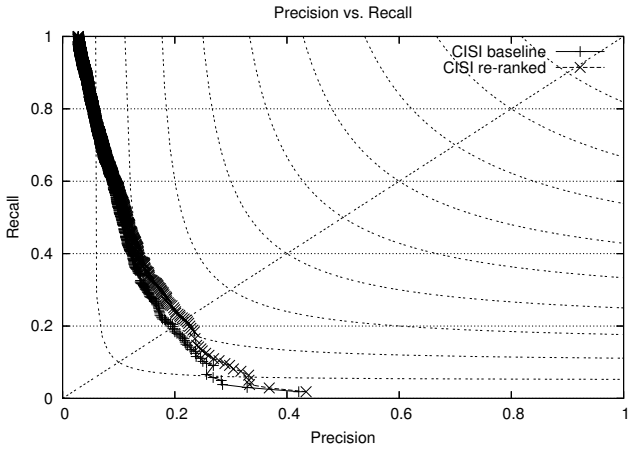

Figure 2: Precision vs. Recall for CISI.

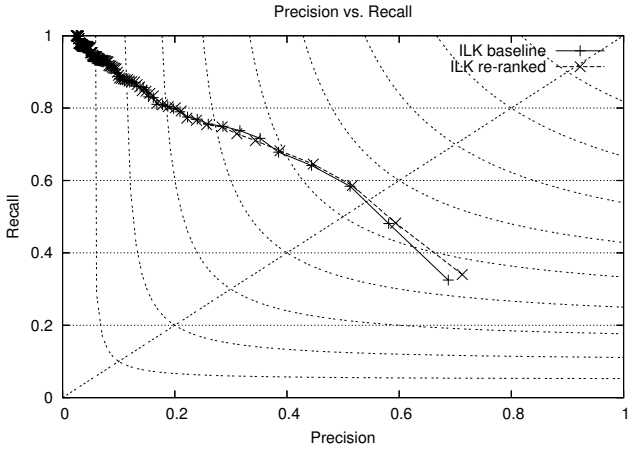

Figure 3: Precision vs. Recall for ILK.

results of, for instance, a probabilistic IR model. However, it would also be interesting to investigate whether using other IR models such as probabilistic retrieval or a language modelling approach indeed show this increase to be universal over the entire range of IR approaches.

Optimal re-ranking performance involves using the expertise of all the authors associated with a document, since considering a smaller number of authors does not increase performance significantly and often decreases it. These findings suggest that more work is needed to determine the exact influence of author rank.

6. ACKNOWLEDGMENTS

This research was funded by the IOP-MMI-program of SenterNovem and the Dutch Ministry of Economic Affairs, as part of the À Propos project. The authors would like to thank Krishna Barat (Google) and Frank Hofstede (IntelliGent B.V.) for fruitful discussions and useful comments.

7. REFERENCES

[1] E. Adar, D. Kargar, and L. Stein. Haystack: Per–user Information Environments. In *CIKM '99: Proceedings of the Eighth International Conference on Information and Knowledge Management*, pages 413–422, New York, NY, 1999.

[2] C. Baumgarten. A Probabilistic Solution to the Selection and Fusion Problem in Distributed Information Retrieval. In *SIGIR '99: Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–253, New York, NY, 1999. ACM Press.

[3] T. Brants. Natural Language Processing in Information Retrieval. In *Proceedings of CLIN 2004*, pages 1–13, Antwerp, Belgium, 2004.

[4] S. Buchholz, J. Veenstra, and W. Daelemans. Cascaded Grammatical Relation Assignment. In *EMNLP-VLC'99, the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

[5] J. Budzik and K. Hammond. Watson: Anticipating and Contextualizing Information Needs. In *62nd Annual Meeting of the American Society for Information Science*, Medford, NJ, 1999.

[6] J. P. Callan, Z. Lu, and W. B. Croft. Searching Distributed Collections with Inference Networks. In *SIGIR '95: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, New York, NY, 1995. ACM Press.

[7] C. Campbell, P. Maglio, A. Cozzi, and B. Dom. Expertise Identification using Email Communications. In *Proceedings of CIKM2003*, pages 528–531, New Orleans, LA, 2003.

[8] E. Chisholm and T. Kolga. New Term Weighting Formulas for the Vector Space Method in Information Retrieval. Technical report ORNL/TM-13756, Computer Science and Mathematics Division, Oak Ridge National Laboratory, 1999.

[9] Dogpile. http://www.dogpile.com, 2006. Visited: January 10th, 2006.

[10] C. Giles, K. Bollacker, and S. Lawrence. CiteSeer: An Automatic Citation Indexing System. In *Proceedings of Digital Libraries 98*, pages 89–98, 1998.

[11] K.-S. Lee, Y.-C. Park, and K.-S. Choi. Re-ranking model based on document clusters. *Information Processing & Management*, 37(1):1–14, 2001.

[12] R. M. Losee and L. Church Jr. Information Retrieval with Distributed Databases: Analytic Models of Performance. *IEEE Transactions on Parallel and Distributed Systems*, 14(12):1–10, December 2003.

[13] M. J. Pazzani, J. Muramatsu, and D. Billsus. Syskill & Webert: Identifying Interesting Web Sites. In *AAAI/IAAI, Vol. 1*, pages 54–61, 1996.

[14] TREC. TREC Enterprise Track, 2005. Visited: October 2005, http://www.ins.cwi.nl/projects/trec-ent/.

[15] C. van Rijsbergen. *Information Retrieval*. Dept. of Computer Science, University of Glasgow, Second edition, 1979. http://www.dcs.gla.ac.uk/Keith/Preface.html.

[16] Vivísimo. http://vivisimo.com, 2006. Visited: January 10th, 2006.

[17] E. M. Voorhees, N. Gupta, and B. Johnson-Laird. The Collection Fusion Problem. In *Proceedings of the Third Text REtrieval Conference (TREC-3)*, pages 95–104, Gaithersburg, MD, 1995.

[18] W. Wu, H. Xiong, and S. Shekhar. *Clustering and Information Retrieval*. Springer, First edition, 2004.

[19] J. Xu and W. Croft. Query Expansion Using Local and Global Document Analysis. In *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, New York, NY, 1996. ACM Press.

[20] Z. Zheng and R. Srihari. Optimally Combining Positive and Negative Features for Text Categorization. In *Workshop for Learning from Imbalanced Datasets II, Proceedings of the ICML*, Washington, DC, 2003.

Utilizing scale-free networks to support the search for scientific publications

Claudia Hauff
Human Media Interaction (HMI)
University of Twente, Enschede, the Netherlands
c.hauff@ewi.utwente.nl

Andreas Nürnberger
Inst. for Knowledge and Language Processing
University of Magdeburg, Magdeburg, Germany
nuernb@iws.cs.uni-magdeburg.de

ABSTRACT

When searching for scientific publications, users today often rely on search engines such as Yahoo.com. Whereas searching for publications whose titles are known is considered to be an easy task, users who are looking for important publications in research fields they are unfamiliar with face greater difficulties since few or no indications of a publication's importance to the respective fields are given. In this paper we investigate the application of the theory of scale-free networks to derive importance indicators for a collection of publications. A tool was developed to support the user in his publication search by visualizing the publications' importance indicators derived from the number of citations received and the publication's age as well as visualizing part of the citation network structure. A preliminary user study indicates the utility of our approach and warrants further research in that direction.

Categories and Subject Descriptors

H.5.0 [**Information Interfaces And Presentation**]: General

Keywords

Information Retrieval, Scale-Free Networks

1. INTRODUCTION

When searching for scientific publications, users today often rely on search engines such as Yahoo.com. Whereas searching for publications whose titles are known is considered to be an easy task, users who are looking for important publications, e.g. publications that are fundamental or had a great influence in the research community, in research fields they are unfamiliar with face greater difficulties. Although the results of such searches are likely to contain publications in the correct research fields, few or no indications are given to the user of how important or influential the publications are to the respective fields.

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

Google Scholar offers such an indicator by providing the total number of received citations for each publication. This simple measurement however has a drawback: it heavily disadvantages recent publications that have not had the time yet to acquire a large number of citations.

In this paper, we present an approach, that aims to provide the user with a more valid importance indicator of publications which takes the publications' age into consideration in a principled way. It relies on the theory of scale-free networks [4] which started to emerge in the late 1990s when it became clear that many real-world networks, including citation networks, have a common property: the distribution of the number of links k connected to a node, the so-called *degree distribution* $P(k)$ of a network, follows a power-law form. This property can be described as follows: the probability of a node to have received few links from other nodes is high, while the probability of a node to have been linked to by a large number of nodes is very low. In the specific case of citation networks publications form the nodes and citations or references represent the directed links (from the citing to the cited publication) of the network. Within the last few years a number of network models were developed [2, 4, 7, 14, 19, 29] that are able to generate networks with the desired degree distribution and as a by-product closely resemble the growth process as it occurs in many real-world networks.

The knowledge gained about the true structure of real-world networks in recent years has so far been rarely exploited. Much research has concentrated on developing network models that resemble real networks as closely as possible, but few applications actually take advantage of this additional knowledge. One notable exception is the research in eradicating epidemics where the knowledge has been applied to identify highly connected nodes that should be treated first in order to decrease a virus' spreading rate [11]. We adopt a different approach and hypothesize that it is possible to gain valuable information by comparing a real-world network with its corresponding network model. The model is created from statistics derived from the real-world network such as the age of the nodes, the network size, the average degree and the degree exponent. While the degree distribution of the model and the real-world network will be the same or at least be very similar, on the individual node level the degrees will almost certainly be different. Previously, we applied this idea to the ad-hoc retrieval task of a collection of Web pages [18].

In brief, our approach works as follows: given the age of a publication and the degree distribution of the citation network under consideration, we are able to predict the expected number of citations pointing to the publication by utilizing a scale-free network model. This number is then compared with the actual number of citations the publication has received. This comparison yields an indicator of how important a publication is - if the actual number of publications citing it is higher than expected, the publication is more important than one with fewer citations than expected. This makes it possible for example, that a paper, published 12 months ago, with 5 citations pointing to it receives a higher importance score than a paper with 10 citations that was published 7 years ago.

In order to evaluate our idea, a software tool called *Visual Paper Finder* (ViPF) was developed. It allows the user to search for scientific publications and visualizes the derived importance for each returned publication as well as a part of the citation network. The user is able to navigate within the citation network, further enhancing the search process.

In a preliminary user study, the usefulness of the introduced approach and of ViPF were evaluated utilizing the collection of publications indexed by Citebase, a web service with more than 360000 publications in the fields of physics, mathematics, biology and computer science. Although the results of the user study were mixed, the general outlook was positive and warrants further research in that direction.

The remainder of this paper is organized as follows: in Section 2 the theory of scale-free networks is introduced. Section 3 presents arguments in favor of utilizing citations as importance indicators and discusses the scale-free character of citation networks. Section 4 describes ViPF in greater detail. In Section 5 the Citebase data set and the conducted user study are presented. The results of the study are described in Section 6. Finally, the conclusion and directions for future work can be found in Section 7.

2. SCALE-FREE NETWORKS

Scale-free networks appear to be abundant in natural and artificial systems and among others can be found in the social [4, 6, 23, 25], biological [21, 28] and technological [5, 16] domain. More unusual examples where one would not readily suspect a (scale-free) network structure are the network of earthquakes [1] and the medieval inquisition [24].

The most basic network model able to produce a power-law degree distribution is the Barabási-Albert (BA) model [4] which is described below. In Section 2.2 the accelerated growth model [12] is presented which is the model chosen for our experiments. It is a particular example of an extension to the original BA model. Other developments include the modeling of clustering within networks [19], of aging and physical limitations of nodes [3] and the introduction of weighted [29] or rewired links [2].

2.1 BA Model

In scale-free networks, the probability $P(k)$ of a node having k links follows a power law with degree exponent γ

$$P(k) \propto k^{-\gamma}.$$

Barabási and his collaborators identified two necessary conditions for the creation of networks with such a degree distribution: *growth* and *preferential attachment*.

The building process of scale-free networks is iterative: starting with a small number m_0 of nodes, at each time step one node with m ($m \leq m_0$) undirected links attached to it joins the network. The free ends of the new links are distributed preferentially among the nodes already in the network. Each node is denoted by its time of birth, thus node s entered the network at time s. Formally, the probability $\prod$ that node s with degree $k(s,t)$ receives a new edge at time t is defined as

$$\prod = \frac{k(s,t)}{\sum_u k(u,t)}. \quad (1)$$

The denominator $\sum_u k(u,t)$ corresponds to the total degree of the network. Thus, the higher the degree $k(s,t)$, the higher the probability of receiving further links. Equation 1 allows us to derive a function that determines the expected number of links a node should have acquired at any time t ($t \geq s$), given the node's age s

$$k(s,t) = m \left(\frac{s}{t}\right)^{-\frac{1}{2}}.$$

Due to its simplicity, the BA model lacks many of the actions possible in real networks: neither can links be rewired or introduced between old nodes, nor can links or nodes be deleted from the network. Furthermore, the algorithm produces only undirected networks. But citation networks are directed and - as will be seen in the Experiment section - the number of links added to the citation network is not constant but accelerates as the network grows. For this reason, the model introduced next is the one chosen for the experiments.

2.2 Accelerated Growth

In directed networks, the in- and out-degree are considered separately. We will concern ourselves only with the in-degree k_{in} of a node as the number of citations received is of importance to us, not the number of citations a publication contains. In those networks, the target ends of the links are of relevance, while the source ends, which can be anywhere within or outside the network, are ignored.

A network exhibits accelerated growth when its number of links grows faster than its number of nodes, leading to a non-stationary average degree. Although negative acceleration - the number of edges grows slower than the number of nodes - is also possible, it will not be considered here.

There are two general processes that lead to accelerated growth. In the first place when the network grows the number of links a new node enters the network with can also grow. This can be the case in citation networks for example, where the amount of literature increases over time, there is more to cite and hence, the average number of references on a publication increases. A second possibility is the addition of new links between old nodes. Actor and collaboration networks can be named here.

It is assumed that the number of links grows faster than the

number of nodes according to a power-law

$$k_{in} = c_0 t^b \tag{2}$$

with b as the growth exponent and c_0 as a constant. It is clear that $b < 1$ for most real-world networks, otherwise the average degree would increase indefinitely.

If the condition $\gamma_{in} > 2$ holds for the in-link power-law distribution (as is the case for the Citebase data set), links are attached to a node with a probability proportional to

$$k_{in}(s,t) + Bc_0 t^b/(1+b)$$

with B is positive constant. This leads to

$$k_{in}(s,t) = \left(\frac{Bc_0 s^b}{1-Bb}\right)\left(\frac{s}{t}\right)^{-(1+b)/(1+B)} - \frac{Bc_0 t^b}{1-Bb} \tag{3}$$

for the expected number of in-links of node s with age s at time t. To summarize, in order to calculate $k_{in}(s,t)$ the following network statistics are necessary: the exponent γ_{in} of the power-law degree distribution, the accelerated growth parameters c_0 and b, the age s of each node and the total number of nodes t in the network.

For a thorough coverage of the mathematical aspects of the theory of scale-free networks and the derivation of the presented formulas as well as an in-depth look at real-world examples, the interested reader is referred to [13].

3. CITATION NETWORKS

3.1 Citations as Importance Indicators

One of the assumptions of this work is the existence of a positive correlation between the number of citations a publication receives and the publication's importance. Intuitively we expect a citation to mean that the two papers are related by content or semantics; that the cited paper is qualitatively good enough to be cited, that an author cites all papers that he should cite and none else. It is not difficult to imagine, that not all these assumptions hold in the real world - it is unrealistic to assume that a researcher knows all papers relevant to his research or that he will cite all papers that ought to be cited as there are constraints on the length of papers. There can also be other reasons for citations: a social relationship between the authors, self-citations purely to increase the citation count, negative citations (citing a paper to criticize it) or the copying of citations from other papers.

A number of studies have been conducted to determine how the citation situation in the real world differs from our expectations. If the number of citations is indeed dependent on the quality, importance or influence of a publication, one possibility to determine the validity of the assumption is to compare the citation count of high quality papers with the ones of average papers as done by Brooks [9]. He defined high-quality papers to be those that received high ratings in the peer review process. The result was that the citation count for best paper award publications was considerably higher than that of other papers. This finding implies a correlation between quality and number of citations, although it should not be forgotten that the best paper award provides a paper with a special awareness in the research community. A similar study was conducted by Rinia et *al.*[26], who compared the citation count of research programs in physics in the Netherlands with peer review judgments. They also found a good - though not perfect - correlation between citation counts and peer review judgments. White et *al.*[27] approached the central question of this section differently. They observed a group of scientists who over a period of 10 years built up personal and professional relationships and found that professional relationships far outweigh the social ties within the group. Therefore, despite the fact that many factors influence the citation behavior of authors, overall the qualitatively good publications are likely to be cited more often than an average paper.

However, for very specific fields of research that are studied only by a small group of researchers this generalized approach most likely fails. Here, further information, e.g. about topic specificity, network cohesion or cliques have to be considered, which is, however, beyond the scope of this work.

3.2 Are Citation Networks Scale-Free?

Several researchers have investigated scientific citation networks as part of the research in scale-free networks [8, 23, 25]. With very few exceptions, only the in-degree distributions were examined and only those citations between papers that both appear in the data set were taken into consideration.

Redner [25] conducted the first large study on citation networks using publications indexed by ISI and a second data set of Physical Review D papers. The in-degree exponent γ_{in} was found to approach 3 for $k_{in} > 500$. In the regions of low k_{in} the degree distribution was following a stretched exponential. A very similar result was achieved when examining the Physical Review D data set Volumes 11-50 from the years 1975 to 1994.

A cleansed version of the SLAC SPIRES database was studied by Lehman et *al.*[23]. 281717 publications were included in the estimation of the degree distribution. They reported a scale-free behavior with two regimes; papers with 50 or less citations follow $P(k_{in}) \propto k_{in}^{-1.3}$ and papers with more than 50 citations $P(k_{in}) \propto k_{in}^{-2.3}$.

Boerner et *al.*[8] reported a best fit for their citation data from the Proceedings of the National Academy of Sciences (PNAS) not for a pure power law, but for a power law with an exponential cut-off. They suggest that the cut-off is due to the aging of papers, as the most cited papers exist less often than predicted by the pure power law form and lowly cited papers exist more often.

4. VISUAL PAPER FINDER

Imagine being faced with the task of becoming familiar with the current developments in a research area you know very little about. Apart from locating the milestone publications within the research area it is also necessary to find recent publications that have attracted much attention. Those publications are likely to contain the current state-of-the-art of the area. ViPF was developed to support users in such a scenario and is mainly aimed at researchers, PhD students, advanced undergraduate students and generally people that are new to a research field.

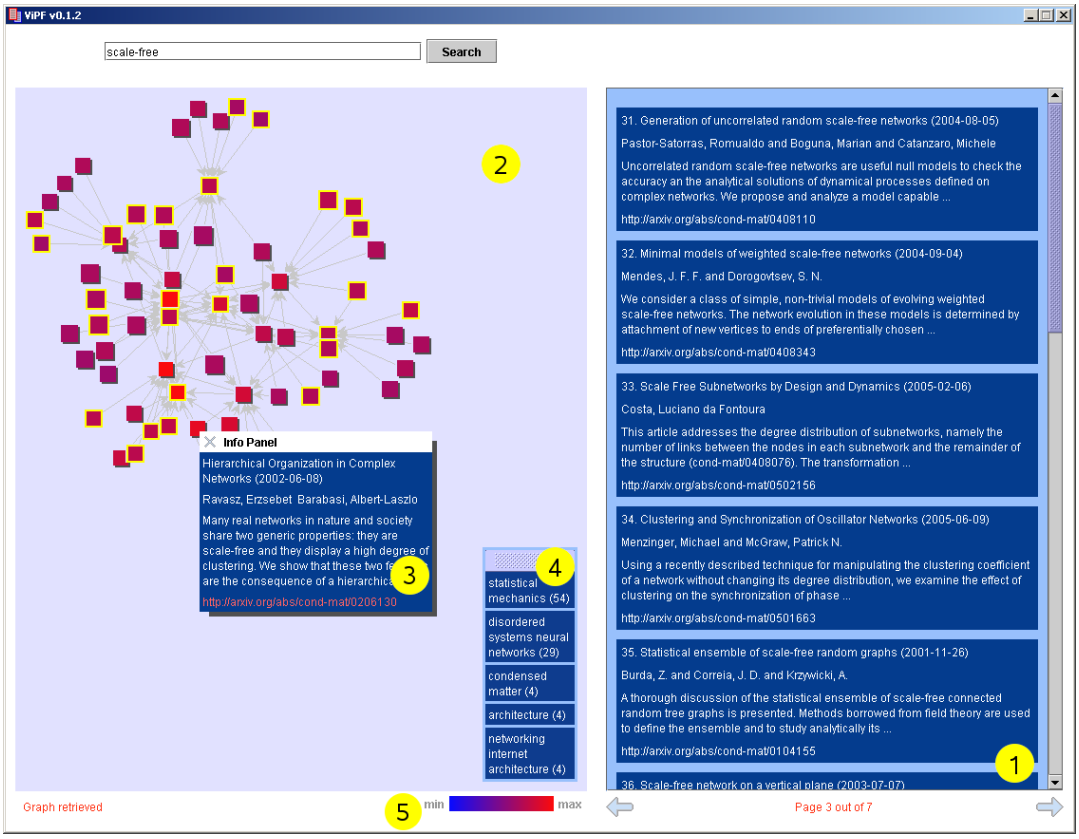

Figure 1: ViPF interface after a query was submitted: result panel (1), graph panel (2), info panel (3), subject panel (4), fitness color bar (5)

Images that visualize citation networks or more generally bibliographic networks are not difficult to find, one source is the InfoVis contest 2004 [17]. But there very few freely available tools that visualize parts of a bibliographic network interactively. CiteSpace [10] tracks the changes of a knowledge domain over time by highlighting major changes between adjacent time slices. The Growing Polygons causality visualization technique is applied in CiteWiz [15] and a multitude of different information panels with information about citations, topics and authors are presented by PaperLens [22]. However these tools require an intensive effort by the user since the visualizations are very complex and not feasible for everyday usage when searching for scientific publications. ViPF was developed with these problems in mind which is reflected in its simple interface.

4.1 Interface

A screenshot of ViPF's interface can be seen in Figure 1. It consists of two panels - a graph panel and a result panel. Given a query, in the result panel the ranked list of returned publications of a content-only search are presented. Each result entry consists of four parts: title and publication or first uploading date of the publication, author(s), the first part of the abstract and the URL that points to the web page where the publication can be downloaded. The graph panel visualizes part of the citation graph with a given publication as root node. After the results for a query are retrieved the citation subgraph for the top returned result is automatically shown. Clicking on an arbitrary result field retrieves the citation graph with the corresponding publication as root node. As we are interested in the publications citing a paper, the subgraph is built up by following the root node's incoming citations and doing this for every other node recursively up to a certain depth. The color of each node indicates its importance or fitness as determined by the comparison between actual and expected number of received citations. For better orientation the gradient color bar at the bottom of the graph panel shows the colors for maximum and minimum fitness. A click on any of the visualized nodes opens an info panel with information about the publication. The size of the visualized nodes varies, depending on the publication's age - the larger the node, the more recent is the publication. This feature shall make it easier to find recent papers without having to open each node's info panel to find out its publication date. The subject panel is a further help to the user. It shows the top five subjects within the retrieved subgraph. The number of nodes belonging to the respective subject is given in brackets. Clicking on a subject highlights the nodes belonging to the subject, visualized by a specially colored border. As one node can belong to more than one subject (or none), the sum of the elements in the top five subjects may exceed or fall below the total number of shown nodes.

The behavior of ViPF is managed through a parameter file. In it the age and subject indicators can be switched on or

off, the depth up to which the graph shall be displayed can be changed and the colors of all elements of the interface can be adjusted.

To avoid an overloaded display, papers with more than a certain number of citations pointing to it have not all citing papers shown. Instead, the papers were ranked according to their importance indicators and only the top n nodes were displayed. The value of n can also be modified through the parameter file.

4.2 Implementation

ViPF was implemented in Java, the graph visualization was realized with the open source libraries JGraph and JGraphAddons. ViPF relies on Citebase's retrieval engine (Xapian) and does not perform the retrieval process itself. Returned to ViPF is an XML file that contains a maximum of 100 retrieved publications. From the XML stream the required information is extracted and presented to the user in the result panel. To keep the traffic on Citebase low, the structure of the visualized part of the citation network is retrieved from a local database. The database was constructed from Citebase's metadata and includes all necessary information for the importance indicator calculation on each publication, that includes the time stamp of each paper, the number of each paper's outgoing and incoming references and a list of subjects the paper belongs to. The necessary network statistics are also stored in the database and retrieved by ViPF after every start of the program.

A question raised during the design of ViPF was whether or not to mix content scores with citation based scores in the result ranking. It was decided to use pure content ranking in order not to bias the ranking against or in favor of highly cited papers. The reasoning is, that most papers will cite the important or ground-breaking papers of a field and thus it should still be possible to gain valuable information from the visualization. Furthermore, in the graph panel it is only possible to explore papers that have received citations or reference a paper within Citebase. Since a sizable portion of papers in Citebase have no citations associated with them (as will be seen in the next section), they would then probably neither appear in the result nor in the graph panel.

5. EXPERIMENTAL SETUP

5.1 The Collection

Citebase, developed at the University of Southampton, is a search service for freely available publications of the Web. We received a citation file from the 2nd June 2005 with all available citations between publications that both appear in Citebase's index. The total number of papers amounts to 363207, with the largest proportion of papers (89%) coming from arXiv.org, an e-print repository for papers in the fields of physics, computer science, mathematics, quantitative biology and non-linear science.

The citation file contained 2501180 citations between 272349 papers, thus 25.02% of the publications have neither incoming nor outgoing references. This can be explained with the fact that Citebase only accounts for references between papers that both appear in Citebase, hence missing a substantial number of references. Although this is not an ideal

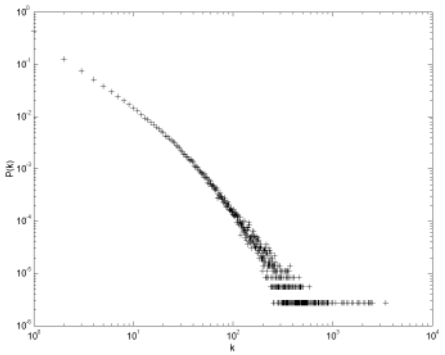

Figure 2: In-degree distribution of the Citebase data set with $\gamma_{in} = 2.1564$

situation for our experiments, at that time it was the best data set available.

5.1.1 The In-Degree Distribution

Of the papers with references, 203813 papers had received citations from other Citebase papers. Apart from the incomplete coverage of the citation network, reasons for the lack of incoming references for a paper include the recency of the paper (no author had yet the chance to reference the publication), the publication of a paper in a highly specialized field or simply the low quality of the paper.

The average number of citations is 6.89, the median is 1. 84.98% of papers have received 10 or fewer citations; only 2.55% of papers have 50 or more references pointing towards them. The top 4.21% of papers generate 50% of all incoming citations. 0.89% of citations are generated by the lowest 50% papers. These uneven numbers suggest, that the degree distribution follows a power law: there are very few highly connected nodes and many lowly connected nodes. In order to determine whether or not the Citebase in-degree distribution is indeed scale-free, $P(k_{in})$ was plotted on a log-log plot. The power law degree distribution is only defined for $k_{in} \geq 1$, the Citebase data set however contains a large portion of papers with $k_{in} = 0$. Ignoring such a large part of the collection is not an option and for this reason, the in-degree of each node was increased by 1. The resulting plot is shown in Figure 2. A slight curvature in the data set is visible. This is not unexpected though, as models are a simplification and idealization of real-world processes. The power-law form is a reasonable estimate of the observed data. We calculated the degree exponent applying the Maximum Likelihood method: $\gamma_{in} = 2.1564$.

5.1.2 Preferential Attachment

In the presented scale-free network models it is assumed that preferential attachment exists. Whether or not this is the case for the Citebase data set was investigated by applying the approach described in [20]. Let $\prod(k_{in})$ be the rate at which nodes with in-degree k_{in} receive further connections. Due to large fluctuations in the higher regions of k_{in}, the cumulative distribution function $F(k_{in})$ yields a more robust

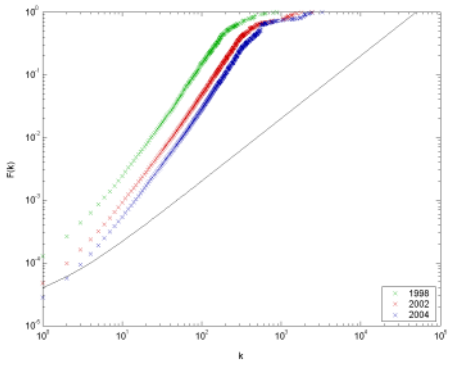

Figure 3: Preferential attachment measurements of the Citebase data set

estimate

$$F(k_{in}) = \int_0^{k_{in}} \prod(k_{in})dk.$$

The shape of $F(k_{in})$ will indicate if preferential attachment is present. If it is absent, $\prod$ is independent of k_{in} and thus $F(k_{in}) \propto k$.

Three set of experiments in different time intervals were performed, Figure 3 shows their cumulative distributions.

The continuous line indicates the shape of a cumulative distribution that is independent of k. Clearly, the increase of $F(k_{in})$ is faster (the slope is steeper), supporting the claim that preferential attachment is at work. Moreover, the form of $F(k_{in})$ is independent of the time interval, the preferential attachment process does not change considerably over time. Although it was argued earlier that old papers should be treated differently as the age is also a determining factor for a publication's citations (older papers are less referenced), this problem is negligible for the Citebase data set as most papers in this database were written in the 1990s or later.

5.1.3 Accelerated Growth

In Table 1 the number of publications, the number of citations and the average in-degree of the Citebase data set are listed for 6 different time periods. The reason for the overall increase in degree is obvious: as more papers are made available in Citebase, the chances that references from a newly published paper point to papers already in the Citebase database increase. Recall that in the accelerated growth model it is assumed that the average degree grows as a power of t. Since the in-degree distribution was estimated by adding one in-link to each node, to keep the estimate consistent, this also happened here. At a total of 29 points in time the average degree was measured. Equation 2 was logarithmized and the parameters c_0 and b were estimated by linear regression: $c_0 = 0.1043$ and $b = 0.3439$. The knowledge of the values for γ_{in}, c_0 and b allows the calculation of the value of the only missing parameter of Equation 3: $B = 0.554$.

5.1.4 The Documents' Age

Now what remains is to assign a time stamp s to each publication of the collection. The papers were ordered by their creation or upload dates. Only 1.49% of publications had an invalid date and had to be assigned an estimated date, minimizing the impact of the erroneous or missing data. When a paper had two or more creation dates, the earliest date was chosen. The oldest paper was assigned the time stamp 1, the second oldest the time stamp 2 and so on. The youngest paper received the time stamp 363207. If two more more papers had the same creation date, their ordering was determined randomly. Papers that have neither outgoing nor incoming references were also included. The accelerated growth model does not require a node to enter the network with links attached to it.

period	#papers	#references	$\overline{k}_{in}$
1900 - 1995	34822	86373	2.4804
1900 - 1997	71042	299754	4.2194
1900 - 1999	124447	668786	5.3741
1900 - 2001	196296	1204148	6.1343
1900 - 2003	285868	1915638	6.7011
1900 - 2005	363207	2501178	6.8864

Table 1: With an increase in network size, the average in-degree $\overline{k}_{in}$ increases.

5.1.5 The Subjects

Citebase's metadata contains one or more subject entries for 147193 (40.52%) publications. 119978 of those belong to more than one subject category. The subject entries had to be cleaned manually, since they contained entries such as 'Reviews' or 'Research Article' which were not useful for the visualization. In a number of cases one or several alphanumerical identifiers (Mathematics Subject Classification) were listed as subjects, which had to be manually converted to meaningful phrases. The final number of subjects was 4212. Using only those subjects, 140683 papers were left with one or more subjects. Those subject assignments were used in the subject panel of ViPF's interface.

5.2 User Evaluation

A preliminary user evaluation was conducted in the first two weeks of September 2005. ViPF was available for download and users were asked to test it and fill out an online questionnaire, available in German and English, afterwards.

The questionnaire was divided into three sections. The first section asked for basic information about the user, including occupation, age and field of study. The users were also asked to estimate the amount of time they had spent using ViPF. The second section consisted of 9 questions about the user's experience with ViPF. Each question had to be answered with a score between 1 (very positive/helpful/useful) and 5 (very negative/unhelpful/unuseful) or 'no opinion' respectively. The last section contained four questions about the user's view of ViPF in free-form and the users were not restricted in the length of their answers. They were asked to describe what kind of papers they had been searching for, their general impression of ViPF and what tools or web interfaces they normally use for the search for scientific publications.

Question	#Users assigning score 1	2	3	4	5	no op.	av. score
1. How useful is the visualization of the reference graph	2	2	6	1	0	0	2.55
2. Was the tool intuitive to use?	0	7	1	3	0	0	2.6
3. How useful is the fitness indicator for each paper?	2	3	5	1	0	0	2.55
4. Inhowfar did the search results meet your expectations?	1	3	1	4	0	2	2.89
5. Do you prefer Google (Scholar) or a similar search engine over ViPF?	1	4	4	1	0	1	2.5
6. When searching within the reference graph did you consciously pick papers with a high fitness value?	4	0	3	2	1	1	2.6
7. In case the age indicator was switched on, did you find it helpful?	0	2	2	0	1	6	3
8. How familiar are you in the research area of your paper search?	1	2	3	3	1	1	3.1
9. In case the subject indicator was switched on, did you find it helpful?	1	3	1	0	0	6	2

Table 2: Evaluation results

6. RESULTS

The questionnaire was filled in by 11 users, 7 questionnaires were returned in German and 4 in English. The average age was 24.5 years, 6 users gave Germany as their home country, 2 the United States of America and one each the United Kingdom and the Netherlands. One user did not provide a country entry. The majority of users (6) were undergraduate university students, three were researchers, one a PhD student and one a high school student. All but one user who did not give any information, stated computer science or a related term as their field of study: computer science (8), information retrieval (1) and computational visualistics (1). The fact that all but one user study or conduct research in the field of computer science was reflected in the searches. Only 5 users looked for publications in fields other than computer science. The amount of time spent using ViPF was less than 1 hour for all but one tester who spent 5 hours testing it. The results for each of the 9 questions are presented in Table 2. Users that voted 'no opinion' on a question were not taken into consideration for the calculation of the average score.

The reference graph and the fitness indicator were viewed rather positively by the users (10 users gave a score between 1 and 3), although in both cases most votes (6 and 5 respectively) were given to the score 3. This undecidedness is reflected in the answer to the question how consciously the fitness indicators were used. Only 4 users answered with a score of 1, 3 users with a score of 3 (partially used) and 3 did not use the fitness indicator consciously. The age and subject indicator were accepted as a useful feature by the majority of users who responded to these 2 questions.

The replies to the question about negative aspects of ViPF were quite similar to each other. The main point of frustration was the lack of a reference graph for many searches. Due to the small number of computer science papers compared to the number of physics papers in the collection, the retrieved papers often had not received a single citation from other Citebase papers, and thus the graph panel was rendered useless. This made it difficult for the users to accurately estimate the usefulness of the fitness parameters as large reference graphs were often available in areas they had not enough knowledge in.

The visualization of the reference graph was noted as a positive aspect by 10 users in the free-form answers. The fitness indicator and the subject indicator were also positively mentioned. The simple layout of the user interface provided little distraction and was also welcomed One user summarized his thoughts about ViPF as follows: 'I guess I am so used to the ranked list kind of interface that even with conscious effort to use the other component [the graph panel], my main entry point was always the ranked list.' This factor might also have contributed to the overall results. Most users are so used to use Google (10 users listed it as search tool of their choice) that it is difficult to introduce a different system.

7. CONCLUSIONS & FUTURE WORK

One research objective of this work was to answer the question whether or not the introduced approach - deriving an importance indicator from the comparison of actual and expected number of received citations - can be utilized to support users in their search for scientific publications. This question could only be answered partially since the results of the user study were too mixed to allow a conclusion. There are several reasons for this, among them the size of the user study and the recruited users. The searches in the field of computer science often only returned a single node, so that the users had no chance to evaluate the fitness indicators properly in their field. For a thorough test, users that are experts in the various fields of physics covered by Citebase need to be recruited to give a valid estimate of the usefulness and correctness of the fitness indicator. The preliminary user study should be followed by a larger one with a clear retrieval task and a set of measurements to evaluate the users' efforts, possible with a different publication data set, to gain more representative results.

The second objective - an appropriate visualization of the citation network - was achieved. The majority of users had a positive attitude towards the presented visualization and the simplicity of the interface.

Apart from a more representative user study, there are two major directions for future work, on the one hand the im-

provement of the ViPF interface and the optimization of the retrieval and graph visualization and on the other hand the extension of the scale-free network approach.

Acknowledgments

We would like to thank Tim Brody of the University of Southampton for providing the necessary data set and allowing the usage of the Citebase web services for our experiments.

8. REFERENCES

[1] S. Abe and N. Suzuki. Scale-free network of earthquakes. *Europhys. Lett.*, 65, 2004.

[2] R. Albert and A. Barabási. Topology of evolving networks: local events and universality. *Phys. Rev. Lett.*, 85(2):5234–5237, 2000.

[3] L. Amaral, A. Scala, M. Barthélémy, and H. Stanley. Classes of small-world networks. In *Proceedings of the National Academy of Sciences*, volume 97, pages 11149–11152, 2002.

[4] A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[5] A. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the World-Wide Web. *Physica A*, 281:69–77, 2000.

[6] A. Barabási, H. Jeong, R. Ravasz, Z. Néda, T. Vicsek, and A. Schubert. On the topology of the scientific collaboration networks. *Physica A*, 311:590–614, 2002.

[7] A. Barrat, M. Barthélemy, and A. Vespignani. Weighted evolving networks: coupling topology and weight dynamics. *Phys. Rev. Lett.*, 92(22):228701, 2004.

[8] K. Boerner, J. Maru, and R. Goldstone. The simultaenous evolution of author and paper networks. In *Proceedings of the National Academy of Scienes*, volume 101, Supplement 1, pages 5266–5273, 2004.

[9] T. Brooks. How good are the best papers of JASIS? *Journal of the American Society for Information Science*, 51(5):485–486, 2000.

[10] C. Chen. Measuring the movement of a research paradigm. *Visualization and Data Analysis*, 5669:63–76, 2005.

[11] Z. Dezsó and A. Barabási. Halting viruses in scale-free networks. *Phys. Rev. E*, 65:055103, 2002.

[12] S. Dorogovtsev and J. Mendes. *Handbook of Graphs and Networks: From the Genome to the Internet*, chapter Accelerated growth of networks. Wiley-VCH, 2002.

[13] S. Dorogovtsev and J. Mendes. *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford University Press, 2003.

[14] S. Dorogovtsev, J. Mendes, and A. Samukhin. Structure of growing networks with preferential linking. *Phys. Rev. Lett.*, 85(21):4633–4636, 2000.

[15] N. Elmqvist and P. Tsigas. Citewiz: A tool for the visualization of scientific citation networks, 2004.

[16] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the Internet topology. In *SIGCOMM'99: Proceedings of the conference on applications, technologies, architectures and protocols for computer communication*, pages 251–262, 1999.

[17] J.-D. Fekete, G. Grinstein, and C. Plaisant, editors. *IEEE InfoVis 2004 Contest, the history of InfoVis*, 2004.

[18] C. Hauff and L. Azzopardi. Age dependent document priors in link structure analysis. In *ECIR*, pages 552–554, 2005.

[19] P. Holme and B. Kim. Growing scale-free networks with tunable clustering. *Phys. Rev. E*, 65:026107, 2002.

[20] H. Jeong, Z. Néda, and A. Barabási. Measuring preferential attachment in evolving networks. *Europhys. Lett.*, 61(4).

[21] H. Jeong, B. Tombor, R. Albert, Z. Oltvai, and A. Barabási. The large-scale organization of metabolic networks. *Nature*, 407:651–654, 2000.

[22] B. Lee, M. Czerwinski, G. Robertson, and B. Bederson. Understanding eight years of InfoVis conferences using PaperLens. In *Posters Compendium of InfoVis 2004*, pages 53–54, 2004.

[23] S. Lehmann, B. Lautrup, and A. Jackson. Citation networks in high energy physics. *Phys. Rev. E*, 68:026113, 2003.

[24] P. Ormerod and A. Roach. The medieval inquisition: scale-free networks and the surpression of heresy. *Physica A*, 339:645–652, 2004.

[25] S. Redner. How popular is your paper? An empirical study of the citation distribution. *Eur. Phys. J. B*, 4:131–134, 1998.

[26] E. Rinia, T. van Leeuwen, H. van Vuren, and A. van Raan. Comparative analysis of a set of bibliometric indicators and central peer review criteria. Evaluation of condensed matter physics in the Netherlands. *Research Policy*, 27, 1998.

[27] H. White, B. Wellman, and N. Nazer. Does citation reflect social structure? Longitudinal evidence from the globenet interdisciplinary research group. *Journal of the American Society for Information Science and Technology*, 55(2):111–126, 2003.

[28] S. Wuchty. Scale-free behavior in protein domain networks. *Mol. Biol. Evol.*, 18(9):1694–1702, 2001.

[29] S. Yook, H. Jeong, and A. Barabási. Weighted evolving networks. *Phys. Rev. Lett.*, 86(25):5835–5838, 2001.

Optimal link categorization for minimal retrieval effort* †

Vera Hollink
Faculty of Science
University of Amsterdam
Amsterdam, The Netherlands
vhollink@science.uva.nl

Maarten van Someren
Faculty of Science
University of Amsterdam
Amsterdam, The Netherlands
maarten@science.uva.nl

ABSTRACT
Various studies have shown that categorizing search results can help users to retrieve their target pages faster. The categorizations save the users the time needed to consider links from irrelevant categories. However, what is often missed out is that the category selections also introduce extra effort for users who are looking for one of the highest ranked results. In general, the expected gain of categorization depends on the relative probability that the user is looking for each of the search results. In this work we present a method to balance the costs of presenting categories against the expected savings of the categorization. In an experiment we demonstrate that this method can reduce retrieval time substantially compared to flat result lists and hierarchies created through traditional hierarchical clustering.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.1.1 [**Models and Principles**]: Systems and Information Theory

General Terms
Algorithms, Experimentation

Keywords
Interactive information retrieval, Information gain, Hierarchical clustering, Active learning

1. INTRODUCTION
The explosive growth of the number of documents accessible via online information systems has intensified the need for navigation means that allow efficient access to the documents sets. Nowadays many navigation systems are publically available. Most of these use a form of keyword search or hierarchical menus. Search engines like Google [5] or AltaVista [2] and web portals like Yahoo! [25] aim at providing access to the whole web. Other systems are limited to documents on certain topics or to the contents of one web site. The common goal of these systems is to help the users reach their target information as fast as possible.

Providing access to web pages requires two steps. First, one has to determine which pages might be the user's target pages. The second step involves the creation of a suitable structure to present links to the candidate pages to the user. Research on search engines typically focuses on the first step: finding the set of pages that best matches the user's query. Recommender systems also address this step using a user profile instead of a search query. In this work we address the second step: automatically creating a hierarchy that allows users to efficiently access the candidate links.

Sets of candidate links can be structured in many different ways. Most search engines present their search results on a number of result pages containing ordered lists of links. In some cases search engines apply clustering techniques to group the search results per topic (e.g. [26, 15, 7, 27]). Deeper hierarchical structures are common in menus of single web sites and web directories. The question that arises is: which structures result in the shortest retrieval times?

The structures that are generated in this work are targeted at users with specific information needs. These users navigate through the provided information structure looking for the pages that are relevant for their purposes, their target pages. In this case the efficiency of a navigation structure is determined by the amount of browsing required to reach the target pages. In a flat list browsing consists of choosing the best link among a set of alternatives. In a hierarchy a series of categories need to be selected.

The optimal shape of a link hierarchy is not the same in all situations. The expected efficiency of a navigation structure depends on the probability that the candidate links are targets. If the system knows almost for sure that the user is interested in certain links, the best strategy is to show these links immediately. In other words, to structure the links as a flat list. On the other hand, if there are many links that have an equally small probability of being a target, a deeper hierarchy can be more efficient. Through the selection of categories in the hierarchy the user provides information about his or her target. This information is used to reduce the number of candidate links the user needs to consider.

In this work we present an algorithm that weights the time needed

*This research is supported as ToKen2000 project by the Netherlands Organization for Scientific Research (NWO) under project number 634.000.006.
†This paper is an extension of the work presented on the IJCAI'05 workshop on Intelligent Techniques for Web Personalization (ITWP'05)

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

to choose a category against the expected gain of providing extra information. At each step in the interaction with the user the algorithm computes the probability that each page is the user's target page. The categories and links with the highest expected information gain are presented. The resulting document hierarchy minimizes the number of clicks the user needs to make to reach his target pages. We demonstrate in a small scale experiment that users need less clicks when they use the hierarchies created by this method than when they use flat lists of links. Moreover, simulation experiments indicate that the created structures are more efficient than document hierarchies created through content clustering.

Section 2 discusses related work on optimizing document hierarchies. Section 3 describes the problem that is addressed in this work. In section 4 we discuss how the most informative sets of links and categories are selected. In section 5 we present the results of the experiments. The last section contains conclusions and discusses our results.

2. RELATED WORK

Related work can be classified into two categories. First, we give a brief overview of metrics for measuring the efficiency of hierarchical link structures. Afterwards, we discuss methods to automatically create and optimize these structures.

Many researches have studied the relation between of the structure of a hierarchy and the time that users need to retrieve items. A majority of the authors find a linear relation between retrieval time and the number of clicks necessary to reach a content page [10, 22, 14]. A linear relation is also the most common choice in models of web navigation [11, 14, 1]. The relation between retrieval time and the number of list items per hierarchy layer depends on the organization of the lists. In an ordered list users can use binary splits so that retrieval time is roughly logarithmic in the number of list items [10, 16]. In an unordered list the relation is linear [11, 22, 14].

Web search result clustering is a common method to assist users in finding relevant links among a set of retrieved web links. After a search engine has retrieved a set of documents matching a user's query, documents with similar contents are placed under a common header. Words that occur frequently in the clusters' documents are used as cluster labels. Several authors report that with result set clustering users need less time to find the relevant information (e.g. [26, 15, 7]). In [3] the documents are not clustered but classified into a predefined hierarchy. Zeng et al. [27] extract keyphrases from the documents and form clusters of pages containing the phrases. The top ranked clusters are labeled with corresponding keyphrases and presented to the user. The advantage of this method is that it yields both query specific clusters and high quality labels. These methods have in common that they all aim at optimizing the clusters' coherence and the clusters' descriptions. To our knowledge no attempts have been made to include the probability distribution over the links and optimize the clusters from an information theoretic perspective.

Other researchers focus on estimating the probabilities that pages are targets, e.g. [9]. Their methods improve the probability distributions over the pages which enables a better ordering of the links on the result pages. However, the improved probabilities are not used to create other structures than flat lists.

Several attempts have been made to select parts of a hierarchy that are interesting for certain users e.g. [18, 6]. These systems do not optimize the structure of the hierarchy, but only hide a part of it. Hiding nodes can improve efficiency as it allows users to reach their target pages without considering uninteresting parts of the hierarchy. However, the selected parts of the hierarchies are not necessarily the most efficient structures for the remaining nodes.

Various algorithms have been developed for improving existing hierarchies. Masthoff [12] presents an algorithm that creates a hierarchy using a number of ontologies as basic hierarchies. She uses hand crafted rules to split and merge menu items with too many or too few subitems. In [22] WAP menus are adapted to the usage of individual users. Frequently used items are moved to more prominent positions in the menu. For both methods the authors show that they can improve the efficiency of the hierarchies. However, there is not guarantee that they converge to maximally efficient structures. A method for which this guarantee can be given is presented by Witten et al. [24]. They optimize the index of a digital phonebook using the entropy of the probability distribution over the names. Their algorithm does make optimal decisions, but it only applies to domains in which the names of the searched items are known and ordered as in the case of a phonebook.

McGinty and Smyth [13] use critiquing to determine the users' targets. They argue that always presenting the links with the highest probability can cause a user to get stuck in an uninteresting part of the page space. They overcome this problem by uniformly spreading the presented links over the page space when the user seems to be making no progress. At each step they either try to maximize the probability of presenting a target or aim at collecting new information. With the approach presented in the present work one does not have to make this choice. The information gain criterion automatically results in broader categories when little is known about the user and in more specific links when more information becomes available.

3. PROBLEM SETTING

The task that is addressed in this work is to find a hierarchical structure for a set of links that minimizes a user's retrieval time. Before a system can accomplish this task it needs to compute for each page the probability that the page is a target page. Furthermore, the pages must be annotated with keywords that can serve as category labels. In this section we explain how the probabilities and keywords can be acquired in various situations. In addition, we present the interface that will be used for evaluation and discuss two popular structures that will serve as baseline structures in the experiments.

The constructed hierarchies consist of candidate links and categories. The candidate links form the terminal nodes in the hierarchy. They point directly to the candidate pages and have as anchors the pages' names. Non terminal nodes are categories. A user who selects a category goes a level deeper in the hierarchy and is presented with a new set of choices. A category node is labeled with a keyword or keyphrase that describes the contents of the pages below the node. Because all categories in the hierarchy must have a label, the available keywords determine the possible categorizations. Users navigate top down through the hierarchy opening links and categories that match their information needs. The task of the system is to place at each hierarchy layer the categories and links that minimize the average retrieval time. This task is depicted graphically in figure 1.

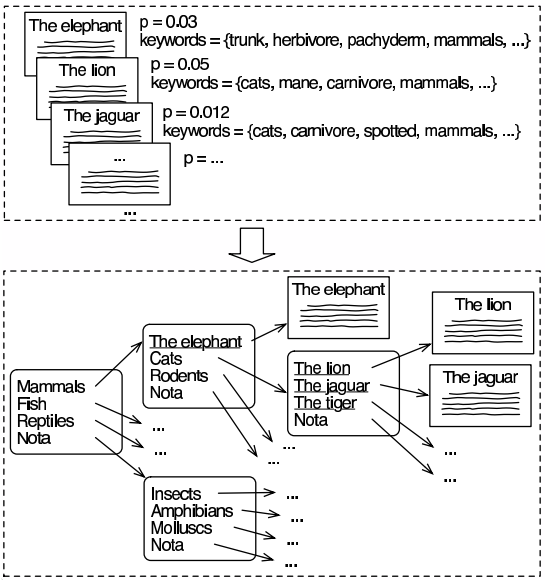

Figure 1: Example of a set of candidate pages with probabilities and keywords and a hierarchical structure for this set. 'Nota' is short for 'None of the above'.

The retrieval time of a given target page depends on the location of the page in the hierarchy. In correspondence with the literature (e.g. [10, 22, 14, 11, 14, 1]) we assume that retrieval time varies linearly with the number of clicks a user needs to make before reaching her target pages (the total path length). The length of the path to a content page is equal to the depth of the page in the hierarchy since users browse top down trough the hierarchy. We fix the number of links per hierarchy layer, so that we not have to make assumptions about the relation between retrieval time and the number of links per layer. Therefore, minimizing the average retrieval time reduces to minimizing the expected path length to the user's target pages.

The structures are built for a closed set of links called the *candidate links*. Some candidate links point to the user's target pages. The set of candidate links is available to the system, but the system does not know which links are targets. However, there is a probability assigned to each candidate link. How the probabilities are computed depends on the application. In a search engine the candidate links are the links that match the user's query. The target pages are the pages that the user finds relevant. The probabilities can be adapted from the relevance scores of the candidate links. For a recommender system the candidate pages are the pages of the web site the system is part of. In this case, the probabilities can represent the pages' access frequencies or the personal interests of the user.

To label the categories in the hierarchy a set of keywords is needed. The system also needs to know which keywords apply to which pages. For simplicity we assume keywords either apply or do not apply to a page, but the presented methods can be adapted straightforwardly to handle probabilistic keywords assignments. Keywords from various sources can be used to annotate the pages. If the candidate pages are already annotated with keyword meta tags, these keywords can be used directly. Otherwise, some keyword extraction mechanism needs to be deployed to extract keywords from the pages' contents (e.g. [27]).

The interface that enables the users to browse through the hierarchy can have different forms. Here we assume it looks similar to the result pages of search engines like Google [5] and AltaVista [2]. This means that users do not see the whole hierarchy, but only the choices corresponding to their current position in the hierarchy. Furthermore, the number of items that is shown in each step is fixed at n. It is always possible that none of the presented links and categories applies to the user's information needs. Therefore, there is always a choice labeled 'None of the above'. This choice is comparable to the link that points to the next result page in a search engine. When a user follows a link to a content page, he receives the page's contents. After reading the page he can stop the interaction or continue with a new search action.

We compare three strategies for determining which links and categories are shown at each hierarchy level. Search engines usually show flat lists of candidate links starting with the n most probable links. If the user's target is not among the presented links she clicks 'None of the above' and receives the next probable links. This structure maximized at each step the probability that the user can reach her target directly. For this reason we call it the *greedy strategy*. *Clustering based strategies* form hierarchies of page clusters on the basis of similarities between the pages. The keywords are used to label the clusters and create categories. In the next section we will present a new strategy called *the information gain strategy*. This strategy selects the links and categories with the highest information gain. It does not maximize the probability of showing a target link, but the information gained in each step.

4. THE INFORMATION GAIN STRATEGY

In this section we explain how the information gain strategy selects the most informative categories and links. In section 4.1 we discuss the computation of the expected information gain of a set of categories and links. Section 4.2 covers heuristics to find sets with high information gain.

4.1 Category Information Gain

In theory the most efficient structure can be determined completely. With the page probabilities we can compute the probability that a user is looking for a page from a certain category. If we make the assumption that users select with some probability the categories that contains their goal pages, we can compute the probability that a category is selected when it is presented to the user. We can write down all possible navigation traces for all possible structures and compute the lengths and probabilities of the traces. Now we just select the structure with the shortest expected path length.

This strategy always results in the optimal path lengths, but unfortunately it is not tractable in practice. We need a more efficient category selection algorithm, especially when all computation must be done while the user is waiting for his page. Often the structure can not be built in advance, for instance because the candidate set depends on a search query or because the target probabilities are adapted at run time to the user's interests.

To deliver the structures in reasonable time we create the structures top down only expanding nodes that are actually visited. Moreover, the navigation structures are optimized node by node instead of globally. At each step the system selects a set of links and categories without considering all possibilities for the deeper hierarchy layers. Below we explain the criterion according to which the information gain strategy selects the category sets. This criterion does not distinguish between links and categories. Links are treated as categories containing exactly one page.

To select the best category set we need to estimate the users' path lengths when a particular set of categories and links is presented. A measure which does exactly this is the *information gain* [19]. The information gain of a category tells us how much knowledge we have gained if the user selects the category. This depends both on the number of pages in the category and the candidate link probabilities. The expected information of a set of categories and links is the expected amount of information that is gained when the set is presented to the user.

The following example illustrates the working of the information gain criterion on the selection of categories for a set of pages about animals. If our only knowledge is that a user searches for an animal, broad categories like 'mammal' and 'fish' are informative. A click on one of these categories tells us in what kind of animal the user is interested. However, if for some reason we expect that the user is looking for a furry animal, the selection of 'mammal' does not provide much new information. In this case more information is gained by presenting narrower categories like 'rodent' and 'nocturnal animal'. Another important point is that showing two distinct categories like 'mammal' and 'fish' provides more information than showing largely overlapping categories like 'fish' and 'water animal'.

Formally, the information gain of a question is the difference between the number of bits of information needed to determine the target before and after asking the question. The expected information gain, IG, of a set of categories and links L is given by:

$$IG(L) = H(P) - \Sigma_{\{l \in L\}}(p(l|L) * H(P|l))$$

Here P is the probability distribution over the set of pages D. $p(l|L)$ is the probability that the user chooses category or link l provided that the items from L are presented. $H(P)$ gives the entropy of P. $H(P|l)$ is the entropy of the probability distribution after l has been chosen. $H(P)$ is given by:

$$H(P) = -\Sigma_{\{d \in D\}}(P(d) log(P(d)))$$

The distribution $P|l$ depends on the node type of l. If l is a link, the selection of l provides certain knowledge that the user was interested in following link l. In other words, the probability of l becomes 1 and the remaining entropy, $H(P|l)$, is 0. If l is a category, the selection of l does not provide certain knowledge about the user's target, but does provide evidence that the user's target belongs to category l. One possibility is to assign zero probability to all pages that are not annotated with the selected category. However, even if the keyword annotations are chosen carefully it can happen that a user finds that a page belongs to a category that is not present the annotation. Therefore we use an update mechanism that ensures that page probabilities are adjusted according to the selected categories, but never become zero. For details on this mechanism see [8].

The a priori candidate link probabilities are used to select the items that are shown at the root of the hierarchy. To create the deeper hierarchy layers the knowledge gained from the selected categories is incorporated in the probabilities. For instance, for the selection of the nodes below the category 'mammal', we use the probability distribution $P|mammal$ as base probabilities.

The information gain strategy selects the set of categories and links with the highest information. A high information gain means that the uncertainty that is left in the probability distribution is low. This indicates that after the users selects a category we need only a small number of steps to get perfect knowledge about the users target. Thus, selecting the category set with the highest information gain on average leads to the shortest path lengths for the user. Note, that this strategy leads to optimal categories for each step, but whether these choices are optimal overall depends on the categorizations available for the later steps.

4.2 Finding Informative Category Sets

The information gain criterion allows us to estimate how much a set of categories and links will shorten the path length without considering all possible continuations of the interaction. Unfortunately, this still does not make the problem tractable. If the number of links that can be presented on a page is n and the total number of categories and links is k, then the number of possible sets is n^k. Because this number can be prohibitively large, in this section we present heuristics to preselect some promising sets. The heuristics do not simplify the computation of the sets' information gain, but reduce the number of sets for which the information gain is computed.

As a first filter we throw out categories with a very small or very large probability of being chosen. If a category is associated with only one page it is obviously better to provide a direct link to the page than to show the category. Therefore, we compute for each category the probability that it contains a target page and throw out all categories with a probability smaller than the probability of the nth most probable page. Furthermore, if it is almost certain that the target belongs to some category, then selecting this category does not provide much new information. For this reason categories with a very large probability are also filtered out.

In a pilot study [8] we compared two heuristics for finding the best set among the categories and links that remain after filtering. The heuristic that proved most effective uses a form of hill climbing. It computes the information gain of all sets containing only one category or link. The n categories or links with the highest information gain are used as start set (n is the allowed number of links per page). One item from the start set is exchanged for another category of link. If this results in a set with a higher information gain the change is pertained; otherwise it is undone. This exchange process is repeated until no more changes can be tried or until a maximum number of steps is reached. The resulting set is presented to the user. Like all hill climbing methods this heuristic can converge to local maximum, but experiments show that in practice it finds good category sets.

5. EVALUATION

5.1 Experimental Setting

To evaluate the information gain strategy, the greedy strategy and the content clustering strategy we measure the efficiency of structures generated by each of the strategies in a series of experiments. In these experiments we use a fixed set of candidate pages and two versions of the probability distribution: a static distribution and a distribution that reflects the users' previous targets.

The candidate set is comprised of the combined sets of pages of two Dutch web sites for elderly people: the SeniorGezond site [21] and the Reumanet site [20]. Both sites were developed by The Netherlands Organization for Applied Scientific Research (TNO) in cooperation with domain specialists from the Geriatric Network and the Leiden University Medical Center. SeniorGezond contains

information about the prevention of falling accidents. Reumanet contains information about rheumatism. The sites have very similar structures: they consist of a set of short texts describing a particular problem or product and a hierarchically structured navigation menu. The menu provides information about the relations between the pages, but each each text is written in such a way that it can also be understood in isolation. From all pages of the two sites we removed the navigation menu and all in text links. Fifteen texts that were in almost the same form present on both sites were mapped onto one page. After this mapping 221 unique pages remained, each consisting of a title and some flat text.

A semi-automatic method was used to assign keywords to the pages. We manually created a domain specific ontology consisting of 800 terms or phrases and a broader term - narrower term relation. The terms from the ontology were automatically assigned to the pages. We counted for each text and each term in the ontology the evidence that the term was a keyword for the text: the number of times the term or one of its descendants appeared in the text. The pages were annotated with all terms with an evidence of at least 2. The domain specific ontology was created by hand, because there was no ontology available for the domain and many of the domain specific keywords were not in the Dutch version of WordNet [4]. The average number of keywords of a page was 7.7.

The quality of the keywords was evaluated in a survey [8]. We found that on average the participants labeled 36% of the keywords in the annotation as not appropriate for the texts. Apparently the precision of the annotation procedure was not very high. Using these keywords as category labels in a navigation structure will probably cause the users to follow a considerable amount of incorrect paths. In the next section we will see how this effects the efficiency of the information gain strategy. Another interesting finding from the keyword evaluation was that there was 80% agreement between the answers of the various participants. This suggests that it is possible to learn the associations between pages and keywords from the behavior of the users. This allows a system to automatically improve the pages' annotations. We plan to explore this idea further in the future.

To decrease the influence of the chosen probability distribution we tested structures that were generated with two different distributions. The first version is a static distribution. In the server logs of the two web sites we counted the number of requests for the candidate pages. From the request frequencies we constructed the a prior probabilities. When users searched for more than one target page, for each search the same a priori probabilities were used to built the navigation structure. Thus, each search started at the root of the same hierarchy.

For the second distribution we used a form of personalization. Each time a user reached a content page the probabilities of the candidate pages were adapted. When the users decided to search further the adapted probabilities were used to built a new navigation structure. The new searches still started at the root of the hierarchies, but the hierarchies became more and more personalized. The personalization process increased the probability of pages that were similar to the visited content page and decreased the probability of dissimilar pages (for details see [8]). The similarity between two pages was computed as the minimal conditional probability of the pages [17]. Like the a priori probabilities the conditional probabilities were taken from the server logs. For the distances of pages from different sites we used a content based measure.

Task: glasses and contact lenses

You have difficulty reading and you think you might need glasses. Find as much information as possible on (buying) glasses and contact lenses.

Target pages: Optician.htm
Seeing+and+hearing.htm

Figure 2: Translated example task with target pages. The target pages were not visible to the participants.

We defined 12 search tasks for which information could be found in the candidate pages. A task consisted of a short description of a specific problem of an elderly person. The users had to search all pages related to the problem. The topics of the tasks were chosen after consultation of the creators of the sites. We tried to choose problems that were realistic in the domain to get a realistic simulation of the site's users. We defined by hand which pages were in the target sets for the tasks. The tasks had between 2 and 12 target pages. A example of a task description is shown in figure 2.

5.2 Experiments

To show the advantage of the information gain strategy over the greedy strategy and the clustering based strategy we performed a series of experiments. The three strategies were used to build hierarchies on the basis of the static and the personalized probability distributions. This resulted in five structures: a greedy structure, a personalized greedy structure, a clustering based structure, an information gain structure and a personalized information gain structure. The structures were not built in advance, but expanded each time a user opened a node.

The structures were built as described in the previous sections. The greedy structures consisted of ordered lists of pages. The most probable pages were located at the root of the hierarchy, the next most probable pages at the second level etc. For the creation of the clustering based structure we used a divisive form of k-means comparable to the bisecting k-means algorithm of [23]. This algorithm split the set of pages at each level in a number of clusters. For each cluster the best matching keyword was used as category label. Pages that did not belong to one of the resulting categories were placed under the 'None of the above' category. The similarity measure was the same one we used for personalization in section 5.1. The clustering based structure was constructed only on the basis of the page similarities and did not discriminate between the two probability distributions. The information gain hierarchies had on each node the most informative categories.

We created an interface that allowed users to browse through the hierarchy. The interface functioned as explained in section 3. The number of links or categories on a page was set at 5, including the 'None of the above' category. This number was purposely chosen to be quite low, so that many clicks would be required to reach the targets. In a perfectly balanced hierarchy 221 pages are reachable in 4 steps. In a greedy structure the 221 pages are divided over 56 result pages.

In the first part of the experiment we evaluated the efficiency of the structures with simulated users. The simulated users had a set of pages which were their target pages. They traversed the hierarchy looking for their targets. They never went to content pages which

Method	No. steps
Greedy	27.7
Personalized greedy	9.0
Clustering	15.1
Information gain	8.2
Personalized information gain	4.6

Table 1: The average number of steps of simulated users with perfect choices.

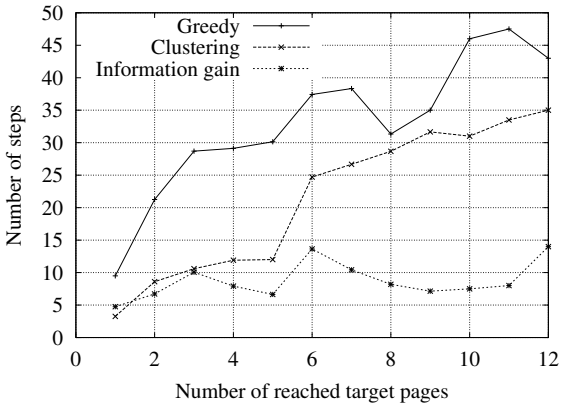

Figure 3: The average number of steps that simulated users with perfect choices needed to reach each of the target pages.

were not in their target set and when a link to a target page was available they always went there directly. When no links to target pages were available they considered the available categories. If one of the categories matched a target, they opened the category. When also no relevant categories were shown, they clicked 'None of the above'. They kept searching until all targets were found.

We compared simulated 'perfect' users that always chose the correct categories with 'imperfect' users that sometimes followed an incorrect path. The imperfect users sometimes chose categories that were not related to their targets or 'None of the above' when an appropriate category was presented. We did not add noise to the content link choices, because we assumed that users could accurately judge the relevance of pages from their titles.

We evaluated the real world value of the greedy and the information gain structures in an experiment with real users. Thirteen participants were asked to perform all 12 multiple target search tasks. The participants got only the topics of the tasks and not the sets of target pages. Every participant used two of the four structures, one during the first 6 tasks and another during the next 6 tasks. The order of the tasks and the structures was varied over the participants. We measured the number of clicks the users needed to find the targets and the number of relevant pages that were found.

5.3 Results

5.3.1 Simulation

Table 1 gives the average number of clicks that the simulated users needed to reach their targets. All figures are averages over 25 runs. With both probability distributions perfect users needed a significantly[1] smaller number of steps in the information gain structure

[1]In the simulation experiments significance is computed with a two tailed paired t-test with a confidence level of 0.95.

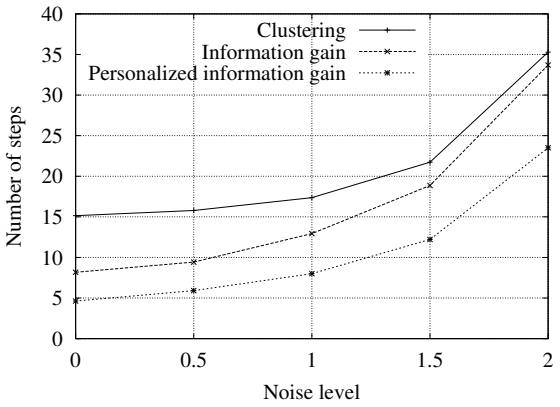

Figure 4: The average number of steps of simulated users with various noise levels.

than in the greedy structure and the clustering structure. Personalization made the greedy and the information gain strategy significantly more efficient. With personalization the difference between the two strategies is smaller, but the information gain strategy is still 49% faster. Personalization does not effect the clustering structure, because the clustering strategy does not use the page probabilities.

Figure 3 shows the number of steps that were needed to reach the various target pages. The greedy and the clustering structures provided short paths to the first targets, but the when the users searched further the paths became very long. The path lengths in the information gain structure were more stable.

In the next experiments we looked at the behavior of users with various amounts of incorrect choices. The results are presented in figure 4. In this figure a noise level 1 correspond to the values found in the keyword evaluation survey (see section 5.1). The incorrect categories had a probability of 0.013 to be opened. The probability of clicking 'None of the above' when an appropriate category was presented was 0.36. For the other noise levels these values were multiplied by the noise level. The efficiency of the greedy structures is not shown. These structures do not use categories and therefore are insensitive to the type of noise that was added.

Figure 4 shows that the efficiency of the category structures decreased rapidly when the users made more mistakes. At the highest noise levels the path lengths become even longer than the path lengths in the greedy structures. The information gain structures performed better than the clustering structure at all noise levels. However, the influence of the amount of incorrect choices appeared to be much larger than the influence of the type of navigation structure. This suggest that information value is a useful criterion to choose between categories with equally good labels, but that the highest priority must be to given to finding categories with high quality labels.

5.3.2 Human Search

Table 2 and Figure 5 show the results of the experiments with real users. Users of the information gain structures needed significantly[2] less steps to reach the targets than users of the greedy structures.

[2]In the experiments with real users significance is computed with a two tailed t-test with a confidence level of 0.95.

Method	No. steps	No. targets
Greedy	17.8	0.9
Personalized Greedy	9.8	1.7
Information gain	11.1	1.5
Personalized information gain	6.9	1.4

Table 2: The average number of steps of human users and the average number of targets that were found by human users.

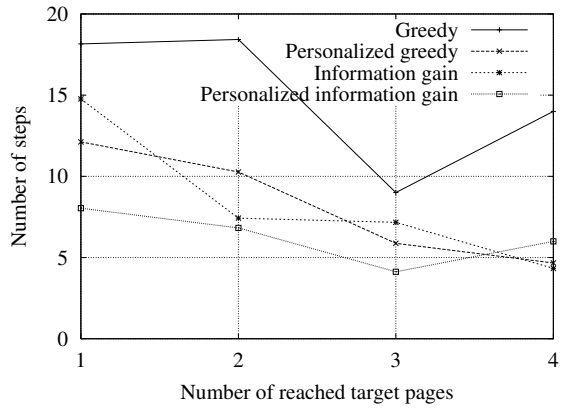

Figure 5: The average number of steps that human users needed to reach their target pages.

The relevance of the categories in the information gain structure was not always clear to the participants which led to suboptimal paths. 17% of the chosen keywords were not in the target pages' annotations. In 7% percent of the cases in which 'None of the above' was clicked, there actually was a relevant category among the presented keywords. Because of these 'mistakes' the participants' paths were much longer than the optimal path lengths measured in the previous section. However, the shorter path lengths of the information gain structure show that even with these high noise levels categorization can be effective.

Personalization significantly reduced the number of steps of both the greedy and the information gain structure. In contrast to what we saw in the simulation experiments Figure 5 shows that personalization not only helped during the later stages of the search, but also reduced the number of clicks needed to find the first target. This is caused by the fact that the real users sometimes clicked on links to pages that were not relevant for the task. Apparently, according to our distance function these pages were close to the target pages, so that the adaptive strategies could lead the users efficiently from these pages to the nearby targets.

We did not find large differences between the numbers of target pages that were found. The only significant result was that users found more targets when assisted with the personalized greedy structure than with the greedy structure. Probably users of the greedy structure were tempted to give up when they saw that they would have to go through the same lists of links again. With all structures the users found very few targets. Most likely this is a consequence of the limited interface. Many participants reported that they had trouble judging how many relevant pages were available, because the interface did not provide an overview of the candidate pages. This problem is less likely to occur on real sites where more information is available like the number of search result.

In conclusion, the simulation experiments showed that maximizing the information gain reduces the length of the paths to the users' target pages provided that the category labels are sufficiently clear. The experiments with human participants show that users are able to make effective use of keyword structures and thus need less clicks in an information gain structure than in a greedy structure. In this work we used a simple method to compute the page probabilities, but the information gain criterion can be used without modification on top of more advanced page probability estimators. Our findings suggest that in any case maximizing the information gain will effectively balance the collection and exploitation of knowledge and so minimize the users' path lengths.

6. CONCLUSION AND DISCUSSION

In a variety of domains systems create hierarchical structures for sets of candidate pages. Search engines and recommender systems typically return series of pages with flat lists of links. At each page they maximize the probability of showing a target link by presenting the most probable candidate links. They focus entirely on using their current knowledge about the user to determine which pages are the most likely targets. In other words, they follow a greedy strategy. Clustering based method do not only show links but also categories. However, these categories are not chosen to minimize the users' path lengths, but to group the most similar pages.

In this work we present a method that actively minimizes the length of the user sessions balancing the costs of collecting more information by showing categories against the expected gain of the extra knowledge. Evaluation with artificial and experimental data shows that this information gain strategy effectively reduces the users' numbers of clicks compared to the greedy and the clustering based strategy.

The advantage of maximizing information gain is independent of how we estimate the probabilities that pages are targets. In this work we used simple algorithms for estimating the page probabilities. More advanced methods, such as the one presented in [9], can make the estimations more accurate which leads to more efficient structures. However, these better estimations improve both the greedy and the information gain structure. To demonstrate this we tested structures generated on the basis of two probability distributions: a static and a personalized distribution. Experiments showed that the extra knowledge provided through personalization reduced the number of clicks in the greedy structure as well as the information gain structure. Improving the estimation accuracy improves efficiency but does not lessen the need for active knowledge collection.

The number candidate pages that we used for the experiment was quite small compared to the number of pages of an average web site or the average number of search results. To be able to measure the effect of categorization the number of links or categories per page was also kept low. Although the theoretical advantage categorization is independent of the size of the domain, more research is needed to show the practical value of the information gain strategy in realistic domains. We are currently incorporating the information gain and the greedy strategy in recommender systems that will be included in the real version of the SeniorGezond site. Running these systems in parallel allows us to compare the value of the strategies in a real world application.

Until now we have not considered the order in which the links are shown on the pages. Especially when the lists are long this is not

realistic, because users do not always consider all available options before clicking a link. This means that on average users need less time to choose top ranked links than to choose links at lower positions. Greedy strategies accommodate for this phenomenon by ordering the links according to their probability. The information gain strategy does not currently include link order. It can present the categories in order of probability, but how the categorization itself should be adapted is an open issue.

The candidate pages in the evaluation were annotated with terms from a manually created ontology. Many of the keywords in the annotations were ambiguous so that the participants made a considerable amount of suboptimal selections. We showed that despite these 'mistakes', the categorizations chosen by the information gain strategy shortened the users' path lengths. However, simulation experiments demonstrated that when users make too many mistakes presenting categories can reduce efficiency. These results suggest that a category's label quality is at least as important as its information value. Consequently, in a real application one should be careful only to use categories for which an adequate label is available. We are currently exploring possibilities to automatically determine the quality of category labels.

7. REFERENCES

[1] J. Allan, A. Feng, and A. Bolivar. Flexible intrinsic evaluation of hierarchical clustering for tdt. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 263–270, New Orleans, USA, 2003.

[2] AltaVista search engine. http://www.altavista.nl.

[3] H. Chen and S. Dumais. Bringing order to the web: automatically categorizing search results. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 145–152, The Hague, The Netherlands, 2000.

[4] EuroWordNet. http://www.illc.uva.nl/eurowordnet/.

[5] Google search engine. http://www.google.com.

[6] P. Haase, A. Hotho, L. Schmidt-Thieme, and S. Y. Collaborative and usage-driven evolution of personal ontologies. In *Proceedings of the 2nd European Semantic Web Conference*, pages 486–499, Heraklion, Greece, 2005.

[7] M. Hearst and J. Pedersen. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *Proceedings of the 19^{th} Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 1996.

[8] V. Hollink, M. Van Someren, S. Ten Hagen, and B. Wielinga. Recommending informative links. In *Proceedings of the IJCAI-05 Workshop on Intelligent Techniques for Web Personalization (ITWP'05)*, Edinburgh, UK, 2005.

[9] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142, Edmonton, Canada, 2002.

[10] T. Landauer and D. Nachbar. Selection from alphabetic and numeric menu trees using a touch screen: Depth, breadth and width. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 73–78, San Francisco, USA, 1985.

[11] E. Lee and J. MacGregor. Minimizing user search time in menu retrieval systems. *Human Factors*, 27(2):157–162, 1985.

[12] J. Masthoff. Automatically constructing hierarchies: An algorithm and study of human behaviour. *Forthcoming*.

[13] L. McGinty and B. Smyth. Tweaking critiquing. In *Proceedings of the Workshop on Intelligent Techniques for Personalization as part of The Eighteenth International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, 2003.

[14] G. Miller and R. Remington. Modeling information navigation: Implications for information architecture. *Human-Computer Interaction*, 19:225–271, 2004.

[15] R. Osdin, I. Ounis, and R. White. Using hierarchical clustering and summarisation approaches for web retrieval: Glasgow at the TREC 2002 interactive track. In *Proceedings of the Eleventh Text REtrieval Conference*, Gaithersburg, USA, 2002.

[16] K. Paap and R. Roske-Hofstrand. The optimal number of menu options per panel. *Human Factors*, 28(4):377–385, 1986.

[17] M. Perkowitz and O. Etzioni. Towards adaptive web sites: Conceptual framework and case study. *Artificial Intelligence*, 118:245–275, 2000.

[18] D. Pierrakos and G. Paliouras. Exploiting probabilistic latent information for the construction of community web directories. In *Proceedings of the 10th International Conference on User Modeling*, pages 89–98, Edinburgh, UK, 2005.

[19] J. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[20] Reumanet. http://www.reumanet.nl/.

[21] SeniorGezond. http://www.seniorgezond.nl/.

[22] B. Smyth and P. Cotter. Intelligent navigation for mobile internet portals. In *Proceedings of the IJCAI'03 Workshop on AI Moves to IA: Workshop on Artificial Intelligence, Information Access, and Mobile Computing*, Acapulco, Mexico, 2003.

[23] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *Proceedings of the KDD Workshop on Text Mining*, Boston, USA, 2000.

[24] I. Witten and J. Cleary. On frequency-based menu-splitting algorithms. *International Journal of Man-Machine Studies*, 21:135–148, 1984.

[25] Yahoo! web directories. http://dir.yahoo.com/.

[26] O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to web search results. In *Proceedings of the Eighth International World Wide Web Conference*, Toronto, Canada, 1999.

[27] H. Zeng, Q. He, Z. Chen, W. Ma, and J. Ma. Learning to cluster web search results. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 210–217, Sheffield, UK, 2004.

Focused Access to Wikipedia

Börkur Sigurbjörnsson[1] Jaap Kamps[1,2] Maarten de Rijke[1]
[1] ISLA, Faculty of Science, University of Amsterdam
[2] Archives and Information Studies, Faculty of Humanities, University of Amsterdam
{borkur,kamps,mdr}@science.uva.nl

ABSTRACT

Wikipedia is a "free" online encyclopedia. It contains millions of entries in many languages and is growing at a fast pace. Due to its volume, search engines play an important role in giving access to the information in Wikipedia. The "free" availability of the collection makes it an attractive corpus for information retrieval experiments. In this paper we describe the evaluation of a search engine that provides focused search access to Wikipedia, i.e., a search engine which gives direct access to individual sections of Wikipedia pages.

The main contributions of this paper are twofold. First, we introduce Wikipedia as a test corpus for information retrieval experiments in general and for semi-structured retrieval in particular. Second, we demonstrate that focused XML retrieval methods can be applied to a wider range of problems than searching scientific journals in XML format, including accessing reference works.

1. INTRODUCTION

Wikipedia [14] is a "free" online encyclopedia that can be edited by anyone. At the time of writing (February 2006), it contains a million articles in English as well as millions of articles in several dozens of other languages. Given the volume of the data, search engines provide an important tool for accessing the information contained in Wikipedia.

There are quite a few search facilities for Wikipedia [16]. The search engines differ both with respect to the indexing scheme and result presentation used. Some engines search over the full content while others only search over the title. Some engines display links to pages, with or without text snippets, while other engines cluster results by category.

In the area of semi-structured retrieval, focused information access has gained much attention, with direct access to relevant parts of documents being an important example. This is one of the major research issues addressed within the Initiative for the Evaluation of XML Retrieval (INEX) [5]. In a previous study, performed as part of the INEX interactive track [7], we evaluated focused access to scientific literature [6]. In the evaluation we used a home-grown XML retrieval interface developed in a student project [1]. The evaluation was also carried out in a student project.

In this paper we describe how we have adapted the XML retrieval interface to provide focused search access to Wikipedia [13]. We describe the system and its evaluation. The main goal of the experiment is to investigate the usefulness of focused access to information. More precisely, we explore the usefulness of giving users access the Wikipedia pages at individual section level, as opposed to page level only.

Our main findings are that users are positive toward focused information access. The main advantage of the section level access is that the users finish their search tasks in less time. Additionally, our experiment revealed that users access the Wikipedia pages equally via search result lists and via browsing within the encyclopedia itself.

The remainder of the paper is organized as follows. In Section 2 we survey Wikipedia and its use as a corpus for information retrieval experiments. In particular, we zoom in on how we use it in this paper. We introduce our Wikipedia search engine in Section 3. In Section 4 we describe the evaluation setup, and in Section 5 we present evaluation results. We conclude and describe future work in Section 6.

2. WIKIPEDIA AS A CORPUS

Wikipedia is *free* in the sense that its contents are written by web users and can be edited by any other web user. As a document collection, Wikipedia has many properties that make it attractive as a corpus for performing information retrieval experiments. For a start, the document collection is freely available, which makes it easy to distribute as part of a test collection. The multi-lingual aspects of the collection support a range multi-lingual retrieval efforts. Furthermore, the semi-structured format of the collection makes it usable for evaluation of semi-structured retrieval techniques, such as those developed for XML element retrieval. And last but not least, the dense link structure of the collection makes it interesting for investigating the interplay between searching and browsing when users seek information [3].

There are several information retrieval evaluation initiatives that plan to use Wikipedia as their corpus. At CLEF 2006 a pilot task will be running where Wikipedia is used as a corpus for question answering [17]. Within the INEX initiative there are ongoing efforts to convert the wiki markup language into a standard XML format, and use the corpus for the evaluation of ad-hoc XML retrieval [5]. In this study, we complement these initiatives by using Wikipedia as a corpus for an interactive experiment.

3. WIKIPEDIA SEARCH ENGINE

In this section we detail our Wikipedia search engine. We index and search Wikipedia using our XML retrieval system [10]. Although the content of the Wikipedia pages is

Proceedings of the sixth Dutch-Belgian Information Retrieval workshop (DIR 2006)

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 to identify section boundaries and nesting levels.

3.1 Retrieval Engine

Our XML retrieval system is based on our home-grown extension of Lucene [8, 4]. The engine uses a simple multinomial language model to rank each indexing unit, in our case individual sections, with respect to relevance to the user's query. For now, no advanced XML specific retrieval methods are used. For example, we have found mixture models useful for ranking XML elements [11], but it remains as future work to make the implementation efficient enough for online usage.

3.2 Indexing Wikipedia

Since the content of Wikipedia pages is not marked up in XML, we created a simple parser for the Wikipedia syntax which allowed us to index the collection as if the pages were stored as XML. 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.

We also extracted and indexed two types of additional fields. Titles of pages and sections were indexed using the 'fields' mechanism of Lucene [8]. For each Wikipedia page we also extracted its categories and indexed as a separate field (of the page on which it occurs). These fields were not used in our current evaluation efforts.

We index the whole Wikipedia distribution package. This means that in addition to the "proper" encyclopedia pages we also index redirect pages and various log-pages. All included, we index 2,086,197 pages which are divided up into 4,095,103 indexing units.

3.3 Wikipedia Search Interface

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, while the other is a focused interface which gives access to individual sections of Wikipedia pages.

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 1.

Our focused Wikipedia search interface is based on our XML retrieval interface xmlfind [1, 6]. A screen-shot of the interface can be seen in Figure 2. We group the retrieved sections and subsections by the wiki page that they belong to. Hence, the main addition of the focused interface, compared to the baseline interface, is that the snippets are broken up by section boundaries, and hyper-links give access to individual sections.

The section-based linking and section-based snippets are the only difference between the two 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.

3.4 Logging User Interaction

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.
- *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.

In Section 5 we describe how we use the collected data to answer the research questions that we will discuss in the next section.

4. EVALUATION

4.1 Research Questions

The goal of the experiments described in this paper is to gain a better understanding of the way in which users interact with a focused retrieval system. Our main research question, then, is:

> Do focused retrieval methods improve users' access to Wikipedia, compared with more traditional document retrieval methods?

Even before exploring this issue, we had reasons to believe that it would be difficult to come to a positive answer to this question if we look at the problem from a broad perspective. I.e., 'on average' the systems are likely to be rather similar. One of the main reasons for this 'pessimism' was that many of the Wikipedia entries are not very long. It is even the nature of Wikipedia as an encyclopedia to create a new separate entry for a 'sub-entry' that gets too long [15]. This means that for many entries there will be no, or little, difference between the two interfaces.

We believe, however, that there exist cases where focused access might be more useful. That is, if the information need is specific, and is satisfied by some text buried deep in a long entry. Hence, we reformulated our research question to a stronger question:

> Do there exist scenarios in which focused access improves users' access to Wikipedia, compared with a more traditional document access?

We believe that the search scenario plays a crucial role when the effect of this sort of focused retrieval is evaluated.

Although focused access is the main goal of this exercise, we will gather data of other interesting user behavior. One interesting aspect in the case of Wikipedia is the interplay between searching and browsing [3]. Wikipedia has very dense linking between entries. Since we keep an extensive log of user interaction with our system we are able to formulate a 'bonus' research question related to the link structure:

> What is the interplay between searching and browsing when users interact with densely hyper-linked sources such as Wikipedia?

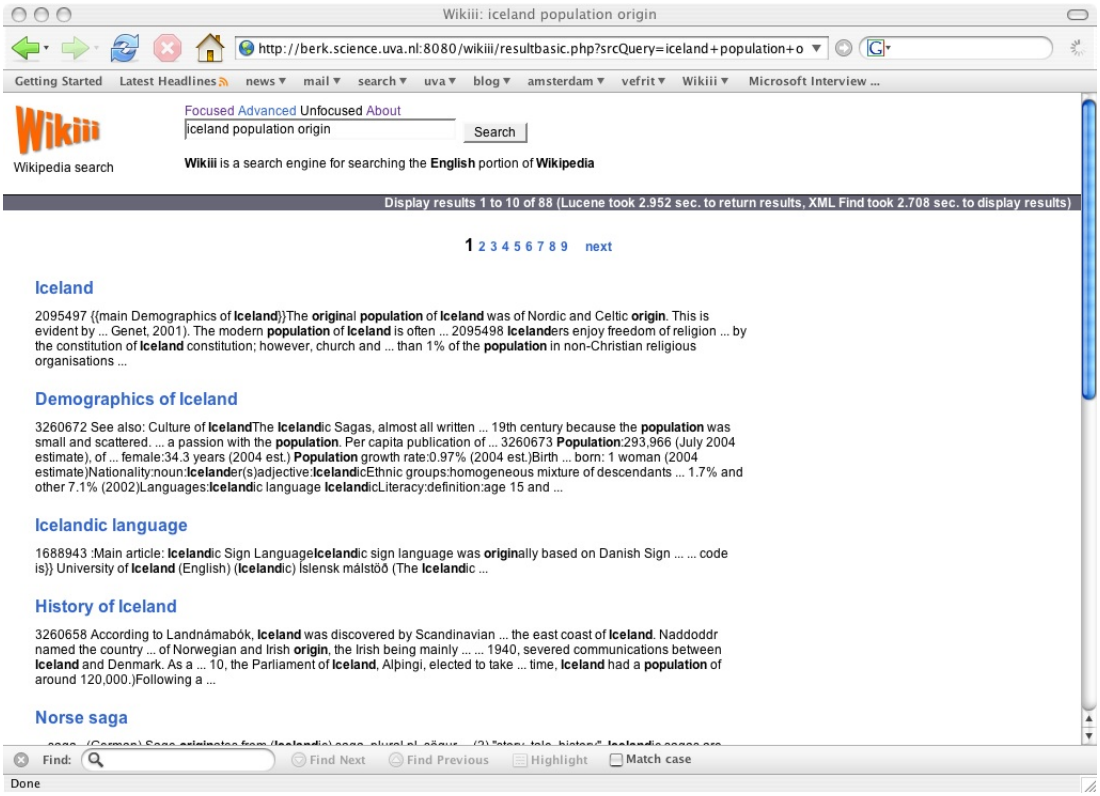

Figure 1: Baseline interface

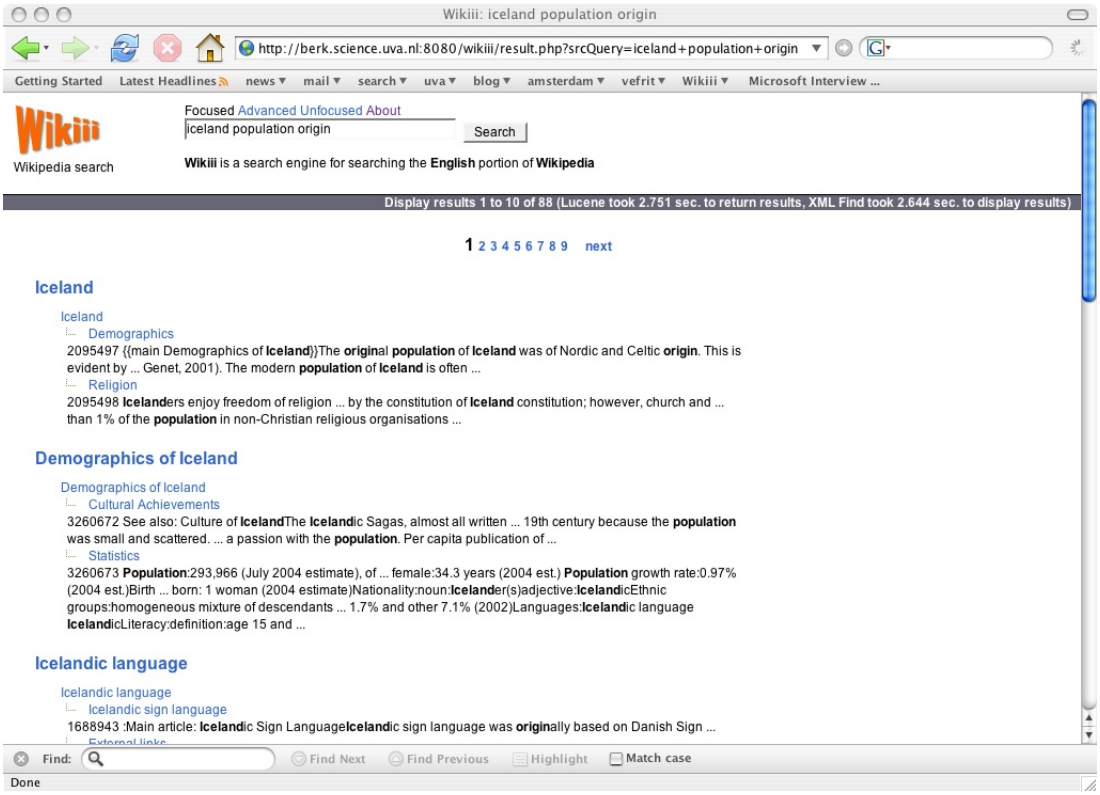

Figure 2: Focused interface

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 3: Example of a possible simulated work task.

Table 1: Experimental matrix for the interactive experiment.

Rotation	Task I	Task II
1	Baseline	Focused
2	Focused	Baseline

4.2 Experimental Setup

In order to answer our research questions we set up an interactive experiment where we asked people to perform simulated work tasks [2]. An example of a simulated work task can be seen in Figure 3. The actual work tasks that were used in the experiment can be found in Figures 7 and 8 in the appendix of this paper. Each of the actual work tasks consisted of three related search assignments. Each search assignment resembled a factoid question or a list question.

Each test subject performed two simulated work tasks, but using different system each time. The experiment matrix is shown in Table 1. Our analysis is based on 12 test persons, evenly distributed between the two rotations.[1]

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:

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

[1]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.

Table 2: Responses on user *satisfaction*: Mean rating and standard deviation (in brackets). Answers were on a 5-point scale, ranging 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)

Table 3: Responses on user *effort*: Mean rating and standard deviation (in brackets). Answers were on a 5-point scale, ranging 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)

5. RESULTS

We start by reporting on the user search experience while using our systems. These results are based on an analysis of the responses to the post-task questionnaires. We will then look at how users interacted with our system by mining the system interaction logs. Finally, we discuss the results in relation to the research questions stated in Section 4.

5.1 User Search Experience

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 2. 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 3. It is interesting to note that in solving the first task, the users rated the baseline system as easier to use. However, in solving the second task, the users rated the focused system as easier to use. Overall, there is very little difference between the two systems.

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 questions: How well do you find this system suitable for specific questions?

Table 4: Responses on system suitability for answering *specific questions*: Mean rating and standard deviation (in brackets). Answers were on a 5-point scale, ranging 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)

Table 5: Responses on system suitability for answering *general questions*: Mean rating and standard deviation (in brackets). Answers we on a 5-point scale, ranging 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)

Table 4 shows how users rated the system's suitability for answering specific questions. The users find 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 questions: How well do you find this system suitable for general questions?

Table 5 shows how suitable the users rated the system's suitability for answering general questions. Now, both systems get a rating better than "neutral". 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 [9].

5.2 User Interaction

We explore the user-system interaction by mining the interaction logs provided by the systems. Let us first look at the number of queries posted. Table 6 shows the mean number of queries issued in each search task. There is not much difference between users of the different systems. Next we look at the number of wiki pages viewed in each search task. Table 7 shows the mean number of pages viewed. This number includes all pages viewed, both via search results and via browsing within the Wikipedia site. Overall, users view more pages when using the focused system than when using the baseline system. The difference is not significant, however. If we look at the individual tasks, we see that we get different results. For Task I, more pages were visited by users of the baseline system. For Task II, the users of the focused system view more pages. In this case, the difference is significant (t-test: $p < 0.05$).

Table 7: Page views per search task: Mean number of page views and standard deviation (in brackets). Each search task was divided into three distinct search assignments.

	Task I		Task II		Overall	
Baseline	19.2	(12.4)	16.3	(4.6)	17.8	(9.0)
Focused	15.5	(8.1)	26.0	(8.0)	20.8	(9.4)

Table 8: 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)

Users of the focused system seem to spend more effort in terms of queries and page views, but what about time? Table 8 shows the average number of minutes needed to complete each search task. We see that despite all the page views, the users of the focused system finish their tasks quicker than the users of the baseline system. The difference is not significant, however.

Let us zoom in now on the interaction with the focused interface. Recall from Figure 2 that there are two types of links in the focused interface: *page-links* that bring you to the beginning of 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 9 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 4 shows the ratio between page-link and focused-link clicks for each user. We see that the click-behavior is very user dependent.

Let us now take a closer look at the focused-links that were clicked. How deep into the documents do users dive? Table 10 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.

Table 6: Queries per search task: Mean number of queries and standard deviation (in brackets). Each search task was divided into three distinct search assignments.

	Task I		Task II		Overall	
Baseline	11.33	(6.53)	8.67	(2.50)	10.00	(4.92)
Focused	12.50	(5.47)	9.50	(5.05)	11.00	(5.26)

Table 9: 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)

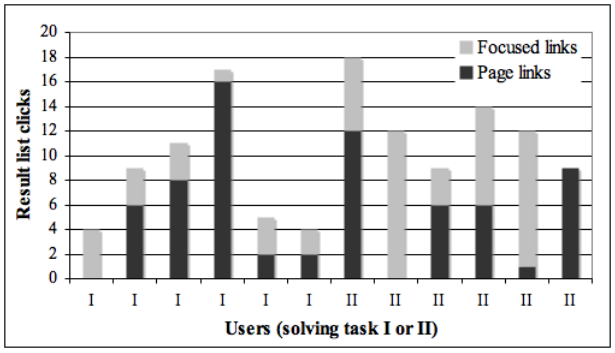

Figure 4: Number of result clicks per user in the focused interface. Dark: Clicks on page-links. Light: Clicks on focused-links.

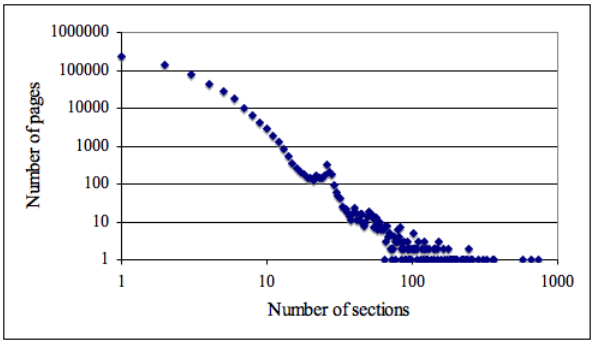

Figure 5: Linear depth of Wikipedia pages which have one or more sections. The distribution of pages over the number of sections is plotted on a log-log scale.

Table 10: 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%

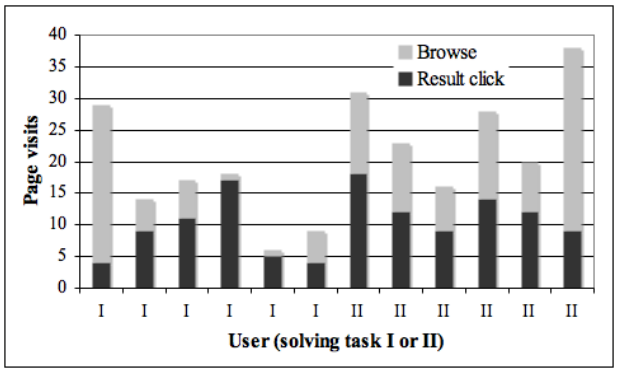

Figure 6: Number of page visits per user in the focused interface. Dark: Pages visited via the result list. Light: Pages visited via internal links.

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 are divided up into sections. Of these pages 224,000 have only one section, and 140,000 have two sections. Figure 5 shows the distribution of pages, based on the section count.

An important characteristic of Wikipedia is that the text is densely populated with hyper-links to other pages within the collection. Hence, it is important to see how users use these links as part of their information seeking behavior. In particular, it is of interest to see the ratio between pages visited via the search result list and pages visited via the internal link structure of Wikipedia. This ratio can be seen in Figure 6. Overall, 124 pages were reached via the search result list, while 125 were reached via internal links. The ratio is thus half-half. The ratio is slightly in favor of result visits for Task I and in favor of internal browsing for Task II.

5.3 Discussion

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 the 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.

Let's now recall our research questions as stated in Section 4. Our first research question read:

> Do focused retrieval methods improve users' access to Wikipedia, compared with more traditional document retrieval methods?

If time is an issue, the focused retrieval methods are promising. Users felt that they could find the right information quicker when using the focused system. This feeling is confirmed by the interaction log files.

It must be noted that most of the search assignments were about finding answers to factoid questions. That is, the assignments were aimed at satisfying "specific" information needs. Hence, our study provides evidence for the claim that focused retrieval methods are useful for "specific" information needs.

Next, let us look at our bonus question:

> What is the interplay between searching and browsing when users interact with densely hyper-linked sources such as Wikipedia?

In our experiment, page visits were evenly distributed between searching and browsing. The popularity of browsing was beyond our expectation. Earlier studies reported little interaction with the search results [12]. This issue deserves more attention. In future work, it might be interesting to go deeper into the role of browsing. Why do users browse?

Because they did not find the answer on the current page? Because they wanted to get broader support for their answer? Or even because they got distracted by an interesting hyper-link that was unrelated to their actual search assignment?

6. CONCLUSIONS AND FUTURE WORK

Wikipedia is an attractive corpus for performing information retrieval experiments. In this paper we described how it can be used to evaluate focused retrieval in an interactive experiment. One of our main findings is that focused access allows users to solve their search task quicker, at least when the information need is specific. Another main finding, derived as a by-product of our study, is that in a richly hyper-linked environment, users access pages equally via search result lists and via internal browsing. We believe that the interaction between searching and browsing deserves further study.

There are many options for extending the work in this paper. For the focused retrieval part, the outcome of the interactive experiment gives strong support to the effort to create a reusable system-oriented test collection based on Wikipedia. The first steps in this direction have already being taken within the INEX community. Focused information access to richly structured corpora also allows for retrieval using more expressive queries in which a user can combine content with structural constraints. With the creation of an XML version of Wikipedia this task becomes particularly interesting. Yet another form of focused information access is automatic question answering based on Wikipedia. Work on that task is already underway within the WiQA task at CLEF 2006. If we look beyond focused retrieval, Wikipedia is also a promising resource for evaluating multilingual retrieval, which will be (partly) addressed in the WiQA task.

Acknowledgments

Many thanks to the students of Project Information Retrieval who set up and carried out the interactive experiment: Monique Arlaud, Deborah Huijsman, Amelia Ibrahim, Natascha Ruimwijk, Deepak Sharma, Youri Sub-Laban, and Wendy van den Broek.

Jaap Kamps was supported by grants from the Netherlands Organization for Scientific Research (NWO) under project numbers 612.066.302 and 640.001.501. Maarten de Rijke was supported by grants from NWO under project numbers 017.001.190, 220-80-001, 264-70-050, 354-20-005, 612-13-001, 612.000.106, 612.000.207, 612.066.302, 612.069.-006, 640.001.501, and 640.002.501.

7. REFERENCES

[1] 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.

[2] P. Borlund and P. Ingwersen. The development of a method for the evaluation of interactive information retrieval systems. *Journal of Documentation*, 53:225–250, 1997.

[3] Y. Chiaramella. Browsing and querying: Two complementary approaches for multimedia information retrieval. In *Hypertext – Information Retrieval – Multimedia (HIM'97)*, pages 9–26. Universitätsverlag Konstanz, 1997.

[4] ILPS. The ILPS extension of the Lucene search engine, 2006. http://ilps.science.uva.nl/Resources/.

[5] Initiative for the evaluation of XML retrieval (INEX), 2006. http://inex.is.informatik.uni-duisburg.de/.

[6] J. Kamps, M. de Rijke, and B. Sigurbjörnsson. What do users think of an XML element retrieval system? In *INEX 2005 Proceedings*, 2006. To appear.

[7] B. Larsen, S. Malik, and T. Tombros. The interactive track at INEX 2005. In *INEX 2005 Preproceedings*, pages 313–327, 2005.

[8] Lucene. Open-source search software, 2006. http://lucene.apache.org/.

[9] 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, 2006.

[10] 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, 2004.

[11] B. Sigurbjörnsson, J. Kamps, and M. de Rijke. Mixture models, overlap, and structural hints in XML element retrieval. In *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 196–210. Springer, 2005.

[12] A. Tombros, B. Larsen, and S. Malik. The interactive track at INEX 2004. In *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 410–423. Springer, 2005.

[13] Wikiii, 2006. http://berk.science.uva.nl:8080/wikiii.

[14] Wikipedia, the free encyclopedia, 2006. http://wikipedia.org/.

[15] Wikipedia:article_size, 2006. http://en.wikipedia.org/wiki/Article_size.

[16] Wikipedia:searching, 2006. http://en.wikipedia.org/wiki/Wikipedia:Searching.

[17] WiQA: Question answering using Wikipedia, 2006. http://ilps.science.uva.nl/WiQA/.

APPENDIX

The test persons in the Interactive experiment were all native Dutch speakers, and the simulated work tasks were formulated in Dutch. Figures 7 and 8 show the descriptions of the simulated work tasks I and II respectively.

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 7: Task I: Simulated work task

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 8: Task II: Simulated work task

Like the tasks, the questionnaires were in Dutch. Below you can find the original Dutch version of the questions mentioned in Section 5.

Satisfaction: *In hoeverre heeft u in dit systeem een bevredigend antwoord gekregen op uw taakvragen*?

Effort: *De antwoorden op de taakvragen waren in dit systeem...* Here the answers vary from *erg makkelijk te vinden* to *erg moeilijk te vinden.*

Specific tasks: *In hoeverre is dit systeem volgens u geschikt voor specifieke taakvragen*?

General tasks: *In hoeverre is dit systeem volgens u geschikt voor algemene taakvragen*?

List of authors