RC 21212(94717)16JUN98 **Computer-Human Interaction**

Research Report

Observations On User Expectations About Query Input For Web Search

Birgit Schmidt-Wesche
IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

Research Division

Almaden - Austin - Beijing - Haifa - T. J. Watson - Tokyo - Zurich

LIMITED DISTRIBUTION NOTICE: This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents. In view of the transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies of the article (e.g. payment of royalties). Copies may be requested from IBM T. J. Watson Research Center [Publications 16-220 ykt] P. O. Box 218, Yorktown Heights, NY 10598. email reports@us.ibm.com
Some reports are available on the internet at http://domino.watson.ibm.com/library/CyberDig.nsf/home

Observations on User Expectations about Query Input for Web Search

Birgit Schmidt-Wesche
IBM TJ Watson Research Center

INTRODUCTION

Users of web search engines expect to have access to a variety of general search features. However, the specific ways in which these features are implemented vary, and users' expectations may not always match these implementations. In this paper, we discuss how well search features of web search engines currently meet users' expectations.

Many search features are taken for granted nowadays. Users expect a query input field with the freedom of either freely entering natural language queries, or controlling the structure of their input through Boolean logic or sometimes through the usage of the '+' and '-' operators attached to their search terms. Users also expect the ability to choose the collection to search, including the Web and Usenet, and to constrain the search through a 'date' filter. At the same time, the scope and flexibility of web search engines increase regularly, in turn giving rise to even greater expectations.

When search engines extend their capabilities, the goal - besides improving the search process - often is to provide easier usage. For the query input field, for example, we experienced the evolution from strictly controlled Boolean expressions, to freely entered key words, accompanied by the symbolic '+' or '-' operator to assign weightings to selected terms, to natural language type input such as "Give me some information on German soccer". Some search engines now suggest additional key terms related to a user's query terms, to be added automatically to the original query terms, and thus give direct support in formulating and refining a query. With this freedom, making queries has become potentially easier, but at the same time this greater freedom and flexibility seems to be a source of misconceptions about how the search engine processes free text query specifications.

In the following, we restrict ourselves to examining textual query input and search on the web. Of course, users have specific expectations of search engines that are equipped with a catalog or a taxonomy of categories to search in, but navigating through them by following the provided hyperlink paths is an issue in its own right. We also will not discuss the various forms of search result displays here. They may have a direct influence on the refinement of the query, since the interaction of query input and search result may automatically correct a user's initially wrong expectations. But the question that guides us here is whether we can tailor our search engine so

that we can present an unambiguous query form in the beginning, with queries being processed according to user expectations.

The issues we present here are collected from two user tests performed in 1997. We also gathered user comments from search engine reviews in various trade journals. The search engine used for the two user tests was IBM NetQuestion (1998), a component of the IBM Intelligent Miner for Text product (1998). Among others, IBM NetQuestion implements the search facility on IBM's internal and external web home pages.

Eight colleagues from IBM TJ Watson Research Center, including the leading librarian, volunteered as test subjects for our first test. For the second test, we hired four temporary employees from an employment agency used by IBM Research. Both groups consisted of people that were experienced in using web search engines as well as people who only occasionally used web search engines.

In the first test, we tested an HTML-form based web search application with separate text entry fields for Boolean and Free Text entry fields. In the second test, we participated in the Interactive Track of TREC-6 (see Schmidt-Wesche et al. (1997)). The user interface for this test was a Java application with tilable windows for query entry, search result, and document display. The query entry field accepted queries in '+/-'syntax, in the sense of the tacitly accepted web standard. The queries were uniformly translated into hybrid queries by simultaneously exploiting the Boolean and the Free Text search mode with each query. Our TREC-6 prototype also featured a 'lexical navigation' component, which offered the users terms related to their query terms (see Cooper and Byrd (1997)).

STRUCTURED VERSUS UNSTRUCTURED INPUT

We propose a global distinction between structured versus unstructured query input, referring to the presence or absence of a specific query syntax, respectively.

We distinguish between three types of users' *unstructured input*:

1. Sequence of query terms:
   Deliberately chosen single query terms.
2. Natural language input:
   The crucial query terms (in the following example, 'e-business') are embedded in a natural language expression such as "Give me some information on e-business.".
3. Specification of concepts, or conceptual search:
   The user provides a description or a paraphrase of the item he or she is looking for without naming the actual item itself. Thus the search engine will have to interpret the user's input to infer the requested items. Examples of this type we encountered with queries such as 'wildlife protection programs' (generic, mostly in form of plurals), or 'ferry sinking events with more than 100 people dead' (verbose description). In both instances, the users expected the search engine to render specific instances of what they described, i.e., names of specific

wildlife protection programs, or names of ferries or locations of ferry sinking events in which more than 100 people died.

The following list shows search features that searchers expect to see with a sequence of query terms or natural language type input:

- Notation to mark consecutive query terms as a unit to specify a phrase like 'The United States of America', where the terms appear in exactly this order and adjacent to each other
- Identification of relevant content words as opposed to grammatical elements such as interrogatives (who, what, etc.)
- Recognition of variants of query terms:
  a) Morphological variants
  The user just enters one morphological form, and the search engine will automatically look for others such as the plural form or various inflections.
  b) Proper name and term variants
  The same expectations as for morphological expansions apply to established (multi-word) *terms* such as 'machine translation'. But here, the search engine is expected to recognize this multi-word term as a unit, and to expand it appropriately to its corresponding plural and case forms ('machine translations'), instead of expanding each constituting word (i.e., all variations of 'machine' and all variations of 'translation').
  Also, users expect a search engine to find all variants of a *proper name* they specify, without them having to detail each single occurrence. For example, when asking for 'Bill Clinton' they would also like to see documents containing 'William J. Clinton' or 'President Clinton'. Especially, they expect acronyms to be resolved to their full form and vice versa ('IBM' and 'International Business Machines').

While phrase marking and content parsing may also be expected with conceptual queries, the expectation of recognizing term variants is not found with the conceptual type of queries. At most, the user here expects to be presented with variants of his or her intended concept. Referring to our above example, this could mean different names for one and the same wildlife protection program.

The *structured input* devices we discuss here are:

- Usage of symbolic +/- syntax:
  Using '+' to strengthen the weight (i.e. importance) of a term, and '-' to exclude a term.
- Usage of Boolean syntax:
  Using Boolean operators AND, OR, NOT and creating nested expressions.
- Proximity filter:
  Specification of the distance between two terms, that is how many words at most should intervene between them.
- Special functions for querying meta-information:
  Restrict the search to a certain domain, a specific URL, or a specific section of a document.
- Date filter:
  Restrict the search to documents last updated at a specific point or period of time.

In the following sections, we discuss in detail structured and unstructured query input, and outline the user expectations we found during our tests, collected from our investigation of web search engines, or found mentioned in trade journal articles. With each item, we state the relevant user expectation, describe how the various web search engines address these issues, and, where appropriate, discuss alternative approaches we believe better meet user expectations.

UNSTRUCTURED INPUT

We discuss the types of unstructured query input and the corresponding search in the order we introduced them above.

Sequence of Query Terms

There are two prominent expectations in connection with a query that is composed of a sequence of deliberately selected query terms. The first one also applies to natural language type input, while the second does not:

1. Conjunctive reading of query terms
2. Left-to-right reading of query terms

Conjunctive reading of query terms

Expectation: All terms are ANDed.
This statement captures the fact that the users expect the best, i.e. topmost, result documents to contain **all** of the entered query terms. For example, the author of 'Using Droids & Agents' in the ONLINE magazine of Nov./Dec. 1997 states that for a search such as 'marketability discount' he prefers "search engines [that] read the phrase as 'marketability AND discount'", over those that "do not assume the Boolean AND and therefore read the phrase as 'marketability OR discount'", and that he therefore likes to use AltaVista in these contexts.

However, since users often also expect other documents with only some of the query terms, it indicates that they do not always have a strict conjunction in mind.

Current Implementations
Most of the web search engines, including IBM's web search facility, process a disjunction of the entered query terms, which means that they look for documents that contain any, but not necessarily all of the terms. An exception here are AltaVista and Hotbot. HotBot processes a disjunction, but ranks documents that contain all of the entered terms higher, so that the users' expectation of seeing the 'all-terms' documents first is met. AltaVista takes this approach one step further: not only do they list the 'all-terms' documents firsts, but they also distinguish between those where all of the terms occur in the body of the text and those where all of the terms occur in the title, and list the latter first, followed by the ones with all terms in the body, followed by the remaining documents that contain some but not all of the terms.

Recommendations
There are two general ways the users' conjunctive expectation could be met:
1. Re-arranging the document result list according to the query terms they contain.
2. Processing a conjunction, followed by the usual disjunction if necessary.

These suggestions are discussed in detail during the discussion of using '+' and '-' operators, where we will make the same two recommendations to overcome misunderstandings with search expressions constructed as 'termA +termB'.

Left-to-right reading of query terms

Expectation: A sequence of query terms is read as a successive modification from left-to-right: (((A) B) C).

This expectation comes in two flavors:
First, there is the stepwise view, where a user starts with entering the query *termA*, searches and refines it to *termA termB*, searches and finally enters *termA termB termC*. With this sequence, the user expects to successively constrain the document result set with each refined query, rather than the query being interpreted as a completely new query with each iteration.

Second, there are users who enter all terms and expect the search engine to automatically conduct the search as if it were successively modifying the result sets. They claim that the leftmost query term is always their most important term, followed by potential restrictions on this main term. This second view was confirmed by one of our test subjects during the TREC-6 Interactive Track test (August 1997).

Current Implementations
While the expectations of this second group of users are difficult to meet, the former expectation is easier to deal with. The user could be given the choice of the target of his refined search, that is whether it should be applied against the present result set or again against the entire collection. Indeed, a number of search engines already offer this choice (e.g., Infoseek, Yahoo). With Infoseek, the user can also directly instruct the search engine to conduct the refined search against the first result set by using their pipe command, which is stated as 'termA | termB'.

Recommendations
No web search engine automatically processes a left-to-right refinement as expected by the second group of users. But, those search engines that process a conjunction of query terms first (as discussed above), indirectly meet the users' expectations here, too, since a successive modification of result sets would result in the set of documents that contain all of the entered query terms. Therefore, if the user sees those 'all-term' documents first, he will be satisfied.

Search Features Connected to a Sequence of Query Terms

In this section we discuss the notation for marking phrases, and a search engine's capability of expanding the query terms into variants.

Phrase markers can also be seen as proximity operators, more specifically as adjacency operators, since in a phrase the terms in questions occur adjacent to each other. Therefore, we could have classified phrase markers just as well under 'proximity' in structured input. The reason to list this device here is that phrase markers explicitly delimit multi-word terms that then will be treated as single terms by the search engine. In addition, they can be assigned a '+' or '-' operator, or occur as a Boolean argument.

Phrase Markers

Expectation: A series of query terms is treated as a unit.
It is a notation to determine that a sequence of terms should occur in exactly the given order, whether we deal with an established phrase like 'United States of America' or an arbitrary list of terms. It is also a nice device for requiring that the terms specified in a phrase be treated literally. For example, if the terms should occur in capital letters, the user can mark the pertaining terms as a phrase, and they will not be case normalized.

Current Implementations
All search engines provide phrase markers in one way or another, mostly as quotes to delimit a phrase, but hyphens are also used (Infoseek), or a phrase is offered as an item in a pull-down menu (HotBot). NetQuestion supports phrases via the means of quotes. Hyphens are not offered, since typographically they are synonymous with the '-' sign that is used as an operator of the '+/-'syntax. In IBM's search facility on its internal and external web pages, hyphenated words are automatically put in quotes in order to avoid any type of confusion with an excluded term.

Variants of query terms

We subsume under this header morphological variants and variants of proper names and established multi-word terms.

Morphological variants

Expectation: This feature is automatically provided.
For example, if a user enters a verb form in its infinitival form, he expects the returned documents to also contain inflections of this verb. For a noun the expectation is to get all number (singular or plural) and case forms.

Current implementations
Most search engines nowadays automatically support this feature. However, with AltaVista and HotBot the user still has to specify each morphological variant. The reason may be that for some highly inflectional languages, automatic expansion is quite an expensive operation. For example,

the German irregular verb ‘gehen (to go)’ has sixteen inflected variants that would have to be searched for. Since AltaVista does not search for morphological variants, it recommends the explicit usage of the asterisk wild card character, which may stand for all possible linear extensions of a term (e.g., *ground-hog** will expand to the singular *ground-hog* as well as the plural *ground-hogs*).

NetQuestion offers the choice between searching for all morphological variants (called ‘fuzzy search’) or making precise searches (‘exact search’) in which it only looks for the terms as specified. For IBM’s web search facility, only exact search has been adopted. As with AltaVista, the usage of the asterisk wild card character is recommended for finding morphological variants.

Variants of proper names and established terms

Expectation: The search engine recognizes the corresponding terms as a single item and conducts the search accordingly.

Current implementations
To our knowledge, no search engine directly handles, for example, acronym to full form correspondences, or different denotations of the same person or institution. In “The Dollar $ign - Where did they hide my company?” (ONLINE, Nov./Dec. 97) the author expresses her disappointment with the fact that “search engines such as AltaVista and HotBot interpret company names quite literally. If you ask it to search for ALCOA, there is no built-in intelligence to equate ALCOA with Aluminum Company of America. Of course, if someone on a Web page misspelled ALCOA as ALCAO, you won’t get that either. ... [Also,] ALCOA and IBM, searched in the Company category of Yahoo!, were unrecognized.” Established multi-word terms, too, may only be recognized if put in quotes, and their variants only via the wild card option.

NetQuestion’s specific type of spell checker is able to adequately treat misspellings of a term as variants of the intended spelling of a query term. The spell checker is not dictionary based, but operates on frequencies of similarly spelled words.

Recommendations
Full-form-to-acronym correspondences could, at least partially, be treated by the incorporation of a lexical navigation component. For example, our TREC-6 prototype provided terms related to the query terms by co-occurrence in the document corpus (see Cooper and Byrd (1997) for details). In some cases, acronyms of query terms or variants of proper names appeared as related terms. The same may be true for the related terms offered by AltaVista or Excite. But methods that specifically address the disambiguation of names and terms as described in Wacholder et al. (1997) will be even more helpful in this context.

Natural Language Input

Many issues of structured input apply to natural language input as well. In addition, natural language input can consist of text that has been cut and pasted into the query field.

Expectation: The search engine can correctly parse the input and detect the relevant key words without misinterpreting special signs or conjunctions such as 'and' or 'or' as logic connectors used in the context of Boolean syntax.

Current Implementations
Some years ago, several attempts had been made to create natural language query alternatives to structured relational database query languages like SQL (for details, see, e.g., Ogden (1988)). Today, to our knowledge, no search engine (at least among the web search engines) truly parses natural language syntax. But the cut-and-paste functionality is supported by all current web search engines, including IBM's web search facility, and the query is treated as regular free text input.

However, not every user knows of the possibility of entering long free text queries, and many search user interfaces do not invite this search mode when they only offer small query input fields: "More complex queries often require more space than the single line entry box... The more complex interface supports a much longer query." (Rosenfeld and Morville; 1998, p. 108).

Conceptual Queries

By conceptual queries we mean queries where the user renders a description and expects the search engine to interpret this description and find specific instances of the derived concept, event, or item, or anything that is related to it.

Expectation: The search engine finds instances of what was described or something related to it.

Current implementations
At best, the search engines that offer related terms (AltaVista, Excite) could come close to meeting this user expectation, since the expected instances may be captured by related terms. But "these attempts to predict the user's search intentions [are found] only occasionally on target, and sometimes simply distracting" (PC Magazine, "Searching the Net", Dec.2, 1997).

In addition to offering related terms, Excite claims "to search the Web by concept. Like most search engines, [they]'ve programmed [their] search engine to look for documents containing the exact words you entered into the query box. But Excite goes further and looks for ideas closely linked to the words in your query. This feature broadens your search." (cf. Excite Help Text). In a search engine survey in Internet World (Dec. 97), the authors claim that "a query for 'martial arts' finds sites about kick-boxing and karate even if the original search term isn't in the page." The author re-ran this query in Excite, but could not confirm this observation for the first 210 documents of the result list.

Recommendations

A) Incorporating lexical navigation tools
Even though the related terms currently offered by search engines may not be perfect yet, we believe that much value lies in offering lexical items related to query terms, and the ability to navigate among them.

For example, one of the tasks in the TREC-6 test was to identify documents that name drugs and their manufacturer for the treatment of the Alzheimer's disease. When the users asked for terms related to the query 'Alzheimer's drug treatment', our TREC-6 prototype offered them 'Cognex' (an Alzheimer's drug) as a term co-occurring with their query terms, and 'Warner&Lambert' (its manufacturer) as a term related to 'Cognex'. The users were very pleased with this result, added co-occurring and related term to their original query terms and thus were able to find the relevant documents.

B) Offering result categories (like Yahoo or Infoseek do)
Along with the usual list of relevant result documents, Yahoo searches its catalog for category names that match (all or some of) the query terms, and offers these categories as a result, to be further navigated by the user.

It would be even more helpful if search engines with categorization capabilities could take this simple category name matching algorithm one step further: first find the documents relevant to the query, and then determine and display the corresponding categories. With each category, one could view the contained documents that resulted from the query or choose to browse the entire category. Approaches into this direction can be found with IBM WebCat (see Chang et al. (1996)).

STRUCTURED INPUT DEVICES

To some extent the following list is arbitrary, but it can also be read as from more confusing to less confusing. The usage of '+/-' syntax seems to give rise to the most types of misunderstandings, while a date filter seems to have fairly the same interpretation for every user.

Usage of '+/-' Syntax

Here, we focus on the difficulties users have with a search expression constructed as 'A +B'. There are two different types of expectations, but the result of both would be the same.

Expectations:
A) TermB constrains the 'termA' document set; in other words, out of all documents with termA, retrieve those that also contain termB.

B) The expression ‘A +B’ is synonymous with the Boolean expression ‘A AND B’, where the search engine looks for documents that contain both termA and termB.

The document result set would be the same with either expectation, i.e., the set of documents that contain both query terms. But both expectations are inappropriate if the selected search engine behaves according to the tacitly accepted rules for ‘+/-’ syntax, by which the ‘+’ and the ‘-’ operators are treated as unary operators, operating on the following argument only, as opposed to the Boolean AND and OR operators which are binary and operate on the two given arguments, joining them to a logic expression. So, while ‘A AND B’ results in a set of documents that contain both, termA and termB, ‘A +B’ results in a set of documents that necessarily contain termB, but only optionally contain termA. This stems from the fact that with the ‘+/-’syntax, the assignment of a ‘+’ operator to a termX is read as ’must contain X’, while an unsigned termY is taken as ‘may contain Y’.

The expectation that the expression ‘A +B’ is synonymous with the Boolean expression ‘A AND B’ is further nourished by remarks like this: “... you may add the Boolean operator AND to a search for ‘athletic ANDequipment’, and the engine will return only Web site titles that include both words. Symbolic operators, such as the plus (+) or minus (-) signs work much the same way.” Or: “Two ... symbolic operators are the plus (+) and minus (-) symbols. Use them as you would the AND and NOT operators.” Both comments appeared in introductions to the usage of world wide web functions such as search services (Witulski, LaChance (1997) in ‘PC Novice - Guide to Going Online’).

Current Implementations

- AltaVista: AltaVista lists as topmost documents those that contain all of the query terms, starting with documents that contain the query terms in their titles, followed by those that contain the query terms in their body (followed by the remaining documents with only some of the query terms).
- HotBot: They basically follow the rules of the ‘+/-’syntax, with a slight adjustment. With a query ‘A +B’ they will return documents that necessarily contain termB as requested, but will rank documents that also mention termA higher. So the user first receives documents with both terms.
  In addition, their initial default setting of the search mode is ‘look for all of the words’, which means a conjunctive reading, unless the user explicitly changes this setting.
- Excite, Infoseek: They simply treat expressions with signed and unsigned terms by the so-called rules. For a query ‘A +B’, this means that a user may see documents that only contain termB. This is especially disappointing if he or she expects the search engine to construct the termA result set first, and then refine it with termB. With this expectation, the user expects to see documents that necessarily contain termA, and if given also termB. But exactly the opposite is calculated by the search engine.

IBM’s web search application of NetQuestion offers an elegant solution for situations where users have the expectation that we listed in A) above. In case the user constructs a query sequentially, the procedure is as follows:

- When a user enters a query consisting of a single *termA,* the query is explicitly re-stated as *+termA* in the query entry field, which is equivalent to *termA* in case of a single term query.
- When he or she continues to refine the query with *+termB*, the query will then read: *+termA +termB*

This expression will return documents with both terms, as expected by the user. If a user does not intend this interpretation, he will need to erase the '+'operator in front of *termA*. If, on the other hand, his expectation was to find documents that eventually contained both terms, he will be satisfied with the results.

However, no such action is taken when the user directly enters the expression *termA +termB*, since substituting *termA* for *+termA* in this context will not result in an expression equivalent to the original one.

Recommendations
Suggestion for meeting expectation A):
Another way to meet expectation A), if query 'A +B' is constructed sequentially, would be to offer the choice of searching either the result list or again the entire collection with a refined search expression (as, e.g., Infoseek or Yahoo do). In case the user sequentially builds the query 'A +B', searching the document result list will lead to the desired output, because all of the documents here already contain termA, and since the refined query requires that they must also contain termB, the result set will consequently contain both terms.

Suggestions for meeting expectation B):
As mentioned above, we see two possibilities here:

1. *Re-arranging the result list*
   A simple re-arranging solution could be adopted if the search engine can provide the list of query terms contained in a result document, which will enable the corresponding re-ordering of the result documents. Documents containing all of the query terms could be listed at the top of the result list, followed by the remaining documents, i.e. documents containing n-1 query terms, etc.
2. *Conjunction instead of disjunction*
   First, translate all query terms entered by the user (except for the minus terms) into a conjunction. If the result on the basis of this conjunction consists of less than the requested amount of result documents (assumed default 100), then also calculate the result set on the basis of the plus terms as originally designed, and fill up the result list to the requested limit.

A concern with the second suggestion is that duplicates occurring in the results from both the conjunction and the disjunction query need to be detected and eliminated. Furthermore, it would be difficult to account for the specification of synonyms as with a set like "search retrieve find locate" or encoded as "search {retrieve, find locate}". In cases like this, the user wants to see documents with either but not all of the query terms contained in the set. Our test experience, however, suggests that these types of queries are not typical for web search applications.

With both alternatives, we will not be able to display the actual relevancy ranking information of a document anymore. With suggestion 1., we re-arrange the result list as originally ranked. With suggestion 2., we process two lists and concatenate them. In each case, a coherent ranking of the new result list may become most difficult.

Usage of Boolean Syntax

Expectation: All recorded expectations met the actual behavior of Boolean syntax.

Current Implementations
Almost all major web search engines provide a Boolean search mode (except for Infoseek). The 12Boolean operators provided include at least AND, OR, and NOT, and often also NEAR (see next section). All search engines with Boolean search allow for nesting of subqueries through the use of parentheses.

In the context of search engines, the usage of Boolean syntax is often referred to as 'advanced search'. Some of our test subjects, however, told us that for them this was the simple and easy mode of search. The usage of the three operators AND, OR, NOT is straightforward and leaves much less room for misinterpretations than the usage of the symbolic '+' or '-' operators. The AND operator especially seems easy to apply, since it resembles the mathematical operation of addition.

These judgements do not apply to the full range of Boolean search expressions. Complex, multiply nested Boolean expressions were still considered difficult to construct. But in particular with short queries, our test subjects preferred the usage of Boolean syntax over the usage of the '+/-'syntax in order to avoid the misunderstandings of the sort we discussed in the previous section. Also, when looking for both terms in a two term query, our test subjects found it more intuitive to formulate the query as 'termA AND termB' instead of '+termA +termB'.

You also find recommendations to drop the Boolean search mode, at least from the primary search interface (Nielsen, 1997). Instead, it should only be offered on a different page that is clearly marked as *advanced search* in order to 'intimidate' the novice searcher and only invite the expert user. Nielsen's recommendation is based on observed difficulties with formulating a query as 'termA AND termB', where in fact the user rather intended to look for 'termA OR termB'.

In general, there does not seem to be a consensus as to which type of query language is the easiest to use. Investigations have focused on several comparative evaluations. Early comparisons of formal relational versus natural language query formulation (e.g., Shneiderman, 1978) showed no convincing evidence for an advantage of natural language over formal language queries. Turtle (1994) reports that query results were significantly better when queries were formulated in natural language than in Boolean logic. Comparing Boolean and natural language queries, Hersh and his colleagues found that in a number of studies the search results were comparable (cf. Hersh et al., 1995). But Hersh and Day (1997) report that their results in the TREC-6 Interactive Track showed that a natural language search system was superior to a

Boolean one (it needs to be noted, though, that the tested natural language system ZPRISE (1998), developed at the National Institute of Standards and Technology NIST, is rather a free text search system than a natural language search system with full-fledged NL parsing capabilities). While the former studies predominantly focussed on retrieval performance, the study by Hersh and Day also emphasizes that the searchers generally ranked ZPRISE as easier to use than their Boolean experimental system (the TREC-6 Interactive Track guidelines required the application of a series of questionnaires, of which one of them explicitly asked for the searchers' ease-of-use judgements).

To our knowledge, there are no studies that explicitly address differences between systems that offer Boolean operators and others that offer the symbolic plus (+) and minus (-) operators. But our searchers' comments suggest that symbolic and Boolean syntax may be equally difficult (or easy) to use.

Specification of Proximity

The proximity filter usually determines how far from each other two query terms are expected to occur.

Expectation: This filter is mostly expected to be provided, without any detailed requests or ideas of how it should function.

Current Implementations
The major web search engines usually provide a binary NEAR operator, for example:

- Lycos: NEAR[25] is the default setting, meaning that termA and termB should occur within 25 words from each other, if specified as 'termA NEAR termB'. This setting can be changed by explicitly specifying NEAR/n instead. Lycos, different from any other search engine, also offers a FAR operator, with the default value of >25, which again may be changed by the user to his or her own setting using FAR/n.
- AltaVista: Their NEAR operator has a default setting of 10, and is not customizable. This is true with other search engines, too.

Proximity operators can also be n-ary, i.e., not restricted to two arguments, and not tied to a range of intervening words, but instead referring to the sentence (SENT) or paragraph (PARA) boundary (as, e.g., in IBM's web search application). If neither is specified, the document boundary is assumed. In addition, two or more terms can occur adjacent (ADJ) to each other, which amounts to a phrase. This treatment corresponds to the scope of proximity of *n* terms as defined in the Z39.50 standard of information retrieval (1998 / 1988). Thus, the user can specify: SENT(termA termB termC etc.), or analogously PARA(termA termB termC etc.) to request that the specified terms occur within a sentence or within a paragraph. Users seem to find these types of proximity operators more intuitive than the commonly provided NEAR operator.

Meta-Information

The set of specific meta information that can be used to constrain a search seems to vary widely. Some of the functions supported by almost every search engine include searching for a specific domain (.org, .edu), a location (.uk, .de), as well as choosing between searching in the title versus the body of a document.

Expectation: Meta information functions are available.
Users expect to have specific functions available with which they can search for documents by exploiting their pertaining meta-information. But, except for requesting the mere availability of these functions, we have not experienced any more detailed user expectations here.

Current Implementations
Quite a few web search engines support searches organized by meta-information now. IBM's web search application supports the distinction between searches in document titles versus searches in the entire document (including the title). AltaVista probably offers the largest variety of functions, while HotBot probably offers the functions in the best structured and most user-friendly way. Searches using specific functions do not necessarily have to be entered through an explicit query language format (as with, e.g., AltaVista): With HotBot the input can be enabled via specifically designed interface controls, such as pull-down menus, check boxes, etc.

Date Filter

We distinguish between two different aspects of date: Dates mentioned within the body of a document and dates as meta information, which define when a document was last updated or last crawled. Because 'date' seems to be the type of meta information that is mentioned most often when users are asked for their requirements in this context, we discuss it separately here.

Expectation: The date filter function (last updated) is expected to be available.
Nowadays almost every user expects to be able to filter his search by the date of the last update. We are not aware of any detailed user expectations as to date specifications occurring within the body of a document.

Current Implementations
Even though it is a highly required function, only three search engines currently seem to support the date filter:

- AltaVista, where a user can specify not only points in time, but also periods of time (from...to)
- HotBot and Yahoo, where a user can specify a point in time, from when the documents should date (from)

All major search engines search for dates mentioned within documents, and even recognize variations (21 January 1957, Jan.21, 1957, etc.). Yahoo calls AltaVista when queried with a date

specification. AltaVista seems to yield the most precise and HotBot the most extensive result list (including some curiosities such as documents containing 'Stories by Jan Neruda, 1957').

FURTHER MISCELLANEOUS SEARCH OPTIONS

In this section, we briefly list a number of search options that some search engines have already incorporated into their frameworks, but of which we have little user experience to report:

- Searching for people, instead of documents.
  HotBot offers this feature as a type of adjacency search: name and surname are taken to be search terms that must occur adjacent to each other in any order.
- Searching for documents in a specific language.
  AltaVista seems to offer the most comprehensive list of different languages.
- Searching for specific media types (audio, video) or formats (Java, VRML).
  Refer to HotBot on this subject matter.
- Saving users' specific search page settings and saving the search results are often expressed as requirements.
  Again, HotBot has taken a first step in this direction.
- Offering national versions.
  The more people become familiar with the web, the more it will be necessary to offer national versions of applications including web search engines. Currently, Yahoo offers the most extensive list of national versions.
- Translating documents to and from English.
  Recently, AltaVista and Systran joined to offer this feature for five European languages.

SUMMARY

We have described a series of most common search features in this paper from the perspective of which features users require most, and how they expect these features to behave.

In summary, users appear to experience much less difficulty with structured query input than with unstructured query input (yet, it should be kept in mind that mastering structured query input requires some training). On the other hand, the more freely a query can be entered the more room there is for misinterpretation of what the search result will be.

In particular, we reported on misunderstandings with conceptual queries where users expected to get documents that mention specific instances of the query term instead of mentioning the query term itself. Support could be found in lexical navigation tools that, on the basis of the indexed document collection, establish and display to the user a network of terms that co-occur with and are related to the original query terms. The user can navigate through this network, browsing relevant terms, and adding them to the original query in order to refine it.

For search engines with categorization capabilities, the display of categories that were calculated to contain documents that match a user's query, is also a useful and helpful device. The user can then view the contained documents that resulted from the query or browse the entire category.

Furthermore, we reported on misunderstandings in the following contexts:

1. Simple sequence of unsigned query terms, which is expected to be ANDed instead of ORed, as practiced by most search engines.
2. Simple sequence of unsigned query terms, which is expected to be read from left-to-right: (((A) B) C).
3. Query that is constructed as 'A +B', which is expected to be read either as synonymous with 'A AND B', or as left-to-right in the sense of '((A) B)'.

For all three cases, though different in nature, a common solution could lie in adopting a conjunctive approach. With a conjunctive approach a search engine would always aim at first presenting documents that contain all of the terms of the query. We have offered various suggestions here as to how to achieve this goal so that we can better meet the expectations of web search engine users.

ACKNOWLEDGEMENTS

We are grateful to Eric Brown, John Prager, and Robert Mack from IBM Watson Research Center, and Monika Matschke, Rolf Baeurle, and Andreas Prokoph from IBM SWSD Boeblingen for the many fruitful discussions that preceded the elaboration of the issues in this paper. We also want to thank Yael Ravin from IBM Watson Research Center for her careful editorial reading of this paper.

REFERENCES

Chang, R., Brown, E., Chong, H., Mak, S.W., Prager, J., and So, E. (1996). *Web Hypermedia Digital Library on the 'Net'*. In: Proceedings of "Modern Engineering & Technology Seminar", Taipei, Taiwan, Dec. 1996. pp.214-220.

Cooper, J. and Byrd, R. (1997). *Lexical Navigation: Visually Prompted Query Expansion and Refinement*. Proceedings of DIGLIB97, Philadelphia, PA, July, 1997. pp.237-246.

Excite (1998). Search engine: http://excite.com. Search Tips: http://excite.com/Info/searching.html?a-tip-t.

Hersh, W., Elliott, D. et al. (1995). *Towards new measures of information retrieval evaluation.* Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA. pp. 164-170.

Hersh, W. and Day, B. (1997). *A Comparison of Boolean and Natural Language Searching for the TREC-6 Interactive Task.* In: Voorhees, E. and Harman, D. (eds.): Proc. 6th Text REtrieval Conference (TREC-6), Gaithersburg, MD, Nov.19-21, 1997. To appear.

IBM Intelligent Miner for Text (1998). Information about IBM Intelligent Miner for Text can be found at: http://www.software.ibm.com/data/iminer/fortext/.

IBM NetQuestion (1998). Information about IBM NetQuestion can be found at: http://www.software.ibm.com/data/iminer/fortext/webEnabled.html.

LaChance, M. (1997). *Exploring the Web With Multiengine Search Tools*. In: PC Novice - Guide to Going Online. Vol.6, Issue 1. pp.95-97.

Lidsky, D. and Kwon, R. (1997). *Your Complete Guide to Searching the Net*. In: PC Magazine, Dec.1997. pp.227-258.

Nielsen, J. (1997). *Search and You May Find*. The Alertbox, July 15, 1997 (http://www.useit.com/alertbox/).

Ogden, W. (1988). Using Natural Language Interfaces. In: Helander, M. (ed.): Handbook of Human-Computer Interaction. Elsevier Science Publishers B.V., North-Holland, Amsterdam. pp.281-299.

Ojala, MaryDee (1997). *The Dollar $ign - Where did they hide my company?* In: ONLINE, Dec.1997. pp34-38.

Rosenfeld, L. and Morville, P. (1998). *Information Architecture for the World Wide Web*. O'Reilly, Sebastopol, CA.

Shneiderman, B. (1978). *Improving the human factors aspect of database interactions*. ACM Transactions on Database Systems, 3, 417-439.

Schmidt-Wesche, B., Mack, R., Lenz Cesar, C., and VanEsselstyn, D. (1997). *IBM Search UI Prototype Evaluation at the Interactive Track of TREC6*. In: Voorhees, E. and Harman, D. (eds.): Proc. 6th Text REtrieval Conference (TREC-6), Gaithersburg, MD, Nov.19-21, 1997. To appear.

Tudor, Jan Davis (1997). *Using Droids & Agents - to treat information overload*. In: ONLINE Dec.1997. pp.50-58.

Turtle, H. (1994). *Natural Language vs. Boolean Query Evaluation: A Comparison of Retrieval Performance*. Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Dublin, Ireland. pp. 212-220.

Wacholder, N., Ravin, Y., and Choi, M. (1997). *Disambiguation of Names in Text*. In: Proc. of the 5th Conf. on Applied Natural Language Processing. Washington, D.C., 1997. pp.202-208.

Witulski, F. (1997). *Using Search Engines Effectively*. In: PC Novice - Guide to Going Online. Vol.6, Issue 1. pp.89-94.

Z39.50 - Standard for Information Retrieval (1998 / 1988). Detailed information can be found at: http://lcweb.loc.gov/z3950/.

ZPRISE (1998). Information can be found at: http://www-nlpir.nist.gov/~over/zp2/.