# Users Want More Sophisticated Search Assistants - Results of a Task-Based Evaluation*

Udo Kruschwitz and Hala Al-Bakour
Department of Computer Science, University of Essex
Wivenhoe Park, Colchester, CO4 3SQ, United Kingdom
Tel.: +44 1206 872669, Fax: +44 1206 872788
{udo|halbak}@essex.ac.uk

**Abstract**

The Web provides a massive knowledge source. The same is true for intranets and other electronic document collections. However, much of that knowledge is encoded implicitly and cannot be applied directly without processing it into some more appropriate structures. Searching, browsing, question answering for example could all benefit from domain specific knowledge contained in the documents; and in applications such as simple search we do not actually need very "deep" knowledge structures such as ontologies but we can get a long way with a model of the domain that consists of term hierarchies. We combine domain knowledge automatically acquired by exploiting the documents' markup structure with knowledge extracted on the fly to assist a user with ad hoc search requests. Such a search system can suggest query modification options derived from the actual data and thus guide a user through the space of documents.

This paper gives a detailed account of a task-based evaluation that compares a search system which utilizes the outlined domain knowledge against a standard search system. We found that users do use the query modification suggestions proposed by the system. The main conclusion we can draw from this evaluation however is that users prefer a system that can suggest query modifications over a standard search engine which simply presents a ranked list of documents. Most interestingly, we observe this user preference despite the fact that the baseline system even performs slightly better under certain criteria.

## 1 Motivation

> *"We believe that the exploitation of layout information can lead to direct and dramatic improvement in web search results." (Henzinger et al., 2002)*

A large number of electronic document collections exist within companies, universities and other institutions, but they remain difficult to access or navigate through due to the lack of more sophisticated indexing and search tools. If we could turn these collections into structured knowledge sources, we would obtain explicit domain knowledge that could be used to search or browse the collections.

How can we do that? We exploit the documents' markup structure to construct such domain knowledge. Electronic documents typically have *some* internal structure. It could be HTML markup in which the documents are encoded; it could as well be some less obvious and more implicit structure. The exploitation of the markup structure allows us to automatically transform the document collections into usable knowledge (*domain models*), i.e. knowledge that can for example assist users who want to search the documents.

The knowledge we construct comprises a number of term hierarchies, explained in more detail in (Kruschwitz, 2003a). Part of one such term hierarchy can be seen in figure 1 (a realistic example from one of our sample collections, the BBC News Web site[1]). This graph gives an idea of how our domain model is structured. We see a tree of related terms that can be used to either assist a user in the search process, perform automatic query refinements or allow the user to browse the collection. Note that the *types* of relation in the sample tree are not formally specified. This is significantly different from formal ontologies or lexical resources such as *WordNet* - a large dictionary-like resource originally developed for English that encodes linguistic relations between lexical items (e.g. synonymy, antonymy etc.) (Fellbaum, 1998).

*This is a preprint of an article accepted for publication in JASIST - Journal of the American Society for Information Science and Technology


[1] http://news.bbc.co.uk

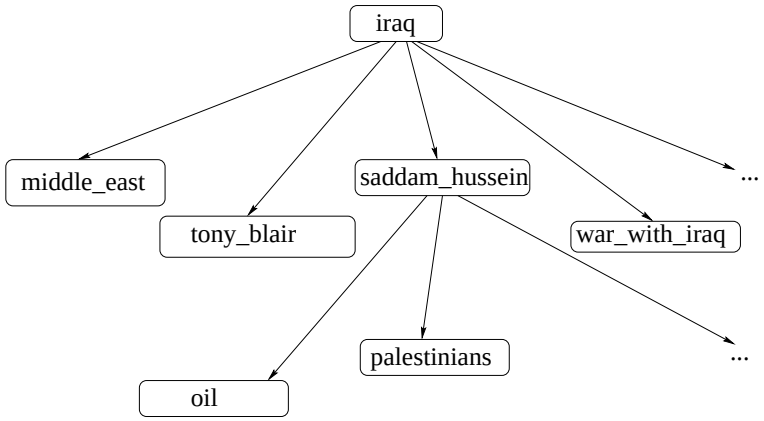


Figure 1: Sample term hierarchy

What is important is the fact that we can construct domain models rapidly for unseen document collections as long as the documents have some markup structure.

Once we have obtained such domain knowledge we can apply it in an ad hoc search system which combines a standard search engine with the added functionality that a domain model can offer, namely the construction of query modification suggestions. Figure 2 shows a screenshot of the *UKSearch* system - our search system that incorporates the automatically acquired domain model - running on the BBC News domain: a user started by searching for *"iraq"* (one of the queries most frequently submitted to the BBC News site according to the log files we are using), and is now presented with the first level of the hierarchy of terms that has *"iraq"* as its root node. The user can also choose to ignore the suggestions and enter some query modification in the appropriate input field (e.g. *"kay"*). In any case the system will map the modified query against the domain model and initiate a new dialogue step.

So far so good. Automatically extracted knowledge has its advantages, the main one is that such knowledge can be constructed rapidly. But if we just construct the model once and then apply it, we will end up with a rather static knowledge source that will be out-of-date after some time unless we re-run the construction process.

We suggest a framework in which the relations encoded in the domain model are combined with what can be extracted on the fly from the documents that match a user query, and those terms are presented alongside each other. The intuition behind this can be summarized by the following observations:

- The domain model we construct does in fact encode term relations that can assist a user in searching the document collection. User experiments reported in (Kruschwitz, 2003b) have shown that the domain model we construct contains very sensible query refinement options.
- Our domain model is a representation of the *entire* document collection, in other words the refinement options which can be derived from such a model will cover a wide range of topics (e.g. in figure 2 we can see refinement terms such as *"oil prices"* alongside *"gulf war"*). The problem is to keep the domain model of a constantly changing document collection up-to-date (something particularly relevant for Web sites such as BBC News).
- Terms extracted on the fly from the top matching documents will give a narrower, more focussed picture of the document collection. However, such terms can possibly be more *up-to-date* and better at describing the few documents which are most relevant for an average query.
- Do users want refinement suggestions in the first place? In a first task-based evaluation we studied how a search system that offers query refinement options based on automatically acquired domain knowledge (i.e. *UKSearch*) compares against a standard search engine (Kruschwitz, 2003b). We found that the majority of users preferred *UKSearch* over a standard search engine. This is a clear argument for the potential benefits such a search system can offer. Apart from that, only 2 out of 16 users did *not* use the query refinement options suggested by *UKSearch* at all. In other words, although users were free to ignore the query modification terms they did actually make use of them.

The conclusion we draw from these observations is that we handle user queries by returning the best matching documents alongside query modification options which are a combination of terms extracted from the domain model and terms extracted from the best matching documents.


Figure 2: Screenshot - BBC News domain

This framework allows us to rapidly acquire domain models for individual document collections and goes beyond our previous work, e.g. (Kruschwitz, 2003a), in that we combine these domain models with knowledge extracted on the fly.

Finally, by recording all the users' selections in the interaction with the search system we get the additional advantage of being able to adjust the domain model automatically by incorporating the users' selections in an evolving domain model.

The paper will be structured as follows. We will start with an overview of how markup structure can be exploited to extract domain knowledge that can then be organized in a domain model (section 2). We will then explain how the model is incorporated in a search system which goes beyond purely matching a user query against a set of documents (section 3). Section 4 describes a detailed task-based evaluation of such a search system. We finish the paper with some discussion of future work as well as related work and some conclusions.

# 2 Domain Model Construction

This section presents a short review of the domain model construction process. This process starts by automatically selecting a small number of terms (nouns and noun phrases) from the document collection. We then organize these terms (which will be called *concepts*) as a set of simple hierarchies which will form the domain model.

We start with some simple assumptions about partially structured documents (of which Web documents are just one particular example) to locate conceptual information, namely:

- We can typically break down documents into different markup contexts (be it that the documents are marked up explicitly or they contain some implicit structure).
- We can use this structure to extract significant terms.
- We do not want to rely on a single type of information that might be used infrequently (for example, *meta* tags in HTML).
- We do not want to assume anything about the semantic interpretation of a particular type of information.
- We only consider keywords and phrases as suitable index terms if they are either nouns or noun phrases that match sequences of part-of-speech tags typically used for detecting collocations.
- We find conceptual information in documents by selecting those index terms that are found in more than one markup context of a document.

In documents marked up in HTML we can identify a number of frequently used markup contexts such as document *titles*, *bold* text, certain text found in *meta* tags, *underlined* text etc. If the documents were articles in a newspaper archive, we might distinguish markup contexts like article *headings*, *captions* of pictures, *summaries* etc. We then define:

**Definition 1** *An index term c is a concept term (or simply concept) of type n for document d, if c occurs in at least n different markup contexts in d.*

We argue, that this abstraction separates important terms from less important ones in a very efficient and domain-independent way; see (Kruschwitz, 2003a) for supporting evidence that this is indeed the case.

We can decide what type of concepts we want to deal with (e.g. type 2 means that concepts are all those terms found in at least two markup contexts of a document). For simplicity we can assume a fixed type n (for example type 2), and talk about *concepts* as a short form for *concepts of type 2*. A more detailed discussion can be found in (Kruschwitz, 2003a).

The identification of conceptual terms is the first step towards the construction of a domain model. The next step is to arrange the extracted terms in some simple usable structures.

One of the main motivations for automatically acquiring a domain model is the inadequacy of a domain-independent knowledge source like *WordNet* in search applications for limited domains. Having said that, the *structure* of a knowledge source like *WordNet* looks very promising for ad hoc search assistance. The reasons include the clear and simple organization, its applicability to natural language engineering systems, and the advantage that the model does not have to be rebuilt every time the document collection needs to be re-indexed. That is because *WordNet* as well as the model we will introduce are independent of the actual documents.

To make the last point clearer, compare this with a very different approach in which the domain knowledge is closely linked to the documents and not separated. The MIT *START* system[2] links knowledge to the actual

[2] `http://www.ai.mit.edu/projects/infolab/`

documents (Katz et al., 2001). *START* is a question answering system that indexes a document collection by annotating documents and storing those annotations in a knowledge base. The online search system tries to match the user query against the annotations. These annotations essentially describe the questions that some part of a document is able to answer.

The domain model we construct is a set of simple concept hierarchies. The interpretation of these hierarchies is very different from a number of other models. In *WordNet* for example the links between two sets of terms are clearly defined semantic relations (e.g. *hypernymy*, *antonymy* etc.). Our aim is not to capture the actual *semantic* relations that exist between concepts but the fact *that* there is some relation, one that can be used to guide a user in a search process. Two examples of *structurally* similar models are described in (Sanderson and Croft, 1999) and (Anick and Tipirneni, 1999). The construction process is however different to the work presented here.

The construction of our model is straightforward and is performed fully automatically in an offline process. The process is based on a simple relation:

**Definition 2** *Two concepts $c_1$ and $c_2$ are related concepts of type n for document $d$, if $c_1$ and $c_2$ are concepts of type n for document d.*

Although this relation is defined for two concepts in the same document it also allows us to abstract from the single document and to consider two concepts related if there is at least one document for which this is true. This will be the assumption on which we construct the domain model as we will see shortly.

Each of the hierarchies in our model consists of nodes represented by a concept (see for example figure 1). We have found that the concepts that have been identified in the indexing process are likely to be among those terms that users submit as real queries when they search the document collection (Kruschwitz, 2003a). Based on this finding the model construction process is a sequence of simulations of user requests that does not use live user queries but the concepts that have been identified in the documents. Each concept is a potential query. Therefore, we construct a hierarchy for every single concept, in other words every concept represents the root node of one hierarchy. We start with a single concept as a possible user query (e.g. *iraq* in figure 1). Utilizing the *related concepts* detected in the source data one can then explore all possible ways of constraining this query by adding a single concept to the query in a query refinement step. The interesting terms which could be added to the query in such a step are all the concepts that were found to be *related* to the original query (concept) term. A new daughter node is created if there are documents in the collection that match the refined query. In the example, we create a daughter node *middle_east* because *middle_east* and *iraq* are related concepts and the query that is a conjunction of both these terms returns a non-empty document set.

The model construction is an iterative process that can be applied to the new queries until eventually one ends up in leaf nodes, i.e. nodes that typically represent very specific queries for which only small sets of documents can be found. In each of those iterative steps one would expand the current query by a single concept that is related to *all* query terms collected so far (figure 1 shows that there are documents in the collection that match the query *" iraq AND saddam hussein AND oil"*).

By limiting the number of branches originating in a node one gets a usable model in which each of the daughter nodes can be interpreted as query refinements of the query represented by the mother node. Weights are associated with each arc indicating the number of matches that each node represents (not displayed in the example).

The resulting hierarchies are applied in a simple dialogue system to assist a user in ad hoc search tasks.

# 3 System Overview

The main principle underlying the system design is that we construct a domain model for the entire collection in an offline process prior to making the system available to the user, but we only acquire knowledge on the fly for those queries the user actually submits. The dialogue manager selects suitable query modifications based on the relations encoded in the domain model and presents them as a flat list of terms alongside the terms extracted on the fly. Figure 3 is a simplified overview of what happens inside *UKSearch*:

- The user query is submitted simultaneously to a search engine (for example to Google via its API[3]) and the *UKSearch* backend.
- The search engine results are displayed alongside the query modification options which are derived from the domain model and the additional terms extracted from the best matching documents.

We will briefly describe how our systems constructs query modifications (i.e. relaxations and refinements). Every time the user interacts with the system these steps are performed:

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

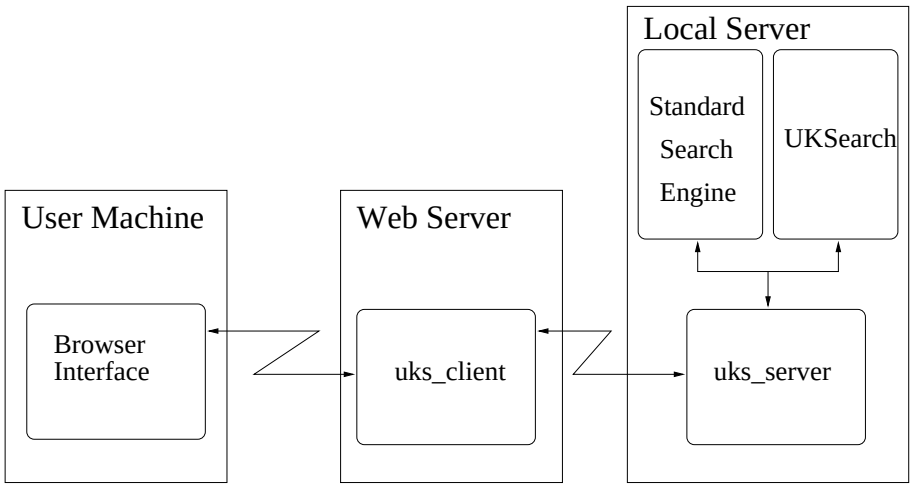


Figure 3: Sketch of information flow in *UKSearch*

- Calculate query refinements
- Calculate query relaxations
- Rank all query modifications
- Select the highest ranked query modifications and construct potential choices.

Since we have uncoupled the domain model from the actual database of documents, the dialogue manager does not actually keep track of what the search engine returns. In other words, in any case we calculate relaxation as well as refinement options.

We apply the domain model to explore a fairly restricted space of query modifications. This is because the domain model is custom-built for exactly this process, i.e. finding refinement or relaxation terms for a given query. There is no need to go deep into a hierarchy, nor are we interested in exploring sets of nodes that are not immediate neighbours in such a hierarchy. Simply speaking:

- Query refinements are calculated by proposing a single concept that could be added to the current query. Since the domain model hierarchies represent (hypothetical) query refinement steps, we just have to find a coherent path that takes us from the root node further down in a hierarchy, and on that way we collect all the current query terms (and do not skip any nodes). Following the last node on this path we have the nodes that contain query refinement terms.
- Query relaxations are constructed by either breaking the current query into parts (i.e. by deleting query terms) or by substituting the query by a single concept. This single concept has to be found in the root node of a concept hierarchy, and all query terms are found in the direct daughter nodes of that root node.

The ranking function ignores all query relaxations if there are potential query refinements. This is based on the observation that *too* general queries are much more likely than *too* specific queries, e.g. see (Kruschwitz, 2003a). Furthermore, we only extract any terms on the fly for query refinement. Otherwise we simply present query relaxations (e.g. suggestions for partial queries) derived from the domain model. An example of a query relaxation step initiated by the system can be seen in figure 4 where a user asked for *"iraq government political figures before invasion"*.

For the extraction of knowledge on the fly we currently use the titles and snippets returned by Google and process this text in the same way as we process the document collection prior to deriving the domain model. That means we assign parts of speech, select nouns and certain noun phrases. Finally we select the most frequent ones and mix them with the refinement terms suggested by the domain model. We display up to 10 terms derived from the domain model followed by the 10 most frequent ones calculated on the fly. The output looks like in figure 2, i.e. the output format does not differ from a system that uses the domain model only for the construction of query modification terms. Note that the user can also choose to replace the current query by any one of the suggested refinement terms.

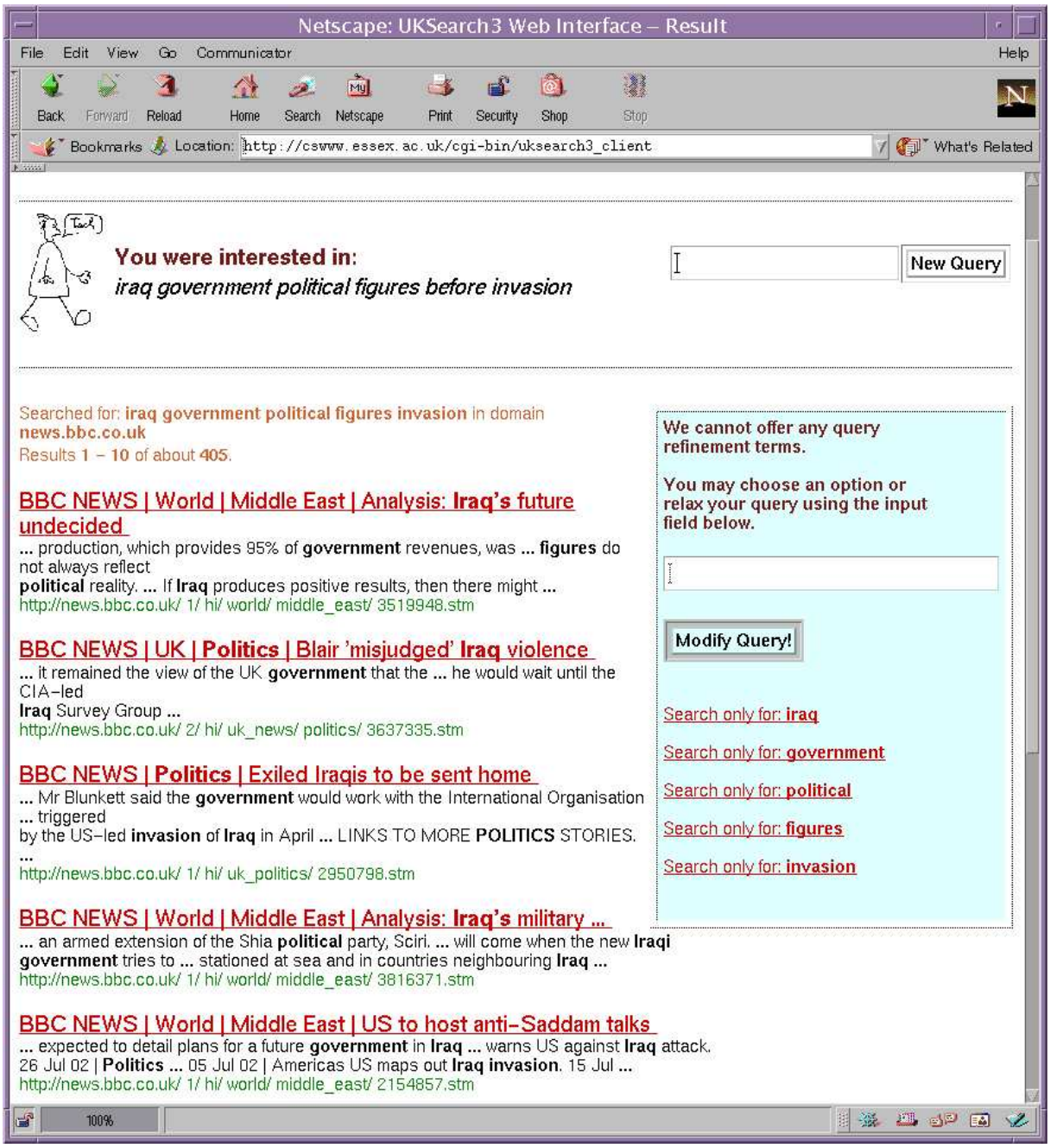


Figure 4: Screenshot - BBC News domain (relaxation)

In a typical query refinement step a user would see a list of 20 terms each of which could be added to the query. However, this is the upper limit because we perform some filtering *after* calculating the potential refinement terms. First of all, only the longest possible query terms are considered and all substrings are ignored. We want to avoid a user having to choose between *chancellor_gerhard_schröder* and *gerhard_schröder*. We discard the second one if both terms were found to be possible refinement options. This seems a sensible strategy although a query for *"dow"* in our BBC News domain will then not include a refinement term *dow_jones* which is discarded in favour of *dow_jones_milestones*. Secondly, when we append the terms extracted on the fly we discard duplicates that have already been identified in the domain model.

More details on how the dialogue manager selects query modification terms are discussed in (Kruschwitz, 2003a).

# 4 Task-Based Evaluation

Evaluating search systems is an important research issue. However simply evaluating a system for the sake of it will not be very useful if we do not follow strict guidelines that make the results comparable and allow us to draw conclusions. Therefore any evaluation will require a careful design step.

Some evaluation results have been reported in (Kruschwitz, 2003b). These evaluations focused on two questions:

- Does the domain model encode term relations which can be useful for query refinement?
- Will users prefer an integrated system (that incorporates our domain model) over a standard search engine?

For both these evaluations we found supporting evidence that the approach we follow is indeed a sensible one. However, we only looked at one domain (the University of Essex Web site) and so far we have *not* combined the domain model with terms extracted on the fly. This is what we investigated in the evaluation discussed here. The actual experimental setup of the evaluation was very similar to the earlier experiment we performed on the *University of Essex* domain (Kruschwitz, 2003b).

The TREC conference series has been successful at setting the standards for comparing systems against each other. We adopted the evaluation methods developed for the TREC interactive track (Voorhees and Harman, 2002). We compared the actual *UKSearch* system and a baseline system. Both systems access the same document collection, the BBC News Web site. The two systems can be characterized as follows:

- *System A* is the baseline system which functions like a standard search engine: the query is submitted by the user, Google (accessed via the Google API) returns the results and the first ten matches are displayed. A user can then either modify the query, go back to start a new search or click through the result set via a link that takes the user to the next 10 matches.
- *System B* is the *UKSearch* system described earlier that uses the automatically constructed domain model to assist a user in the search process by offering ways to relax or constrain the query. It also uses Google's API to display the best matching documents alongside the options that have been found.

The systems look almost identical; the difference is only that in *System A* no query modification options are constructed. Apart from that the GUI and functionality is exactly the same in both systems. Figure 2 shows how a response of *System B* looks like if the system suggests query refinements in response to a user query *"iraq"* (but note that the screenshot only presents terms derived from the domain model whereas in *System B* these terms are followed by those selected on the fly from the best matching documents).

16 subjects and 8 search tasks are needed to perform an evaluation according to the guidelines originally developed for the TREC-9 interactive track (Hersh and Over, 2001). We will first discuss the experimental setup in more detail. After that we will discuss the results.

## 4.1 Search Tasks

Apart from adopting the TREC interactive track guidelines we also wanted to address one particular limitation of the experimental setup in that track, namely the fact that for the experiments the "conditions are artificial" (Hersh, 2003). Unlike in the TREC interactive track we did not want to construct hypothetical search tasks but use the logfiles of queries submitted to the existing search engine at the BBC News Web site to make the search tasks as realistic as possible. We have been able to acquire a substantial corpus of queries submitted in December 2003 to the search engine installed at the BBC Web site. This corpus was used to construct the search task which:

1. are based on *frequently submitted queries* (four of them using frequent queries submitted to the BBC News world edition, the other four based on queries to the UK edition),

2. seem realistic (although we cannot be sure about the actual information need a user had by just looking at logged user queries, we can try to construct tasks that could have resulted in the query the task was derived from),

3. aim at single documents that contain the answers for the particular tasks (this seems realistic for searches in a news domain and furthermore makes the creation of tasks simpler),

4. are not trivial, in the sense that we tried to prevent queries that return an appropriate answer for the search task among the top 10 ranked documents (because in that case we would have a very short interaction with the system).

To summarize, we designed tasks with fairly precise targets for which we knew documents existed that satisfy the information request (possibly more than one), and where the main difficulty would be in finding such a document rather than assessing lots of very similar documents or collecting information from different documents.

Just like in our first evaluation we found the last point in our list of bullet points to be particularly difficult to satisfy due to the fact that we used Google as the backend search engine, and the results returned by Google tend to be ordered in a way that the most sensible matches for a user query can be found early on in the list of matching documents.

We also made sure that the search tasks were not tailored to fit our domain model. In other words, if we have access to the domain model when constructing the search tasks, then there could be a temptation to select certain tasks simply because we assume that the domain model is a useful aid in guiding the user in the search. Therefore we first constructed the tasks and then indexed the document collection and built the domain model.

The following tasks were constructed for this evaluation (the query that the task is based on is shown in brackets and is not part of the actual search task).

- **Search Task 1** (*brazil*): You are asked to find information about the current president of Brazil. In particular, locate a document that has detailed information about what he did in the 1980's and 1990's before becoming the president of Brazil in 2002. Documents which give a general profile of the country are not relevant.

- **Search Task 2** (*iraq*): Locate a document that contains short summaries of the main political figures in Iraq *before* the last Iraq war started.

- **Search Task 3** (*aids*): The end of last year saw the start of a new big campaign to fight Aids worldwide. Find a document that gives a summary of initiatives from around the world. Documents about activities in individual countries only and documents prior to 2003 are not relevant.

- **Search Task 4** (*flu*): Some scientists say that a new global outbreak of flu is inevitable. Find a document that has details of how Britain was affected by a recent outbreak of flu and how the country coped with that.

- **Search Task 5** (*lotto* and *lottery*): Find a document that has recent, detailed examples of how the money that the UK government raised by selling Lotto tickets was spent. Information which is more than one year old is not relevant.

- **Search Task 6** (*m6 toll*): There was a lot of discussion about the first privately financed motorway in Britain that opened recently. Find a document that has information about how some of the money that users of this motorway have to pay will be used to support other projects.

- **Search Task 7** (*travel*): Imagine you want to travel abroad and you are not sure what exactly you are (or are not) allowed to take with you in your hand luggage when boarding a plane. Locate a document that has details about who you can contact to get up-to-date information.

- **Search Task 8** (*euro*): Find a document that has details about the development of the Euro currency since its introduction in 1999. Documents are only relevant if they have milestones of the Euro's development covering the entire period from 1999 till at least the end of 2003.

## 4.2 Experimental Setup

The questionnaires used in this study are based on the ones proposed by the TREC-9 interactive track (using a 5-point Likert scale where appropriate)[4]. We used the following questionnaires:

- Entry questionnaire
- Post-search questionnaire
- Post-system questionnaire
- Exit questionnaire.

Some of the questionnaires were customized to reflect the particular experimental setup. For example, in the post-search questionnaire for *System B* the user was asked whether the query modification options presented by the system were sensible.

The assignment of subjects to tasks was based on the searcher-by-question matrix displayed in table 1. This table contains the mapping of tasks to searchers as proposed in (Hersh and Over, 2001). Note that in the table *System A* is the baseline system and *System B* is the actual *UKSearch* system.

| **Searcher** | **System: Questions** | **System: Questions** |
|---|---|---|
| 1 | B: 4-7-5-8 | A: 1-3-2-6 |
| 2 | A: 3-5-7-1 | B: 8-4-6-2 |
| 3 | A: 1-3-4-6 | B: 2-8-7-5 |
| 4 | A: 5-2-6-3 | B: 4-7-1-8 |
| 5 | B: 7-6-2-4 | A: 3-5-8-1 |
| 6 | B: 8-4-3-2 | A: 6-1-5-7 |
| 7 | A: 6-1-8-7 | B: 5-2-4-3 |
| 8 | B: 2-8-1-5 | A: 7-6-3-4 |
| 9 | A: 4-7-5-8 | B: 1-3-2-6 |
| 10 | B: 3-5-7-1 | A: 8-4-6-2 |
| 11 | B: 1-3-4-6 | A: 2-8-7-5 |
| 12 | B: 5-2-6-3 | A: 4-7-1-8 |
| 13 | A: 7-6-2-4 | B: 3-5-8-1 |
| 14 | A: 8-4-3-2 | B: 6-1-5-7 |
| 15 | B: 6-1-8-7 | A: 5-2-4-3 |
| 16 | A: 2-8-1-5 | B: 7-6-3-4 |

Table 1: Searcher-by-question matrix

## 4.3 Procedure

The procedure that every subject had to go through has been adopted from (Craswell et al., 2003):

- Subjects started by filling in the entry questionnaire.
- This was followed by a demonstration of the two systems. The users were informed that they will be using two different search engines, but they were not told anything about the technology behind them, nor about the use of Google as the backend search engine. Subjects were free to ask any questions.
- Users were then asked to perform four search tasks on one system followed by four tasks on the other one (according to the matrix in table 1). Users were asked to use the (online) study worksheet for every search task to submit the result and their confidence level.
- After each task users were asked to fill in the post-search questionnaire.
- After completing all four search tasks on one system users were asked to fill in the post-system questionnaire.

[4] `http://www-nlpir.nist.gov/projects/t9i/qforms.html`

- Finally, after finishing all search tasks users had to fill in the exit questionnaire.

Subjects had 10 minutes for each task. After that time they were informed that the 10 minutes had passed. Why did we restrict the search to 10 minutes? There are two main reasons. First of all, it has been found that in the TREC interactive track *"there was a strong desire to reduce the time per search (previously: 20 minutes)"* (Hersh and Over, 2001), so that the guidelines now just state that users should be given at least 10 minutes on each task (Hersh, 2003). The second reason is that our results would be comparable with the evaluation results reported in (Kruschwitz, 2003b) where we used an identical evaluation procedure.

## 4.4 Results

This section gives an overview of the most interesting results. In all cases, t-tests have been used for significance testing. A discussion of the results can be found in section 4.5.

### 4.4.1 Subjects

The aim was to find a set of volunteers who could be potential users of a search engine such as *UKSearch*. In order to get a good selection of different types of users and to avoid any bias in the selection process we sent an email to the local University mailing list and selected the first 16 volunteers who replied. In the email we specified that we required the subjects to be native speakers. This is different to our first task-based evaluation.

Out of the 16 volunteers 6 were male and 10 female. Their ages ranged overall from 20 to 46 (average age 27.7). This time we had a variety of backgrounds, which included 11 students from different departments (including Literature, Electronics, American Studies, and Computer Science) as well as members of staff (clerical as well as academic). None of the subjects had taken part in previous studies in online search. The average time subjects have been doing online searching is 5.6 years (12 of them between 5 and 10 years, but there was also a user who stated 0 years). When asked for their searching behaviour, the average value was 4.75, where 5 means daily, 4 means weekly. No one selected any other value (this observation as well as the average value are identical to our first evaluation experiment). An interesting observation is that only one subject strongly agreed (i.e. value of 5) with the statement "*I enjoy carrying out information searches.*", one selected 2, four subjects selected 3, everybody else selected 4 (i.e. average of 3.69).

Some other interesting statistics are summarized in table 2 (based on a 5-point Likert scale, where 1 means *"none"* and 5 means "*a great deal*"). Particularly interesting is that all users have "a great deal" of experience using a point-and-click interface and searching the Web whereas hardly anyone has experience searching on commercial online systems.

| **Experience** | **Mean** |
|---|---|
| Using a point-and-click interface | 5.00 |
| Searching library catalogues | 3.27 |
| Searching CD ROM systems | 2.81 |
| Searching commercial online systems | 1.31 |
| Searching the Web | 5.00 |

Table 2: Subject experience with computers and search systems

### 4.4.2 Search Statistics

The average length of the initial user query was 2.89 words (*System A*: 2.82, *System B*: 2.95), i.e. longer than queries typically submitted to Web search engines (2.35 words in average according to (Silverstein et al., 1998)). There was one query of length 8 and one of length 7, all others were shorter than that. In our first evaluation we observed shorter queries, an indication that the tasks we set this time were more specific and perhaps more difficult.

Table 3 gives a picture of the average completion time broken down for each task. We decided to measure the time between presenting the search task to the users and the submission of the result. If the search was not completed after 10 minutes, we asked the user to start with the next task. The users could then either submit a document (and possibly indicate a low confidence) or not submit anything. There were 8 cases in which users did not submit any answer (5 on *System A* and 3 on *System B*). In those cases we added a 60 second penalty to

the 10 minutes as suggested by (Osdin et al., 2003) in a similar evaluation task. In table 3 the t-test gives us one significant difference (task 1: $p < 0.05$).

| System | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| A | 230.0 | 501.1 | 416.5 | 289.0 | 291.4 | 424.8 | 319.4 | 435.8 |
| B | 331.1 | 490.1 | 521.6 | 316.9 | 291.5 | 511.4 | 222.5 | 381.8 |

Table 3: Average completion time (in seconds)

Overall, the average time spent on a search task on *System A* was 363.5 seconds, on *System B* 383.4 seconds. There is no significant difference. Note that the average time spent on a search task is much higher than in our first experiment (around 285 seconds on average) which again suggests that the tasks were more difficult.

There are a number of other aspects that can be derived from the logged data. One particularly interesting fact is that users had to inspect fewer documents on *System B* to finish a task (on average 6.9 which compares to 8.0 on *System A*). The breakdown of the average number of documents inspected for each task is included in tables 5 and 6 . No significant differences could be found however.

A second notable aspect is the fact that with guidance by the system the user is able to complete the task in fewer steps, although the difference is marginal and the average values per task vary a lot. In table 4 we break down the average completion of a task into a number of "turns", i.e. the steps it takes to find the answer for a search task. A turn can be inputting a query (or modifying the query), selecting a modification option, following the link to the next ten matches or following a hyperlink to open a document. On average users needed 12.7 turns on *System B* (compared to 13.2 on *System A*). Tables 5 and 6 present a more detailed picture.

| System | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| A | 7.9 | 15.1 | 13.0 | 10.2 | 9.1 | 17.4 | 12.1 | 20.5 |
| B | 10.9 | 17.9 | 20.1 | 9.6 | 9.1 | 12.1 | 7.4 | 14.6 |

Table 4: Average number of turns to complete a task

| Type of Turn | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| Reformulations (incl. "next 10") | 2.8 | 6.0 | 4.8 | 3.9 | 4.9 | 7.4 | 5.8 | 5.6 |
| Documents viewed | 5.1 | 9.1 | 8.2 | 6.4 | 4.2 | 10.0 | 6.4 | 14.9 |

Table 5: Average number of turns to complete a task on *System A*

As to the query modification suggestions presented by the system, users did indeed make use of them although not excessively (this reflects one of the objectives of our search framework which is to allow users to ignore the suggestions and instead use the system as a standard search engine if they want to). On average users selected 1.2 query modifications in each task performed on *System B*. These modifications include query relaxations, refinements or replacing the current query by some suggested term. The breakdown indicates that in total the modification suggestions based on the automatically constructed domain model and the dialogue manager were selected nearly three times as often as terms which were selected from the best matching documents on the fly (the "baseline terms") although there is a big variation across the tasks. More discussion of this particular aspect of the system will follow in section 4.5.

| Type of Turn | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| Model suggestions selected | 1.6 | 0.6 | 1.5 | 1.2 | 1.1 | 0.4 | 0.4 | 0.4 |
| Baseline terms selected | 0.0 | 0.5 | 0.9 | 0.0 | 0.5 | 0.1 | 0.1 | 0.4 |
| Reformulations (incl. "next 10") | 4.6 | 6.6 | 6.4 | 2.8 | 4.1 | 4.6 | 2.4 | 5.1 |
| Documents viewed | 4.6 | 10.1 | 11.4 | 5.6 | 3.4 | 7.0 | 4.5 | 8.8 |

Table 6: Average number of turns to complete a task on *System B*

In the post-search questionnaire users were asked to state whether they were able to successfully complete their search task. For *System A* 9 answered with *No*, for *System B* there were 5 cases altogether. However, we also went through every submitted document identifier and judged (based on the information contained in the sample documents we identified at task construction time and on the actual search task) whether we would consider it a match for the task. We found that there was a large number of submitted documents that did not *exactly* match the information request as specified by the task. That includes partial matches or documents that did not match the outlined constraints. Only 83 of the 128 search tasks resulted in exact matches (43 on *System A* and 40 on *System B*). That is a clear indication that the tasks were very difficult. There was no significant difference between the two systems in that respect, but there were two particularly difficult tasks: tasks 2 and 6 (clearly reflected by the user satisfaction values in table 7 further down). Only 3 of the 16 users found a correct document for task 6, and 5 were correctly submitted for task 2. Typical comments in the post-search questionnaire for task 6 were *"Found information about the new Motorway but not how the funds are used"* and *"found relevant information but no answer to specific question of local project benefit"*.

If we look at all 45 unsuccessful tasks in detail and compare it against the results reported earlier on, we find that in average users looked at more documents (*System A*: 10.6, *System B*: 9.1) and needed more turns (*System A*: 17.6, *System B*: 15.3). Interestingly, despite longer interactions with the system users selected fewer modification options presented by *System B* (1.0 in average per task; domain model suggestions were selected exactly twice as often as terms extracted on the fly).

#### 4.4.3 Post-Search Questionnaires

After finishing each search task a post-search questionnaire had to be filled in. The questions that were common for both systems were the following (using a 5-point Likert scale, where 1 means *"not at all"* and 5 means *"extremely"*):

- *"Are you familiar with the search topic?"*
- *"Was it easy to get started on this search?"*
- *"Was it easy to do the search on this topic?"*
- *"Are you satisfied with your search results?"*
- *"Did you have enough time to do an effective search?"*
- *"Did your previous knowledge help you with your search?"*
- *"Have you learned anything new about the topic during your search?"*

Table 7 gives a breakdown of the results for the question *"Are you satisfied with your search results?"* Overall users were marginally more satisfied with the results returned by *System A* than with *System B*, but no statistical significance can be observed for the overall result or in fact any of the data reported in table 7.

Table 8 presents the results for the question *"Have you learned anything new about the topic during your search?"* Overall users indicated that they have learned more when using *System B* than using *System A*. Significant is that users learned more when performing task 4 on *System B* than on the baseline system ($p <= 0.01$).

Table 9 gives figures for the other questions of the post-search questionnaires. None of the differences are significant. It is interesting to point out that every single value in table 9 is smaller than the corresponding value calculated in our earlier evaluation (Kruschwitz, 2003b). This is the strongest evidence that the tasks we asked

| System | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| A | 4.38 | 2.88 | 3.12 | 3.38 | 3.25 | 2.62 | 3.38 | 3.38 |
| B | 4.00 | 2.38 | 2.88 | 3.25 | 3.25 | 2.38 | 4.38 | 2.75 |

Table 7: Post-search questionnaire (user satisfaction for each task)

| System | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| A | 3.25 | 2.25 | 2.75 | 2.00 | 2.75 | 2.50 | 2.12 | 2.50 |
| B | 3.00 | 2.25 | 3.00 | 3.38 | 2.75 | 3.12 | 2.75 | 2.38 |

Table 8: Post-search questionnaire (something learned)

| System | Familiarity | Start | Search | Satisfied | Time | Knowledge | Learned |
|---|---|---|---|---|---|---|---|
| A | 2.47 | 3.41 | 3.22 | 3.30 | 3.77 | 2.27 | 2.52 |
| B | 2.27 | 3.23 | 3.08 | 3.16 | 3.43 | 2.30 | 2.83 |

Table 9: Post-search questionnaire

users to perform in this evaluation were difficult indeed. That sentiment is also reflected by a number of comments the users provided in the questionnaires, e.g. *"It was hard to search for something that I don't know very much about"*, *"Found it hard to find the specific page.*", and *"I think most of the documents requested are specific, so it's not that possible to just search and obtain them.*"

Now it could be desirable to quantify the complexity or difficulty of a task, but constructing tasks of different complexity is an art on its own. We do however know that task complexity can have a significant impact on issues such as search success and user satisfaction (Bell and Ruthven, 2004). We will not try to establish different levels of complexity in our tasks now that the evaluation has been performed. However, in table 10 we give a task-by-task breakdown of some of the properties displayed in table 9. Here we do not distinguish between the two systems, because we want to get a picture of what the users' perceptions were about the difficulty of the tasks in general. There is a clear pattern which shows that the higher the value for user satisfaction the easier users found it to get started *and* to do the search. The only task that does not strictly follow this pattern is the task with the lowest average user satisfaction value (i.e. task 6). But note also that there seems to be no obvious correlation between familiarity with a topic and the "difficulty" of a task. For example, task 1 is the one users were least familiar with but it is the task they judged to be easiest to get started with and do the search. One would perhaps expect a close correspondence between these features (i.e. easier to search if more familiar) as for example reported in another experiment performed as part of the TREC interactive track: *"It appears that an inherent feature of a difficult topic is the level of familiarity and the understanding of the content and context of the issues.*" (Beaulieu et al., 1999).

The post-search questionnaire for *System B* contained two additional questions, that were not relevant for the basic search engine. One question was *"Did you know at each point in the interaction with the system what options you had to continue the search task?"* (a type of question adopted from the *PARADISE* framework for the evaluation of spoken dialogue systems (Walker et al., 2000)). The mean value for that question was 3.77. The other question was *"Were the query modification options presented by the system sensible?"*. Here we had a mean value of 3.16. Both these results indicate that the search system we propose (i.e. *UKSearch*) is easy to use and typically presents sensible query modification options. Having said that we also observed that although users typically find

| Criterion | Task 1 | Task 2 | Task 3 | Task 4 | Task 5 | Task 6 | Task 7 | Task 8 |
|---|---|---|---|---|---|---|---|---|
| Familiarity | 1.38 | 2.50 | 2.50 | 2.44 | 2.56 | 1.62 | 3.19 | 2.75 |
| Start | 3.88 | 2.62 | 2.94 | 3.62 | 3.50 | 2.88 | 3.88 | 3.25 |
| Search | 3.88 | 2.38 | 2.62 | 3.44 | 3.44 | 2.88 | 3.69 | 2.88 |
| Satisfied | 4.19 | 2.62 | 3.00 | 3.31 | 3.25 | 2.50 | 3.88 | 3.06 |

Table 10: Post-search questions (task-by-task)

| Question | System A | System B |
|---|---|---|
| How easy was it to *learn to use* this information system? | 4.31 | 4.19 |
| How easy was it to *use* this information system? | 3.53 | 4.38 |
| How well did you *understand how to use* the information system? | 4.19 | 4.12 |

Table 11: Post-system questionnaire

the modification options sensible, they have not been used heavily. A clue could be that the tasks were rather specific (e.g. sample user comments: *"Found a great deal of general information but difficult to find exactly what the task required, even with the search options"*, and *"Some of the links given for the second system weren't that relevant to the current search"*). Another explanation is that the suggestions are useful not just for choosing the suggested query modifications but to get a feel for the document collection, e.g. one user commented *"With the extra option list offered by the system, I found it far easier to use the suggestions offered by the system or using the ideas and word combinations it gave me to modify my search."*

#### 4.4.4 Post-System Questionnaires

After performing four search tasks on one system a post-system questionnaire had to be filled in. Here we only present the statistics for the multiple choice questions. Later we will discuss any additional comments made by the subjects in more detail. Table 11 gives a breakdown of the results. The only significant result is that users found *System B* easier to use than *System A* ($p < 0.006$). This is an important result since we consider the baseline system (*System A*) to be very simple and easy to use like most standard Web search engines.

#### 4.4.5 Exit Questionnaire

In the exit questionnaire users were asked to answer the question *"Which of the two systems did you like the best overall?"*. Users strongly prefered *System B*. Despite the fact that users were slightly more satisfied by the results returned by the baseline system, their overall perception was in preference of *System B*. 13 users prefered *System B*, 1 prefered *System A* and 2 found no difference. Although our first evaluation indicated the same preference we did get a much more significant difference this time. Remember, that the underlying search engine in both systems was Google, which means that *System B* essentially competed with a version of Google. Note however that the users were not told anything about the underlying search engine.

We should also stress that such a strong preference for the system that we propose cannot just be taken for granted. In a recent study which also compared a baseline search system with a system that incorporates automatically constructed concept hierarchies it was found that more than half the subjects prefered the baseline system *"'because of its simplicity and familiarity"* (Joho et al., 2004). Perhaps the fact that we do not present the full hierarchies but simply some query modification terms alongside the search results makes our system more familiar and hence more preferable by the users.

Furthermore, a large majority of users also judged that *System B* was easier to use than the baseline system. Finally, when looking at the question of which system was easier to learn to use it should be remembered that there

| Criterion | System A | System B | No difference |
|---|---|---|---|
| Easier to learn to use | 5 | 4 | 7 |
| Easier to use | 2 | 10 | 4 |
| Best overall | 1 | 13 | 2 |

Table 12: Exit questionnaire (system preference)

| Question | Mean |
|---|---|
| To what extent did you understand the nature of the searching task? | 4.25 |
| To what extent did you find this task similar to other searching tasks that you typically perform? | 3.56 |
| How different did you find the systems from one another? | 3.06 |

Table 13: Exit questionnaire (search experience)

is a set of different options which *System B* could come back with. Apart from presenting query refinements it may instead present query relaxations. Getting used to these options involves a short learning process that the baseline system does not require. Therefore it might be surprising that not more people voted for *System A* to be simpler to learn to use.

Table 12 summarizes these results. Displayed are the numbers of users who selected each of the choices. The figures in this table confirm the results of the post-system questionnaires in that users found *System B* much easier to use whereas *System A* was a tiny bit easier to learn to use.

Table 13 summarizes the answers users gave in the exit questionnaire concerning the search experience they had in the experiment (where 1 means *"not at all"* and 5 means *"completely"*).

The main conclusion that we derive from the statistical evidence is that in the given context of searching for documents in a large collection of news articles users strongly prefer a search system that offers more than just a ranked list of documents. Our users favour a system that offers query refinement and relaxation options and guides them through the available information. We can further conclude that users consider such a system to be significantly easier to use. The second interesting conclusion to be drawn is that users favour such a system although there was no significant difference between the two systems according to a number of different measures.

We can also conclude that the presented query modification options were generally considered sensible although in the given setup they did not result in measurable benefits. We hypothesize that the major reason for that is that the presented modification terms were not always helpful for the extremely specific tasks that had to be answered in this context.

## 4.5 Discussion

So far we have mainly been concerned with the statistical evidence. In this section we will discuss some of the other issues, such as patterns in the users' search behaviour as well as feedback that came from the subjects.

### 4.5.1 Patterns in User Behaviour

First of all we wanted to know whether users make use of any suggestions presented by *System B*, be it for refinement, relaxation or replacement. We found that only a single user did not make use of any such options. Remember that users were free to ignore any options and could use the system just like a standard search engine. The fact that users actually utilize the suggestions proposed by the system confirm the observations in our first experiments (Kruschwitz, 2003b).

Now if we look at it in more detail we find:

- 14 users selected *refinement* terms presented by the system. Those are terms which are added to the current query. Of those 14 users there were 9 who selected terms which have been derived from the domain model,

i.e. *concepts* that have been extracted in the model construction process. As outlined earlier on we present such concepts alongside terms extracted on the fly from the best matching documents.

- 13 users selected *relaxation* options suggested by the system. Those options are presented to break down the query into individual parts for example.

- 8 users selected *replacement* options, i.e. they replaced their query by one of the refinement terms suggested by the system.

As outlined earlier, in those cases where our search system presents refinement suggestions we display a list of terms that could be added to the query, and the first in the list will be those derived from the domain model (followed by terms extracted on the fly). The intuition is based on our experiments that found terms encoded in the domain model to be more suitable as query refinement options than those extracted on the fly (Kruschwitz, 2003b). We pointed out that the domain model concepts were selected much more frequently than the terms extracted on the fly, but we do not know to what extent this might have been affected by the order of term presentation.

We also observed that the tasks seemed to be particularly difficult if we look at the figures for measures such as time taken and user satisfaction and compare them with our earlier evaluation. Obviously, we deliberately avoided simple tasks because we did not want users to type in a query and get the results back straightaway. However, in a future evaluation we would need to mix simple with more difficult tasks, because we assume that simple tasks are the majority of realistic user requests, e.g. a lot of users who submit the query *"iraq"* will be interested in what is going on in Iraq right now, something that will typically be listed on the first result page.

Although the information requests we constructed were very specific, we found that with one exception all the original queries that the tasks were based on in the first place were actually submitted by at least one user (partly as the initial query, partly to replace some original query by a new one). The exception is *"travel"*, in which case we only found similar queries such as *"travel guide"* and *"travel information"*.

In our first evaluation experiments we noticed a large number of typos in the user queries. And again, nearly 20% of all search tasks in this evaluation (24 out of 128) contained queries with typographical errors, e.g. *"govenrment"*, *"wordwide"*, *"AIDS compaign"*, *"britain coped flue"*, *"sadam"*, and *"euro currancy"*. This is a significant number and indicates again the type of problems to be addressed by a search engine.

We shall now look at some of the individual tasks since a task-by-task analysis can lead to interesting differences in the interaction style (Beaulieu et al., 1999). We selected three tasks which according to table 10 represent the most difficult task we asked users to perform (task 2), the easiest one (task 1) and one in the middle (task 5). We are mainly interested in the interaction with *System B*.

- **Search Task 1** (*brazil*): This was the task where the average user satisfaction value was the highest and where users submitted the shortest initial queries in average (2.44 words). Only one submission was incorrect. If we only consider *System B*, we find that five users made use of query modification suggestions proposed by the system (all but one of them more than once). Interestingly, the query refinement terms that were selected came all from the domain model. The log files suggest that a typical query such as *"president brazil"* or *"President of Brazil"* was handled nicely by the system; two users selected a refinement term immediately after submitting a query (*"lula da silva"* and *"luiz inacio lula"*, respectively), both of which were followed by more selections of system suggestions later on. One user first modified the query using the input field before selecting *"lula da silva"* as a refinement term. The other two users who opted for automatically generated query modifications first rephrased their queries before choosing a relaxation option (*"Search only for: brazil and president"*, or *"Search only for: president"*, respectively). These last two cases suggest that if the query contained additional words which were not filtered as stopwords (e.g. *"Current president of brazil"*) or contained typos (*"President of Brasil"*), then no refinement option would be displayed and users needed to rephrase their query or select relaxations.

  This is also the task where a number of users typed in dates (e.g. *"1980-1990"*) which were ignored by the system which led to confusion on the users' end.

  In general users who performed this search task on *System B* considered the query modification suggestions sensible (mean value 3.75 on the 5-point Likert scale, one user selected a 1, i.e. *"not at all sensible"*, all others selected 3,4 or 5).

- **Search Task 2** (*iraq*): This was the task with the longest average initial query length (3.88 words), and at the same time one of the two tasks where the success rate was really low (only 5 correct submissions) as well as the user satisfaction value. Typically, users submitted fairly specific queries but could not find any matching documents, so that they then rephrased the query (e.g. sample queries are *"main political figures in Iraq"* and *"political figures in IRAQ pre-war"*). Five users of *System B* selected query modification suggestions,

one of them only once, the others twice. Interestingly, all three correctly submitted results on *System B* were submitted by one of those five users. However, it appears that the suggestions returned by the system were not always helpful. Only in two cases did the users select a system suggestion following the initial query, in both cases they selected a relaxation which happened to be the same one (*"Search only for: iraq"*).

We noticed that it was difficult to find documents that deal with information for the specified time. Most of the documents seem to be about current affairs. We also assume that one of the reasons for not being able to locate a suitable document easily is because words such as *"before"* and *"pre"* would be filtered as stopwords.

Asked whether the query modification options generated by *System B* were sensible we get a very homogeneous result: average value 3.00 with one users scoring a 2, one user scoring 4, all others selected 3.

- **Search Task 5** (*lotto* and *lottery* ): The average length of an initial user query was 2.62 words. Only one submission was not considered correct. Four users selected query modifications suggested by *System B*, the other four did not select any of those options. Interestingly, all those users who did choose such an option did that more than just once (up to five options). Some users of *System B* submitted queries which did not require them to do any resubmissions or modifications, but a simple inspection of the returned documents was sufficient to perform the task (e.g. *"Lotto tickets UK government"* and *"lottery funding"*). Three of the four users who selected query modifications did in fact choose a query relaxation in the first step (e.g. a user submitted *"Lottery projects 2003"* as the initial query and then selected the search option *"Search only for: lottery"*; similarly the query *"Lotto money spent"* was broken down into relaxation options and the user selected *"Search only for: lotto"*. In all three cases the user later added a term suggested by the system in a refinement step (e.g. *"lottery cash"*).

  For this task the user opinion as to whether the query modification suggestions are sensible varied a lot (average value 3.00, but the values ranged from 1 to 5).

The picture we get by analyzing the tasks is that in any one task about half of the users apply the suggestions proposed by the the system. Furthermore, if users do select query suggestions, then they typically do it not just once for each task. It also appears that in some tasks users were more likely to relax their initial query whereas in other tasks they refined it. We do not want to over-interpret these results and instead focus on the comments that users had after using the systems.

#### 4.5.2 User Feedback

Before we do so we should add that one of the criticisms in the first task-based evaluation was that users were confused by the input field in the old system which was wrongly assumed to start a new query, but in fact it was there to modify the query. We now provide two separate input fields (one for modifying the query and one for a new query). This has been appreciated because the common feedback was that both systems were easy to use and very intuitive. In fact, a frequent comment made by the users in the exit questionnaire was that both systems were simple to use. The simplicity of the layout was noted and the fact that there were no distractions on the screen.

Users generally liked the idea of refinement options. Several users expressed that such options can give you an idea of what information is available and that the system can suggest *"similar topics that weren't originally thought of"*. Other users commented that the offered terms could either be selected or help in getting an idea of how to modify the query. One user said that *System B "helps with the spelling of keywords and can give the user more specific knowledge about the topic they are searching for."* Other comments concerning the strong points of *System B* were *"System B had refinements which really helped"*, *"This system provided more search alternatives, that yielded a wider number of relevant search results"*, and *"No clutter - just the search box and results."*

A criticism for *System B* was that the query modification options were not always helpful and that they could actually divert the search to completely different topics. One user noted: *"I liked the idea of refinement options, just didn't find them useful."* Another comment was: *"The modified query options it gave didn't always help the search - sometimes you ended up going round in circles if you weren't familiar with the search topic already. It is better than with no options however like the first system as long as you don't rely on them."*

One problem we identified was that we did not index on dates and such information was simply ignored, which meant that users were not happy if they asked for *"2003"* and the input was simply deleted. That problem can easily be fixed. Another aspect specific to the domain was that users would have found it more helpful if there had been a date associated with the documents or the documents had been sorted by date.

As a confirmation of what we observed earlier a user commented *"The topics were very specific and it was quite difficult to find an exact match for any of the questions."*

So what did the three users say who prefered the baseline system or had no preference at all? The user who prefered the baseline system criticised that the options which are suggested by *System B "are not complete,*

*therefore can tempt the user to use them, diverting the search"*. Of the two users with no preference for either of the systems one noted that *System B* was in fact more useful in breaking down the query with refinements, but that *"sometimes the refinement options were not useful"*. The other user commented that both systems were not really different to what he or she is used to and *"it is just as easy to type in your own refinements. However, when searching a subject that you are not familiar with it may help to have refinement suggestions.*". We conclude that the main criticism does not so much affect the underlying principle of a search assistant that guides a user through the space of documents but rather the quality of the knowledge sources we incorporate. Therefore we conclude that (at least for the type of search scenario investigated here) users do prefer search systems which assist a user in query modification steps over a standard search engine.

# 5 Next Steps

A particular focus of our future work is to maintain the domain model once it has been constructed. As with any statistical approach the automatically constructed model can only be as good as the data it is derived from, and it will only reflect the information contained in the documents which was available at construction time. Naturally, there are flaws. As indicated by the users' comments some of the relations encoded in the domain model could even be misleading. If such knowledge is to be used in search applications, we can see two obvious ways of addressing the shortcomings: to manually adjust the domain model, which is very expensive, or to apply user feedback which has been shown to be not very effective since most users simply ignore it (e.g. (Ruthven et al., 2001)).

In (Fasli and Kruschwitz, 2001) we introduced some ideas of how the model can be maintained and adjusted automatically. The idea is straightforward. Any selection performed by a user in the interaction with the system is interpreted as positive relevance feedback.

The domain model (e.g. see the partial term hierarchy in figure 1) is used to suggest query refinement terms if the query matches a large number of documents. In this case the term hierarchies that make up the domain model will be applied so that all potential refinement options are nodes that sit below some other node. The terms that populate these nodes (i.e. our *concepts*) are presented to the user, e.g. the query *"iraq"* triggers the system to suggest *"middle east"*, *"tony blair"* etc. as query refinement terms as depicted in figure 2.

Now assume, for every concept in the domain model the weights associated with the branches originating in a node are equal and sum up to 1. These weights change, if:

- a concept in a daughter node is offered and selected by the user (increase)
- a concept in a daughter node is offered and not selected (decrease).

This will only change weights of relations already in place. The result is that the good parts of the domain model will gain importance, the rest will be less and less relevant. But that does not allow the creation of new links. However, because we combine the domain model which has been constructed at an earlier stage with terms extracted on the fly, we will be able to introduce new links in the domain model if the user selects one of the terms not part of the domain model. These new links will gain importance if this sequence of user inputs is observed a number of times. We also provide an input field alongside the query modification options. If a user ignores all the options and inputs something in this field (e.g. *"kay"* in the example pictured in figure 2), then a new link is created in the same way.

This whole aspect is the focus of a new research proposal, because we will need to perform large scale experiments to validate the usefulness of this idea on various document collections.

# 6 Related Work

Web search as such is obviously closely related to the overall problem we are addressing. We will focus on some aspects that are important for the actual interaction between user and system. We shall first look at some motivating investigations concerning the users' behaviour when searching the Web. The most comprehensive study of Web queries so far is (Silverstein et al., 1998). An evaluation of a large log file of nearly a billion queries submitted to the *AltaVista* search engine in a period of 43 days comes to a number of interesting conclusions. First of all, queries are normally very short. The average length of a user query is 2.35 words (similar results are reported in an earlier but smaller evaluation of queries submitted to Excite (Jansen et al., 1998)). That means queries on the Web are shorter than in more traditional information retrieval systems. Secondly, the 25 most common queries account for 1.5% of all queries, even though they are only a small fraction of all unique queries. As the third interesting result it was found that *"surprisingly, for 85% of the queries only the first result screen is viewed, and 77% of the sessions only contain 1 query, i.e. the queries were not modified in these sessions"*.

There are at least two lessons to be learnt from this work. First of all, user queries are generally very short which will naturally lead to a large number of documents being returned. But the second aspect is that the majority of users did not perform any query modifications. A system which applies a model of the domain to propose possible query refinements must perform extremely well to be accepted by the user.

Furthermore, a number of studies have been conducted to find out whether the search process could benefit from offering potentially relevant terms to the user in an interactive query expansion process. One of these studies is (Margennis and van Rijsbergen, 1997) where potential expansion terms are automatically derived from the documents retrieved by the original query. Their underlying assumption reads as follows:

> *"It seems reasonable to assume that a searcher, given a list of the query expansion terms, will be able to distinguish the good terms from the bad terms."*

They found that interactive query expansion performed by an experienced user has a potential to significantly improve the search process. However, they have also found that inexperienced users did not make good term selections and hence interactive query expansion led to no improvement in the search process. They conclude: *"Without good strategies and careful reasoning it is unlikely that a searcher will be able to use techniques such as interactive query expansion effectively"*.

As indicated earlier on we already found that users do make use of query modification terms when we performed our first task-based evaluation (Kruschwitz, 2003b). Apart from that, to our knowledge there has so far been no evaluation of the usability of markup-based knowledge extraction in search application. Nevertheless, an evaluation of different types of Web search has shown that some guidance in helping the user reformulate a query can significantly improve the relevance of the retrieved documents compared to standard Web search (Bruza et al., 2000). Part of the study was the comparison of Web search using *Google* and search via the *Hyperindex Browser*, an interactive tool which passes the user query to a search engine and displays a list of phrases found in the top matching documents returned by the search engine. Those phrases can be selected by the user to constrain the query.

Recent experiments that have been conducted as part of the TREC interactive track have shown that hierarchical clustering and summarization can significantly help the user to locate relevant documents quickly. The experimental search system called *HuddleSearch* uses a new hierarchical clustering algorithm which dynamically organises a set of retrieved documents for a user query. Uncompleted tasks and average time to finish a search task were reduced by the use of *HuddleSearch* compared to a standard list based search engine (Osdin et al., 2003).

It is worth reviewing some more work on usability issues in the widest sense. A number of studies (ranging from preliminary investigations to some larger scale studies) have come to conclusions that are relevant here.

The work on the *HappyAssistant* dialogue system included user studies which showed that the number of clicks as well as the time spent on searching could be reduced significantly by replacing a menu driven system with natural language navigation. Interestingly, *"the vast majority of user queries (85%) were relatively short, and consisted of noun phrases or lists of keywords"*. They conclude that *"in such contexts, sophisticated dialog management is more important than the ability to handle complex natural language sentences"* (Chai et al., 2001). Studies have also shown that the users clearly prefered dialogue-based to non-dialogue based searches (Chai et al., 2002).

A comparison of interactive search in a homogeneous domain using a keyword-based search engine and one that uses an ontology is reported in (Sutcliffe and White, 2002). For the ontology-based system the documents were attached to the appropriate nodes in the ontology. The search started at the top level, proceeding step by step to the levels beneath. The application domain was word processing using the Lotus Ami Pro program. The result of the study was that the keyword-based system performed better than the one using the ontology. However, more interesting are the conclusions drawn from the study. One of them was that *"if an ontology search does not lead directly to a correct answer it is essentially a complete failure"*, unlike keyword-based search. The characteristics that were identified to be important when using some taxonomy to search were:

- It must be possible to decide easily which category at a level is relevant to the query;
- Categories should ideally be mutually exclusive;
- The level of branching at any level must be limited;
- The number of documents attached to leaf nodes must be small.

Observational experiments conducted in the TREC interactive track focused on finding out whether users choose a particular presentation type for a particular search task (Craswell et al., 2002). The different types of presentations were ranked document lists, a clustering interface and an integrated interface combining ranked lists,

clusters and expert links. The tasks were searches for single documents or sets of documents, respectively. It was concluded that users did not select a particular interface for a particular search task nor was any of the presentation types found to be superior for any particular task.

The question remains how the query modification suggestions the system presents are to be calculated. One can tackle the problem of matching user need against available data by processing the data into some conceptual model that can be used as a tool for guiding the user to a small set of relevant matches. The model can either be constructed offline or online by investigating the documents retrieved for the initial query. The system can then apply the model to help the user navigate through the answer set. There are at least two good reasons for this. First, the user gets a feel for what data is actually available. Rather than guessing new query terms, the system presents the query in context and proposes possible query refinement or relaxation terms. A second reason is to support a user who wants to browse rather than search a document collection, because "*current search mechanisms are not much use if you are not looking for a specific piece of information, but are generally exploring the collection.*" (Paynter et al., 2000).

The idea of automatically acquiring domain models from documents is not new. In the literature one can find different approaches to construct a model automatically. However, this research has so far not focused on the use of markup structure but on word co-occurrences or linguistic information, e.g. (Sanderson and Croft, 1999; Anick and Tipirneni, 1999; Lawrie and Croft, 2003). Furthermore, our approach can be performed automatically with no expert knowledge. It is largely independent of the actual language used in the documents and more importantly it uncovers the structure of a collection of documents while itself being domain-independent. Some more detailed discussion of related work in this area can be found in (Kruschwitz, 2003a).

So far we have mainly been looking at work that has to do with the presentation of query modifications and the construction as well as navigation of concept hierarchies. However, once the domain model has been constructed we want it to be automatically adjustable as part of the search system's functionality. This is quite a different research area.

"*Users usually will not make an effort to give explicit feedback but nonetheless leave implicit feedback information such as the results on which they clicked.*" (Henzinger et al., 2002). This is a commonly reported observation, which is why we record such implicit information by keeping track of the users' search behaviour. This differs from *explicit relevance feedback* (van Rijsbergen, 1979; Buckley et al., 1994) in that the user is not required to judge the relevance of query modification options or documents but instead the user's selections are being observed. The idea is similar to the one described in (White et al., 2002) where *implicit relevance feedback* is used to update the result set in an ad hoc search situation. If a user moves the mouse over a document title, then it is interpreted as an expression of interest in the particular document. However, we only use the actual user selections to update the domain model.

In that sense, we can see our approach as a particular application of collaborative filtering. Collaborative filtering is based on identifying the opinions and preferences of similar users in order to predict the preferences and to recommend items to others as used in systems to recommend news (e.g. *GroupLens* (Resnick et al., 1994)) or movies (e.g. *Video Recommender* (Hill et al., 1995)). Collaborative search has also been proposed for using similar past queries to automatically expand new queries, e.g. (Fitzpatrick and Dent, 1997) and (Raghavan and Sever, 1995) where documents that correspond well to similar queries also provide feedback on the original query.

Finally, it needs to be pointed out that we try to combine the best of several worlds: a standard search engine to deliver the best matching documents for a query (which will be sufficient for a large number of user queries), knowledge extracted from the documents at domain model construction time and finally terms found in the best matching documents for a particular query. That is we do not want to compete with approaches that have been shown to be simple and effective but rather enhance those techniques to end up with a system which is better than any of the individual approaches.

# 7 Conclusions

We can avoid expensive manual construction and customization of knowledge sources if we are able to perform such processes fully automatically. Intranets and other document collections are perfect candidates for such an approach since they contain some implicit structure suitable for the extraction of domain specific knowledge. In our case this leads to a domain model that consists of term hierarchies.

However, such a model will be static and in order to adjust and maintain the domain knowledge and make it a useful resources for search applications we combine it with knowledge extracted on the fly.

So far we have learned that users do prefer a search system which employs such domain knowledge over a standard search engine. This confirms results from an earlier evaluation that we performed using a different document collection. We also learned that users find such a search system easier to use than a standard search system.

We have also outlined how the automatically acquired domain knowledge can be improved by observing the users' selections in the search process. This will be the main research focus of our ongoing work.

## Acknowledgements

Thanks to BBCi for providing the query corpus. We also want to thank all the volunteers who took part in this evaluation.

We would like to thank the anonymous reviewers for their detailed and constructive comments which have hopefully all been addressed by the revised paper.

This project is supported by the Engineering and Physical Sciences Research Council (EPSRC) under grant GR/R92813/01.

## References

Anick, P. G. and Tipirneni, S. (1999). The paraphrase search assistant: terminological feedback for iterative information seeking. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 153–159, Berkeley, CA.

Beaulieu, M., Fowkes, H., Alemayehu, N., and Sanderson, M. (1999). Interactive Okapi at Sheffield - TREC-8. In *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*, pages 689–698, NIST Special Publication 500-246.

Bell, D. J. and Ruthven, I. (2004). Searcher's Assessments of Task Complexity for Web Searching. In *Proceedings of the $26^{th}$ European Conference on Information Retrieval (ECIR'04)*, Lecture Notes in Computer Science, pages 57–71, Sunderland. Springer Verlag.

Bruza, P., McArthur, R., and Dennis, S. (2000). Interactive Internet search: keyword, directory and query reformulation mechanisms compared. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 280–287, Athens, Greece.

Buckley, C., Salton, G., and Allan, J. (1994). The effect of adding relevance information in a relevance feedback environment. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information*, pages 292–301.

Chai, J., Horvath, V., Nicolov, N., Stys, M., Kambhatla, N., Zadrozny, W., and Melville, P. (2002). Natural Language Assistant - A Dialog System for Online Product Recommendation. *AI Magazine*, 23(2):63–75.

Chai, J., Lin, J., Zadrozny, W., Ye, T., Stys-Budzikowska, M., Horvath, V., Kambhatla, N., and Wolf, C. (2001). The Role of a Natural Language Conversational Interface in Online Sales: A Case Study. *International Journal of Speech Technology*, 4:285–295.

Craswell, N., Hawking, D., Thom, J., Upstill, T., Wilkinson, R., and Wu, M. (2003). TREC11 Web and Interactive Tracks at CSIRO. In *Proceedings of the Eleventh Text Retrieval Conference (TREC-2002)*, Gaithersburg, Maryland.

Craswell, N., Hawking, D., Wilkinson, R., and Wu, M. (2002). TREC10 Web and Interactive Tracks at CSIRO. In *Proceedings of the Tenth Text Retrieval Conference (TREC-2001)*, pages 151–158, NIST Special Publication 500-250.

Fasli, M. and Kruschwitz, U. (2001). Using Implicit Relevance Feedback in a Web Search Assistant. In Zhong, N., Yao, Y., Liu, J., and Ohsuga, S., editors, *Web Intelligence: Research and Development*, Lecture Notes in Artificial Intelligence 2198, pages 356–360. Springer Verlag.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Fitzpatrick, L. and Dent, M. (1997). Automatic Feedback Using Past Queries: Social Searching? In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 306–313, Philadelphia, PA. ACM.

Henzinger, M. R., Motwani, R., and Silverstein, C. (2002). Challenges in web search engines. *SIGIR Forum*, 36(2):11–22.

Hersh, W. (2003). TREC 2002 Interactive Track Report. In *Proceedings of the Eleventh Text Retrieval Conference (TREC-2002)*, Gaithersburg, Maryland.

Hersh, W. and Over, P. (2001). TREC-9 Interactive Track Report. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9)*, pages 41–50, NIST Special Publication 500-249.

Hill, W., Stead, L., Rosenstein, M., and Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'95*, New York. ACM.

Jansen, B. J., Bateman, J., and Saracevic, T. (1998). Real life information retrieval: A study of user queries on the web. *SIGIR Forum*, 32(1):5–17.

Joho, H., Sanderson, M., and Beaulieu, M. (2004). A Study of User Interaction with a Concept-based Interactive Query Expansion Support Tool. In *Proceedings of the $26^{th}$ European Conference on Information Retrieval (ECIR'04)*, Lecture Notes in Computer Science, pages 42–56, Sunderland. Springer Verlag.

Katz, B., Lin, J., and Felshin, S. (2001). Gathering Knowledge for a Question Answering System from Heterogeneous Information Sources. In *Proceedings of the ACL/EACL Workshop on Human Language Technology and Knowledge Management*, Toulouse.

Kruschwitz, U. (2003a). An Adaptable Search System for Collections of Partially Structured Documents. *IEEE Intelligent Systems*, 18(4):44–52.

Kruschwitz, U. (2003b). Automatically Acquired Domain Knowledge for ad hoc Search: Evaluation Results. In *Proceedings of the 2003 International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*, pages 525–532, Beijing. IEEE.

Lawrie, D. J. and Croft, W. B. (2003). Generating Hierarchical Summaries for Web Searches. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 457–458, Toronto, Canada.

Margennis, M. and van Rijsbergen, C. J. (1997). The potential and actual effectiveness of interactive query expansion. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 324–332, Philadelphia, PA. ACM.

Osdin, R., Ounis, I., and White, R. W. (2003). Using Hierarchical Clustering and Summarization Approaches for Web Retrieval: Glasgow at the TREC 2002 Interactive Track. In *Proceedings of the Eleventh Text Retrieval Conference (TREC-2002)*, Gaithersburg, Maryland.

Paynter, G. W., Witten, I. H., Cunningham, S. J., and Buchanan, G. (2000). Scalable browsing for large collections: a case study. In *Proceedings of the $5^{th}$ ACM Conference on Digital Libraries*, pages 215–223.

Raghavan, V. V. and Sever, H. (1995). On the reuse of past optimal queries. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Feedback Methods, pages 344–350.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, pages 175–186.

Ruthven, I., Tombros, A., and Jose, J. M. (2001). A Study on the Use of Summaries and Summary-based Query Expansion for a Question-answering Task. In *Proceedings of the $23^{rd}$ European Colloquium on Information Retrieval Research (ECIR'01)*, pages 41–53, Darmstadt.

Sanderson, M. and Croft, B. (1999). Deriving concept hierarchies from text. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 206–213, Berkeley, CA.

Silverstein, C., Henzinger, M., and Marais, H. (1998). Analysis of a Very Large AltaVista Query Log. Digital SRC Technical Note 1998-014.

Sutcliffe, R. F. E. and White, K. (2002). Searching via keywords or concept hierarchies - which is better? In *Proceedings of the $3^{rd}$ International Conference on Language Resources and Evaluation*, pages 2103–2106, Las Palmas de Gran Canaria, Spain.

van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths.

Voorhees, E. and Harman, D. (2002). Overview of TREC 2001. In *Proceedings of the Tenth Text Retrieval Conference (TREC-2001)*, pages 1–15, NIST Special Publication 500-250.

Walker, M., Kamm, C., and Litman, D. (2000). Towards developing general models of usability with PARADISE. *Natural Language Engineering*, 6(3).

White, R. W., Ruthven, I., and Jose, J. M. (2002). The Use of Implicit Evidence for Relevance Feedback in Web Retrieval. In Crestani, F., Girolami, M., and van Rijsbergen, C. J., editors, *Proceedings of the 24$^{th}$ European Colloquium on Information Retrieval Research (ECIR'02)*, Lecture Notes in Computer Science 2291, pages 93–109. Springer Verlag.