Intelligent Search Assistance

(Journal of the American Society of Information Scientists, Vol. 43, No. 2, March 1992, 175-182)

Susan Gauch

Department of Electrical Engineering and Computer Science

University of Kansas

sgauch@eecs.ukans.edu

ABSTRACT

Unfamiliarity with search tactics creates difficulties for many users of online retrieval systems. User observations indicate that even experienced searchers use vocabulary incorrectly and rarely reformulate their queries. Distributed sources of online information, for example the World Wide Web (WWW), require new types of search techniques. To address these problems, intelligent search assistants have been developed. These systems may assist users with the mechanics of the search, query formulation, query reformulation or all of the above.

1 DRIVING PROBLEM

Technology to produce, store, and distribute massive quantities of electronic information has matured. The Information Highway is becoming a reality. The increase in access to the Internet by the public at large, combined with the development of easy to use graphical browsing interfaces, for example, Mosaic and Netscape, have lead to an explosion in the information being added. In particular, the World Wide Web (WWW) is being used to present an exponentially growing amount and range of information through which people can browse. However, to make effective use of the information accessible from their desks, technology must be developed which allows end-users to search effectively. That is the goal of intelligent search agents, whether they search a single database of bibliographic records or a network of distributed, heterogenous, hypertext documents.

In the early days of online information retrieval systems, individuals met with search intermediaries who were trained to use the online systems and often knowledgeable about the information seeker's area of interest. Through an interview process, the search intermediary would determine the individual's information need, perform the actual searches, and send the results to the information seeker. While this approach works well, and expert searchers can achieve quality search results, a combination of factors converged in the 1980s to encourage the development of online intelligent search assistants to allow individuals to perform their own searches. Technology, in the form of personal computers and networks, provided many people with the means to access the online databases from their own offices. No longer did they need to walk to the library to meet with someone who had a unique ability to connect to online information.

Technology also impacted the amount and type of information being stored. A few dozen bibliographic databases have mushroomed into several thousand databases of everything from full-text documents to movie clips. To give a feel for the magnitude of the growth, the Lycos system [December, 1994], indexes over 860,000 documents form 34,000 different sources and is able to add 5,000 documents per day. The increase in information available, along with the increased number of users connected to the Internet, has lead to increased demand for information. For example, the WWW Worm search system currently has over 2,000,000 search requests per month.

Studies of user behavior have found that merely giving end-users the ability to search is not enough. One user study found that whereas system mechanics are rarely a problem for any but very inexperienced and infrequent users, even experienced searchers have significant problems with search strategy and output performance (Borgman, 1986a). Another found experienced searchers lost sight of the search logic, missed obvious synonyms, and searched too simply (Fenichel, 1981). In spite of low recall, half of the searchers never modified the original query in an attempt to improve their results. Studies of inexperienced searchers find even more problems with search strategy. In one study, a quarter of the subjects were unable to pass a benchmark test of minimum searching skill (Borgman, 1986b). In an experiment contrasting the searching of novices versus experienced searchers, the novices found some relevant documents easily, but they failed to achieve high recall and were unable to reformulate queries well (Oldroyd, 1984). The experienced searchers in this study were more persistent and willing to experiment than the novices, enabling them to achieve better search results.

Blair and Maron (Blair & Maron, 1985) paint a bleak picture for searching full-text databases. Legal assistants searching a legal database achieved only 20% recall, although they were attempting to do a high recall search. The factors, as identified by the authors, leading to this poor performance were poor searching technique (failure to use stemming and synonyms), stopping the query iteration too soon, and the inability to search on inter-document relationships. The authors argued that vocabulary problems make high recall impossible on full-text databases, though this has been disputed (Salton, 1986).

Advanced retrieval systems to improve search performance and/or enable novice users to search more effectively fall into three main categories: 1) intelligent front-ends to traditional information retrieval systems (Section 2); 2) question-answering systems which are based on knowledge representations of the contents of the documents (Section 3); 3) intelligent search agents which search over networks for information on behalf of users (Section 4).

3 INTELLIGENT FRONT-ENDS

Intelligent front ends to traditional information retrieval systems have been built. Those tightly linked to a specific domain tend to emphasize helping users formulate a good query whereas others attempt to assist by modelling the user or the search process in general.

3.1 Query Formation Assistance

3.1.1 Overview

Shoval (1985) developed an expert system to assist users in selecting the right vocabulary terms for a database search. The knowledge base of words, concepts, and phrases and their semantic relationships is stored in a semantic network. Decision rules, based on common search practice, are used to locate candidate vocabulary terms in the semantic network and suggest them to the user for possible query expansion. The user then decides whether or not the candidate terms are relevant and should be used to replace the terms in the nodes which point to it.

IOTA (Chiaramella & Defude, 1987) is an expert system which incorporates a natural language interface. Passages are retrieved from an online book based on keywords which index each passage. Much of the research effort has gone into processing the user's queries, but some simple query reformulation is also done. Specifically, queries are broadened by replacing a term by its parent from an online thesaurus and narrowed by removing OR terms. Their results, based on a small-scale experiment, indicate an increase in precision and recall using the expert system.

RUBRIC (McCune et al, 1985; Tong et al, 1987) provides query formation assistance for full-text databases. Users create rules which describe the domain knowledge for the system as a hierarchy of topics and subtopics. Rules may have weights representing the certainty and/or importance of the defined relationships. The lowest level subtopics define patterns in the text which indicate the presence of that subtopic. Although the query process is very powerful, it places a heavy burden on the user.

3.2.1 In Depth Case Study: CANSEARCH

CANSEARCH (Pollitt, 1984; Pollitt, 1987) is one of the earliest expert systems for bibliographic retrieval. It helps doctors to formulate queries to retrieve cancer-therapy-related documents from teh MEDLINE database. The system is designed for users who are domain experts, but who are novice users of document retrieval systems. The expert system presents the users with a hierarchy of menus, guiding him in the process of query formulation. The user selects from a menu by touching the desired kitem on the screen, elimininating the need for yping. A formal query is formed form the user's selections. The use of menuse results in an easy-to-use system, but CANSEARCH inherits the problems inherient in menu-based interfaces - the user must specify the query top-down, which can be tedious for an experienced searcher. Further exploration in the use of menus for query formulation resulted in GENSEARCH and MenUSE (Pollitt, 1988). Knowledge about the subject domain of clinical cancer therapy, the Medical Sugject Headings vocabulary used to index cancer documents in MEDLINE, the hierarchy of user interface menus, and search statement formulation is represented by rules in PROLOG.

Sample Interaction

The user is first presented with the top-most menu from which they select the broad categories representing their interests (Fig. 1).

INSERT FIGURE 1 HERE

The user thens selects "cancer at a particular site" and "therapy," both of which require further specification. The next menu is used to narrow the cancer site (Fig. 2).

INSERT FIGURE 2 HERE

From this screen, the user selects "specific primary site(s)," causing a menu of cancer sites to appear (Fig. 3).

INSERT FIGURE 3 HERE

The user selects "breast" which the system assumes to be the primary, not secondary, site of interest. The "cancer at a particular site" has now been fully specified and the user makes selections from three further mnues to specify this part of thw uery, selectin "chemotherapy" from a menu of treatment mentods, "antineoplastic antimeabolites" from a menu of drug classes, and finally "fluorouracil" from a menu of antineoplastic antimetabolites. The cancer site and treatment method are now fully specified so the system forms a MEDLINE query to submit to the retrieval system.

3.2 User Modelling

3.2.1 Overview

IR-NLI II (Brajnik, Guida, & Tasso, 1988) incorporates user modeling into a domain-independent bibliographic retrieval expert system. Domain knowledge is supplied separately by an online thesaurus. A user model is built based on the user's amount of domain knowledge and search experience. This model is used to tailor the dialogue between the system and the user. Initially, the user lists some terms which describe his interests. The expert system, through a lengthy dialogue, clarifies its model of the query, proposes terms to expand the query, and comments on the user's search strategy. No automatic query reformulation is done.

3.2.2 In Depth Case Study: I^{3}R

I^{3}R (Croft & Thompson, 1987; Thompson, 1988), Intelligent Interface for Information Retrieval, provides a highly flexible interface that allows a user to use a varitey of means to find informatin in a database on computer science. The system incorporates user modeling in the form of stereoptyes and relevance feedback. The stereotype used foar a particular session is based on questions answered by the user about the current session and the stereotypes established in prefious sessions. The

Stereotype determines the number of documents to be retrieved, the estimate of the number of searches necessary, the amount of information displayed, and whether or not the system should automatically trigger a new search based on teh relevance feedback.

The query formation process is a dialogue between the user and th system during which the user supplies a short natural language quere, an initial document, or a Boolean query. The user identifies the initial search terms by hilghlighting them on the screen. Related concepts are found in the domain knowledge base and presented to the user for confirmatin. If no related concepts are found, the user may be asked to provide additional keywords. The search results are presented as a ranked list of documents from which the user indicates any new keywords of interest.

The system organisation is based on the blackboard model (Nii, 1986a, 1986b) which consists of a number of independently operated knowledge sources that work cooperatively to solve a common problem. All experts have access to the knowledge base of user records andsubject domain knowledge. The user records stor informatin about previous user sessions, including any suser-supplied domain knowledge. The system-supplied domain knowledge is contained in a semantic network/thesaurs containing the folowing relationships: synonym, related, broader, narrower, instance, phrase, and nearest neighbor.

The following example shows the system interacting with a user with a Novice sterotyp who desires a high-precision search. These factors set the size of the retrieval set at twenty, the expected number of relevant documents at five, and the predicted number of searches at two. The user chooses to enter an intitial textual query and highlights the phrases "fault toleraance" and "message passing." He also uses a phrase window to enter "concurrent processes" and "distributed processes," whic do not appear in the text of the query (see Fig. 4). Based on the user's query, the system displays related phrases from the thesaurus ("university programs," "utility programs," "analysis of programs," "programming techniques," and "efficiency of algorithms"). The user may view the thesauras entry for a phrase or mark them as relevant. In this case, the user marks "analysis of programs" and "programming techniques" as relevant phrases.

INSERT FIGURE 4 HERE

In response to the query, titles of the five highest-ranking documents are displayed to the user (see Fig. 5). The bar graph to the left of the title indicates the document's relevance weight relative to the relevance witght of the first document. The user may d isplay the document abstract by selecting the *Show* optionor mark the document as relevant by selecting the *Rel* option. The remaining fifteen document may be viewed by scrolilng the screen. In this example, the user selectes documents one, two, three, and seventeen as relevant.

INSERT FIGURE 5 HERE

While viewing document three, the user selects "synchronizing" and adds the phrase "concurrent reading" (see Fig. 6). Other words and phrases are selected from the other relevant docuemnts. Since only four ot the twenty documents were relevant, a new search is automatically generated incorporating the new search terms. The new search finds two additional relvant docuemnts for a total of six. Thsi fulfills the expectation that a novice searcher should find five documents in two searches so the search ends.

INSERT FIGURE 6 HERE

3.3 Search Process Assistance

3.3.1 Overview

Search strategies employed by both novice and experienced searchers have been widely studied. Bates (1979) has compiled a thorough catalogue of search tactics. She outlines 29 search tactics in four areas: monitoring, file structure, search formulation, and term manipulation. The tactics for search formulation and term manipulation describe the available techniques to broaden and narrow queries. The search formulation tactics include the selection of appropriate initial search terms and the manipulation of query structure; the term manipulation tactics describe the use of context, thesaural terms, and stemming to modify queries. The tactics she lists provide the basic operations for our expert system; however, she includes no guideline as to when each tactic is appropriate.

In a related study based on observation of 47 professional online searchers, Fidel has developed a formal decision tree that represents the intuitive rules searchers use when they select search terms. The options available to the searchers are enumerated, as are the conditions under which each option is selected. The author defines rules for deciding when to use textwords or descriptors or both to search indexed databases, and guidelines for including thesaural relationships. In contrast to the tactics proposed by (Smith et al, 1989), the emphasis is on identifying techniques which are domain independent.

PLEXUS (Vickery & Brooks, 1987) is an expert system to help novice users find information about gardening. The initial query formation consists of a dialogue with the user. Natural language queries are accepted and information is extracted to fill in frames. The system has a knowledge base of search strategies and term classifications similar to a thesaurus. Most of the domain knowledge is in the classification, but some appears in the rule base, itself. If queries are too broad (defined as more than 10 references) , no narrowing is attempted. The references are displayed 5 at a time to the user. If the query is too narrow (defined as nothing retrieved at all), three strategies are attempted: 1) if two or more terms appear in the same subcategory, OR them together rather than AND; 2) drop one of the terms; 3) replace a term by its parent.

3.3.2 In Depth Case Study (Domain Dependent): EP-X

EP-X (Krawczac, Smith, & Shute, 1987; Smith et al, 1989), Environmental Pollution eXpert, is a prototype knowledge-based system that assists users in conducting bibliogaphic searches of the environmental pollution literature. The main emphasis of this system is the use of domain knowledge to provide expert assistance with query formation and reformulation. This approach is based on the author's observation of the performance of a number of expert search intermediaries and an empirical study of 17 end-user searchers. The authors classify the query refomulation techniques used by the intermediary, concluding that the intermediary: (1) makes extensive use of domain knowledge to suggest topic refinements; (2) generates most knowledge-based suggestions spontaneously rather than responding to a cue from the retrieved documents; and (3) very rarely changes logical operators.

Based on the above results, EP-X was designed with a rich comain knwoledge base, represented using frames (Minsky, 1975). Each frame describes a topic in the field of environmental chemistry. For example, there is a frame for "Removal of a *pollutant* in *polluted medium* using a *removal/treatment process*." Each italicized phrase in the topic represnts a slot in the frame that can be filled by a specific instance. One instantiation of the frame would be "Removal of *mercury* in *wastewater* using *ion exchange*." There is also a hierarchy of semantic primitives which represent concepts in the domain and a mapping of words to semantic primitives.

The following scenario shows EP-X's use of domain knowledge to guide the user during query formation, including suggested query reformulation techniques. In addition to the capabilities shown here, EP-X is able to use its domain knowledge to resolve ambiguous keywords and idnetify keywords with overlapping meanings.

Initially, the user enters a query consisting of a list of keywords. Once they have indicated that the list is complete (see Fig. 7), a search is performed against the Chemical Abstracts database.

INSERT FIGURE 7 HERE

The system's interpretation of the query, along with the results of the search, are presented to the user (see Fig. 8). Note that EP-X has inferred that NATURAL WATERS is a member of the general class *polluted media*; RADON-222, CESIUM-137, and STRONTIUM-90 are all members of the general class *chemical pollutants*; and the relationship between *polluted media* and *chemical pollutants* is that pollutants occur in polluted media.

INSERT FIGURE 8 HERE

Not satisfied with retrieving 14 documents, the user chooses to broaden the query. In response, EP-X determines that the chemicals mentioned are all very specific concepts that fall under the general term RADIOACTIVE SUBSTANCES. Therefore, it suggests broadening the query by replacing the specific radioactive substances with the more general term and informs the user, *a priori*, what the effect on the retrieval set size will be (in this case, 73 additional documents would be retrieved). The user decides to accept the suggestion which results in 87 documents being retrieved. Next, the user decides to narrow the query. EP-X asks which concept to narrow: chemical pollutants (radioactive substances) or polluted media (natural waters). The user chooses the latter concept and EP-X shows the user the hierarchy of semantic prmitives for the concept which the user can prune to make the search more specific (see Fig. 9). In thes example, the user decides to display the current search results rather than narrow by removing any particular type of body of water.

INSERT FIGURE 9 HERE

EP-X was evaluated using data from 32 subjects. Although subjects were able to search successfully with the system, their search results were mediocre. Although EP-X provides users with the ability to broaden or narrow their queries, users did not take advantage of the refinement suggestions even when they would have helped. This reluctance of users to reformulate queries has been encountered in other studies (Fenichel, 1981; Oldroyd, 1984), indicating that perhaps the query refinement should be automatic.

3.3.3 In Depth Case Study (Domain Independent): MICROARRAS

MICROARRAS (Gauch & Smith, 1992, 1993) incorporates an expert system front-end designed to augment the searching capabilities of novice searchers by providing automatic query reformulation. Users enter an initial Boolean query and a desired number of paragraphs which is used to guide the search of an online textbook of computer architecture (Brooks, 1987). The query is broadened or narrowed as necessary until the number of passages retrieved is within 20% of the target number. The retrieved paragraphs are presented to the user in rank order. In an empirical stucy, it was shown that (1) the expert system significatnly reduced the number of queries necessary to find relevant passages; (2) the expert system produced marginally significant improvements in precision; and (3) overall, relevant passages were ranked above irrelevant ones.

Each search term is interpreted to represent a unique concept. Subject domain knowledge is encapsulated in a hierarchical thesaurus which maps between search terms and concepts, storing, for each concept, stem and synonym relationships. In addition, the inter-concept relationships of broader and narrower are represented. The knowledge of search tactics is represented in a production rule base. The main query reformulation techniques used are: adding related terms to a concept, adjusting the context required for co-occurrence, and manipulating the Boolean operators. The ranking function is based on the relative frequency of occurrence of the search terms in the paragraph, the source of the query term (user-supplied or extracted from the thesaurus), and the number of search terms for each concept. The system may run with or without user confirmation of term selections from the thesaurus.

In a sample scenario, the user enters a Boolean query and a target number of 15. Consider a user looking for passages about the alignment of words within computer memory.

User: boundary AND word ANDNOT page

System: That query retrieved 1 passage.

Initially, the AND operator context is "same sentence" which requires that concepts ANDed together must occur within the same sentence. Similarly, ANDNOT requires that the excluded concept must not occur within the same sentence as the desired concept. Since only 1 passage is retrieved, the expert system would use its first broadening technique: expanding individual search terms. Each search term is interpreted to represent a unique concept, identified by the online thesaurus. The intra-concept relationship of *same stem* is used to broaden boundary and word to the sets of all words sharing the same stem (indicated by the use of initial upper-case letters). Note that *page* was not expanded. Since it is a concept to be excluded, broadening it would actually narrow the query as a whole.

User: Boundary AND Word ANDNOT page

System: That query retrieved 4 passages.

The next broadening technique uses the intra-concept relationship of *synonym* to expand the *Boundary* and *Word* concepts, resulting in the retrieval of 7 passages. Note that there was no synonym for *Word* in the thesaurus.

User: (Boundary or Limit) AND Word ANDNOT page

System: That query retrieved 7 passages.

Having exhausted the intra-concept relationships, the system loosens the context around the AND and ANDNOT

2 SYSTEM ARCHITECTURE

The prototype system consists of five major components (see Figure 1):

1) MICROARRAS (Smith, Weiss, & Ferguson, 1987), which serves as the full-text search and retrieval engine

2) a full-text database of over 188,000 words

3) a hierarchical thesaurus of approximately 7,424 words specific to the textbase's domain

4) an expert system of 85 OPS83 rules and over 5,000 lines of C code, which interprets the user's queries, controls the search process, analyzes the retrieved text, and ranks the search results

5) a user interface, which accepts the user's queries, presents requests for information from the expert system, and displays the search results.

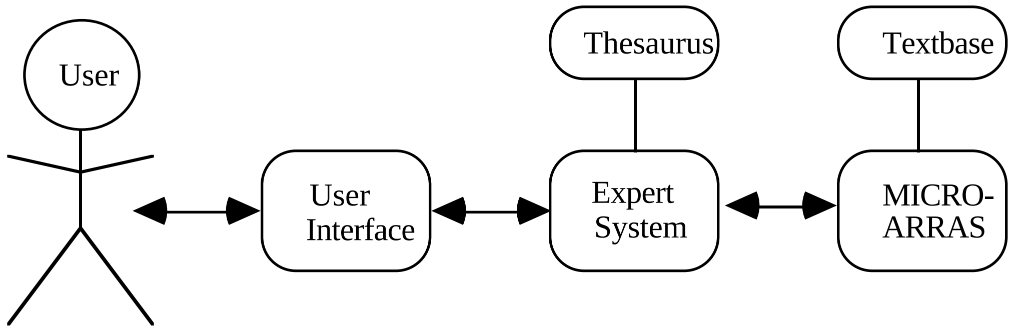

Figure 1. System Architecture

The system is implemented on a Sun 3 workstation. MICROARRAS and the thesaurus construction and access routines are written in the C language. The expert system consists of a knowledge base of production rules, written in OPS83, and a set of C language functions to carry out the actions prescribed by the rule-base. The textual database for the current demonstration project consists of an unpublished manuscript on computer architecture written by Gerrit A. Blaauw and Frederick P. Brooks, Jr. (1986). The search process consists of a dialogue between the user and the expert system. The user enters the initial Boolean query and the number of passages (i.e. paragraphs) he would like to retrieve. The expert system parses the query and translates it into a request for information from MICROARRAS. MICROARRAS retrieves text passages from the full-text database and informs the expert system of the number of passages that satisfy the request. The expert system compares the number retrieved with the target number to decide whether or not to reformulate the query, and, if so, how. Once the target number has been reached, or the expert system has run out of reformulations to try, the retrieved passages are presented to the user in rank-order.

A major advantage of this architecture is the separation of strategic knowledge, contained in the knowledge base for the expert system, from domain knowledge, contained in the thesaurus. Now that the search strategy rules have been developed and tested with the existing textbase, the expert system can be tested with other content domains by simply providing a suitable thesaurus for the new textbase.

2.1 MICROARRAS

MICROARRAS is a full-text retrieval and analysis system. The system provides immediate access to any passage in the textbase, regardless of the length of that document. Contexts for searches can be indicated in terms of words, sentences, paragraphs, etc., for the entire search expression or for different parts of it. To be inserted into MICROARRAS' textbase, documents must first be inverted (i.e. a dictionary is created with an entry for each word in the text. Each entry contains the word and the numerical position in the text of each occurrence of that word). However, they require no semantic preprocessing. Once stored in the textbase, they can be examined individually or in groups. They can also be moved from one textbase to another. Thus, documents can be processed on a workstation or microcomputer, uploaded into a textbase on a mainframe or textbase server, searched and analyzed there, or downloaded for local use once again.

2.2 Textbase

The textbase contains the Fall, 1986 draft of *Computer Architecture, Volume 1 - Design Decisions* by Blaauw and Brooks. The manuscript consists of 188,278 words comprising 8 chapters, titled: "Introduction", "Machine Language", "Addresses", "Data", "Operations", "Instruction Sequence", "Supervision", and "Input/Output". TeX format marks were already present and were used to display of the retrieved text (line, italics, label), as well as provide structural information (chapter, section, subsection, subsubsection, paragraph, sentence, item).

2.3 Thesaurus

All domain-specific knowledge is contained in a hierarchical thesaurus. The expert system uses this information to reformulate queries. The thesaurus was built by the author from the Brooks and Blaauw text, and it strongly reflects the word usage of that textbase. In general, it should not be necessary to provide a unique thesaurus for each textbase. An existing thesaurus for the domain could be used, as long as there is a good match between thesaurus classes and textbase word usage.

Word types which share a common stem are grouped into *stemgroups*. The members of a given stemgroup are called *stemwords*. Each word type in the Blaauw and Brooks text appears in exactly one stemgroup. Thesaurus classes contain stemgroups which are synonyms for each other. Stemgroups may appear in zero, one, or more than one thesaurus class. Because the thesaurus classes are linked together with parent-child links, they are also referred to as *nodes*. The arrangement of the words into stemgroups, stemgroups to thesaurus classes, and the classes into a hierarchy is discussed in (Gauch, 1991). To capture the relationships between thesaurus classes for use by the expert system and the user, the high-frequency terms are included in the thesaurus.

2.4 Query Language

A Boolean query language was chosen because it is the most common type available on existing systems. The operators provided, in decreasing order of operator precedence, are: ANDNOT, AND, and OR. A default context of one sentence is used for the AND and ANDNOT operators. When a query is parsed, the expert system interprets each search term to represent a unique concept. The concepts, and the operators, are flagged as positive or negative based on whether they are specifying information the user does, or does not, wish to receive. For example, the query 'i/o

ANDNOT (device OR interrupt)' contains three concepts: i/*o*, *device*, and *interrupt*. I/*O* is a concept on which the user wishes information, so it is considered a positive concept. *Device* and *interrupt* indicate concepts on which the user does not wish information, so they are considered negative concepts. The ANDNOT and OR operators are followed by negative concepts, so they too are flagged as negative.

When the user is searching with the expert system, the expert system controls the context. Initially, the default of one sentence is used, but the expert system may adjust the context during query reformulation. However, when the user is searching without the expert system, the AND and ANDNOT operators may be augmented with a user-specified context. The user may define the search context for AND or ANDNOT in terms of words, sentences, and / or paragraphs.

2.5 Knowledge Base

The expert system performs three main functions:

1) it controls the operation of the system as a whole;

2) it reformulates the Boolean query based on previous search results;

3) it ranks the retrieved passages in decreasing order of estimated relevance for presentation to the user.

To perform these functions the expert system contains a knowledge base of the search process, search strategies, and passage ranking procedures. Domain knowledge is contained in the hierarchically structured thesaurus. The system has no knowledge of the user's true information needs, other than the target number they specify to indicate how many passages they wish to retrieve.

2.5.1 Query Reformulation

Queries are reformulated based on the target number, the number of passages retrieved, and the history of broadening and narrowing techniques already applied. The expert system has a collection of reformulation tactics at its disposal. Bates (1979) and others have identified successful search tactics and Fidel (1991) discusses when to use free-text terms versus descriptors. However, no one has outlined an overall query reformulation strategy for free-text searching. The guiding principles for the expert system's query reformulation knowledge base were: 1) each search term in the initial query represents one concept on which the user does, or explicitly does not, want information; 2) the user's initial search terms are the best indication of the user's areas of interest; 3) some terms from the thesaurus may be helpful, but others will not; 4) the expert system should never discard concepts in which the user has indicated an interest.

<u>Query Reformulation Techniques</u>

The expert system reformulates queries using three different techniques: 1) expanding concepts; 2) adjusting context; and 3) changing the query structure.

Expanding Concepts

To broaden a query, search terms are added to the positive concepts, whereas narrowing a query adds search terms to negative concepts. Concepts may be expanded by stemming, adding synonyms, and adding related search terms from the thesaurus. The order in which the terms are added from the thesaurus is: parents, then siblings, then children. Replacing a term with its parent to broaden a query is a common practice, both by searchers (Bates, 1979; Salton, 1986), and in systems which automatically reformulate queries (Chiaramella & Defude, 1987; Vickery & Brooks, 1987). The rationale, based on experiences with keyworded bibliographic systems, is that since parent terms represent broader concepts, adding the parent term should broaden the scope of the query. In full-text databases, we believe that the reverse order may make more sense. Broadening a concept containing *virtual memory* with children terms, yielding 'virtual_memory OR paging OR segmentation', seems more likely to retrieve relevant passages than broadening with the parent terms, yielding 'virtual_memory OR memory'.

Crouch (1988) found that augmenting a query with thesaurus terms, rather than replacing the original search terms, lead to improved results. With this in mind, concepts are expanded by adding thesaural terms (ORing them with the terms already in the concept) rather than by replacing the terms already present.

The belief that some stemgroups from the thesaurus will be useful, while other will not, is the basis for providing user filtering of the candidate thesaurus terms. One expert system uses domain-dependent search strategies to choose the appropriate terms from a thesaurus (Smith et al, 1989). In addition, Harman (1988) showed that search results improved when thesaural terms were filtered by the user. Based on these two studies, we decided to allow the users to select which stemgroups to add from a set of thesaural candidates.

Finally, candidate search terms selected from the thesaurus are filtered to remove those which already occur in the query and extremely high frequency terms. The remaining terms are added one at a time, in reverse order of frequency, and the new number of retrieved passages is compared to the target number.

Adjusting Context

The expert system manipulates four different contexts; it adjusts the distance between words in positive and negative multiword phrases as well as the distance between positive and negative search concepts. The expert system broadens queries by increasing the positive contexts and decreasing the negative ones. Narrowing is done by decreasing the positive contexts and increasing the negative ones.

Changing Query Structure

The final variable the expert system can manipulate to reformulate the query structure. The query can be broadened in two different ways: first, the positive AND operators can be switched to OR operators (and the negative OR operators switched to ANDs); second, the negative parts of the query can be dropped altogether. All of the AND operators are replaced at the same time. A better

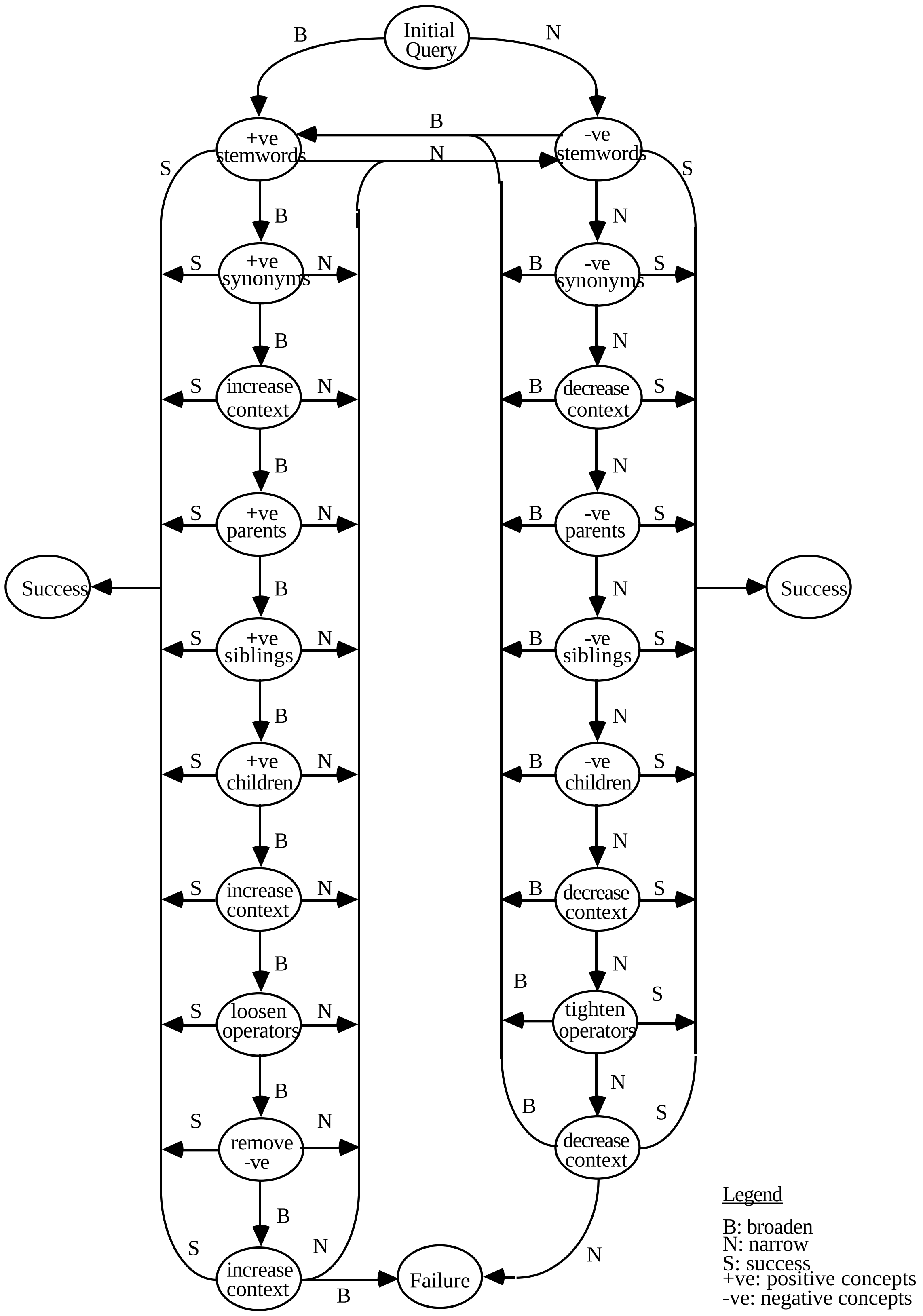
Initial Query
B
N
+ve stemwords
-ve stemwords
+ve synonyms
-ve synonyms
increase context
decrease context
+ve parents
-ve parents
+ve siblings
-ve siblings
+ve children
-ve children
increase context
decrease context
loosen operators
tighten operators
remove -ve
increase context
decrease context
Success
Success
Failure
S
Legend
B: broaden
N: narrow
S: success
+ve: positive concepts
-ve: negative concepts

Figure 2. Query Reformulation Techniques

strategy would be to replace them one at a time, in inverse order of the frequency of occurrence of the concepts. Similarly, the query can be narrowed by replacing OR operators with ANDs. The expert system does not have enough information about the user's information needs to decide which positive parts of the query to drop, so this technique is not employed to narrow queries. Because manipulating query structure causes major changes to the user's original query, these techniques are only tried as a last resort. It is not likely that the new query will find passages that the user will find highly relevant, but the goal is to find somewhat relevant passages that users can read in order to reformulate their own queries and try again.

Flow of Control

Figure 2 diagrams the flow of control among the reformulation techniques. The left side of the Figure 2 diagrams the broadening techniques, the right side the narrowing techniques. This figure

is somewhat simplified since it does not show the use of context to converge to the target number once queries have been found which bracket the target number from above and below. The expert system records the type of initial query reformulation as the global objective. If the reformulations in the original direction overshoot the target number without achieving success, reformulations in the opposite, or local, direction are tried, beginning at the top node on that side of the diagram. Reformulation never continues in the local direction farther than it reached in the global direction. At this point, queries have already been formed which bracket the target number from below and above, otherwise the system would not have tried both narrowing and broadening techniques. Rather than using techniques which are considered less likely to produce good results, the expert system adjusts the context.

Stopping

Knowing when to stop a search is a difficult problem. We partially side-step this problem by having the user explicitly state the number of passages he wishes to retrieve. Since the target number he supplies is likely to be a rough guess, a range of 20% is considered successful. A larger range may be desirable, but since the user is able to stop the reformulation process himself, the size of the range is not important. Left on its own, the expert system stops the reformulation process when it achieves success, or it has run out of techniques to try.

Sample Scenario

The following sample scenario illustrates how the reformulation rules are applied. Since our current textbase concerns the domain of computer architecture, the example describes the interactions of the system with a user searching for information on the alignment of word boundaries in memory.

The user might enter a query 'boundary AND word ANDNOT page', which indicates that he wishes to retrieve passages containing information on word boundaries but not page boundaries. Assume a target number of 15. Applied to this textbase, the original query would retrieve only one passage, so the expert system would attempt to broaden the query. The first step would be to replace the word types *boundary* and *word* with their stemgroups. The resulting query would be 'Boundary AND Word ANDNOT page', where the capitalized search terms indicate the whole stemgroup is included. Notice that *page* has not been expanded to its stemgroup, as it is a negative, or excluded, concept. Four passages would now be retrieved.

The next step would be to broaden the query by including synonym stemgroups for each of the positive search terms, in turn. From the thesaurus it is found that Boundary has one synonym, Limit, however there is no synonym for Word. The query now becomes '(Boundary OR Limit) AND Word ANDNOT page', which retrieves seven passages. Relaxing the context around the AND operator to adjacent sentences while decreasing the context around the ANDNOT operator to within 5 words increases the number of passages retrieved to nine. To further broaden the query, the parent stemgroups for the positive concepts are added. Block and Segment are added to the concept Boundary. The Word concept remains unchanged, since Word has no parent in the thesaurus. The query becomes '(Boundary OR Limit OR Block OR Segment) AND Word ANDNOT page', which retrieves twelve passages. Twelve is within 20% of the fifteen passages requested, so the reformulation stops. If the user requests to see the retrieved passages, the expert system would rank the retrieved passages and present them to the user in decreasing rank-order.

2.5.2 Passage Ranking Rules

The dialogue between the expert system and MICROARRAS normally produces a set of passages to be displayed to the user. The last task performed by the expert system is to rank order those passages in terms of their probable interest to the user. To do this, it performs an elementary content analysis on each passage and computes a weight representing probable interest.

Ranking algorithms for document retrieval systems have been extensively studied (Harman, 1986). There has been less work done on ranking for passage retrieval systems. The FAIR system (Chang & Chow, 1988) performs a simple ranking based on the distance between word pairs, the number of search terms represented and the number of occurrences of the terms. Ro (1988) did an extensive comparison of different ranking algorithms for full-text, but failed to demonstrate significant differences in performance. Al-Hawamdeh (1991) ranks full-text paragraphs based on their similarity to a query expressed as an unstructured list of keywords, finding that the nearest-neighbor based searching is as effective as Boolean searching.

The ranking algorithm used by our expert system considers the following factors: the number of different concepts represented in the passage; the number of different word types for each concept; the relationship of the concept's word types to the user's original search terms; the number of occurrences for each word type from the search expression appearing in the passage; and the contextual distance between search terms. The passages are then ranked according to their respective index values and presented to the user in order of decreasing rank.

Calculating Passage Weights

The weight W_{pq} of passage p for query q, $0 <= W_{pq} <= 1$, is a function of the weight C_{ip} of each query concept i in p, the relationship between the concepts (determined by the parse tree), and the contextual closeness between the concepts. The concept weights are combined by applying the rules for fuzzy logic (Zadeh, 1965) to the Boolean structure of the query. Additionally, a closeness factor is associated with each of the AND and ANDNOT operators. The closeness factor for the AND operator is set to one of three values (1.0 for same sentence, 0.9 for adjacent sentences, 0.8 for same paragraph). The closer two positive concepts appear in the passage, the higher weight that passage receives. Complementary closeness values are used for the ANDNOT operator (0.8 for same sentence, 0.9 for adjacent sentences, 1.0 for same paragraph).

$$W_{p(Ci \text{ AND } Cj)} = \min(C_{ip}, C_{jp}) * \text{PositiveCloseness} \qquad (1)$$

$$W_{p(Ci \text{ OR } Cj)} = \max(C_{ip}, C_{jp}) \qquad (2)$$

$$W_{p(\text{NOT } Cj)} = (1 - C_{jp}) \qquad (3)$$

From (1) and (3)

$$W_{p(Ci \text{ ANDNOT } Cj)} = \min(C_{ip}, 1 - C_{jp}) * \text{NegativeCloseness} \qquad (4)$$

The concept weights and closeness factors fall in the range [0,1], therefore the passage weights also fall in the range [0,1].

Calculating Concept Weights

The weight of concept i in passage p, Cip, is a function of the weight of each concept term T in query q, denoted T_{jq} for search term j, and the weight of each concept term in the passage, denoted T_{jp} for search term j, and the number of search terms for the concept. The weight of a search term in the passage is multiplied by the weight of that search term in the query. Thus, the highest weight search terms are those which are important in the query as well as the passage. The weights for all the concept's search terms are summed together and normalized by the number of search terms for the concept, N.

$$C_{ip} = 1/N \sum_{j=1}^{N} T_{jq} * T_{jp} \quad \text{where term j is in concept i} \qquad (5)$$

The term weights fall in the range [0,1], therefore, the concept weights also fall in the range [0,1].

Calculating Term Weights

Two different term weights, T, are calculated: the weight of the search term i in query q, T_{iq}, and the weight of the search term i in passage p, T_{ip}.

Query Term Weights

The weight of the search term i in query q, T_{iq}, reflects the relationship of the search term to the user's original term. The relationships, from closest to most remote, are: same word, stemgroup, synonym, parent, sibling, child. These distances reflect the order in which search terms are added to the concepts, which in turn reflects confidence in the closeness of the relation of the search term to the original term.

$$T_{iq} = 1.0 \text{ (word)}, 0.9 \text{ (stemgroup)}, 0.8 \text{ (synonym)}, 0.6 \text{ (parent)}, 0.5 \text{ (sibling)}, 0.4 \text{ (child)} \qquad (6)$$

The query term weights fall in the range [0,1] as required, with the original word receiving a weight of 1.0. Terms added by the expert system receive weights which decrease by 0.1 for every step away from the original term, except for the step from synonym to parent terms. This step decreases the term weight by 0.2, reflecting the large decrease in confidence which occurs when terms are added from outside the thesaurus class.

Passage Term Weights

The weight of the search term i in passage p, T_{ip}, reflects the frequency of the search term in the passage, f_{ip}, and the frequency of the search term in the textbase, f_{it}. An evaluation several full-text ranking algorithms and concluded that those based on relative document frequency provided the acceptable performance (Ro, 1988). Since this is also simple to calculate, we chose relative frequency for the term passage weights.

$$T_{ip} = f_{ip} / f_{it} \tag{7}$$

The term passage weights fall in the range [0,1], as required.

3 EVALUATION

Our primary goal is to demonstrate that using an expert system to reformulate queries can improve search performance for novice searchers. To evaluate the system, users queried the textbase using three interfaces with different capabilities: an interface whose only function was to accept contextual Boolean queries and display search results; a similar interface which also allowed the user to explore the online thesaurus; and a third which incorporated the searching expert system. Each subject's search performance with the three interfaces was monitored and compared.

3.1 Hypotheses

Hypothesis 1: The expert system improves the search effectiveness for a novice searcher.

Hypothesis 2: The expert system improves the search efficiency for a novice searcher.

Hypothesis 3: The expert system can rank the passages retrieved by the search in decreasing order of relevance.

The effectiveness of the retrieval output is evaluated by looking at recall (the number of relevant items found / the total number of relevant items in the database) and precision (the number of relevant items retrieved / the number of items retrieved). Two estimates of the number of relevant items retrieved are examined: the number of passages the users mark as relevant and the number of passages retrieved from the set of passages deemed relevant by the author. The efficiency of the systems is measured by the number of Boolean queries the subjects entered for each of several high-level questions, and by the amount of time they spent searching for relevant passages for each question. The ranking algorithm was evaluated by comparing the order of appearance of relevant passages after they have been ranked with a random order of appearance.

3.2 Method

3.2.1 Subjects

Twelve computer science graduate students participated as subjects in the study. All subjects were knowledgeable in the use of computers, but unfamiliar with online searching. Thus, they were representative of the anticipated users of future information retrieval systems.

3.2.2 Apparatus

<u>Information Retrieval Systems</u>

The *user-alone* configuration consisted of a Sun 3 running MICROARRAS and a rudimentary expert system. This expert system performed only the system control function, and did no query reformulation or ranking of retrieved passages. The user was prompted for a contextual Boolean query, this query was sent to MICROARRAS, and the number of passages retrieved was reported back to the user. The user could display the passages retrieved, if there were fewer than 25, or try another query. Typing was minimized by using the Sun's windowing package to cut and paste the previous query, edit, and re-run it.

The *user-thesaurus* version consisted of a Sun 3 with one window running MICROARRAS, as in the user-alone system, and a second window running a thesaurus access function. In the thesaurus window the user had access to all the thesaurus information available to the expert system. He could find out the stemname for a specific word's stemgroup. For any stemname, he could ask for the stemnames of the corresponding synonym, parent, sibling, or child stemgroups. These stemnames could be used in the user's query to MICROARRAS. Typing was minimized by using the Sun's windowing package to cut the stemgroup from the thesaurus window and paste it into the appropriate concept of the query.

In the *user-expert system* version the user did not have access to the online thesaurus. Context and the addition of stemgroups were controlled by the expert system. Thus, the user entered a Boolean query and a target number of passages and the expert system reformulated the user's query to attempt to get close to the target number. The user was prompted to filter search terms found in the thesaurus, and to continue or abandon the current reformulation. To keep the response time approximately the same as for the other two configurations it was necessary to run MICROARRAS remotely on the Sun 4 file server containing the textbase. The user worked with one window on a Sun 3 which ran the full version of the query reformulation expert system. The expert system communicated with MICROARRAS over the network. This setup was approximately twice as fast as when MICROARRAS was run on the user's Sun 3. This speed up was necessary, not because the expert system code itself was slow, but rather because the expert system tended to form very long queries involving many MICROARRAS categories, and MICROARRAS slows down linearly with the number of search terms in a query.

Questions

Three sets of five questions were devised. Each set contained one training question and four questions on which the subjects were monitored. The questions covered material ranging over the whole textbase. For the fifteen questions, the number of relevant passages in the textbase ranged from a low of 7 to a high of 23 with a mean of 15.4.

3.2.3 Procedure

Subjects were asked to try to find on the order of ten relevant passages from the textbase in response to the questions they would be given. They were informed that they might not always be able to find that many, and they were allowed to stop working on a query whenever they were satisfied that they had found as much as they could. The target number of ten was chosen because it was large enough to require a high recall search, yet small enough that the users would not become tired reading passages.

Each subject worked with each of the three systems, in turn. This was done to compensate for the large individual differences found in searching ability (Borgman, 1987). To compensate for learning during the experiment, the order of presentation of the three systems was counterbalanced among subjects. The order of presentation of the question sets was the same for all subjects (Set A first, then B, then C). Thus, each question set was searched on each system four times. The subjects received a training session with each system before they began their monitored searches. When they had completed all three sessions, they were asked to fill out the questionnaire stating their preferences and opinions.

3.2.4 Data Collection

Raw Data

Data was collected in a trace file while the subjects worked with the system. Each communication from the subject to the retrieval system, and vice versa, was stored with a time stamp. Thus, timing information was collected along with the history of queries entered by the subject and the search results. When the subject chose to display the retrieved passages, those passages and the subject's relevance judgment of them were also stored.

Definitions

A *unique query* was any error-free query entered by a subject. If a subject entered a query which contained a typographic or logical error, and he indicated that he noticed the error by aborting the search and re-entering a corrected version, then the erroneous query was not considered a unique query. However, if the subject gave no indication that he was aware of the error, but instead moved on to a different query altogether, then the erroneous query was considered unique.

The *relevance weight* of a passage is the relevance number assigned to the passage by the subject. A *very relevant (user)* passage is one assigned a relevance weight of two. A *somewhat relevant (user)* passage has a relevance weight of one. A *relevant passage (user)* is one that is either very relevant or somewhat relevant, as judged by the user. An *irrelevant passage (user)* is a passage given a relevance number of zero.

It is necessary to have an estimate of the total number of relevant passages available for each question, in order to calculate recall. This estimate was calculated by forming the union, for each question, of the set of passages judged very relevant by any subject. Passages in this set judged irrelevant by the author were removed. The remaining passages form the *absolute retrieval set* and are called the *relevant passages*. It was necessary to remove some passages marked very relevant by a subject because, perhaps due to a misinterpretation of the question or a misunderstanding of the passage, some subjects gave a relevance weight of two to irrelevant or marginally relevant passages. This tendency to overestimate the relevance of passages may also be because, in some cases, subjects were unable to find the truly relevant passages, and thought that they had retrieved the best passages available when in fact they had not.

A *successful retrieval set* is a retrieval set containing at least five relevant passages. Since the subjects were attempting to find ten relevant passages, a successful retrieval set contains at least half the number for which they were looking. The textbase contained approximately the same number of relevant passages for each question, allowing the target number and size of the successful retrieval set to be held constant.

The *final retrieval set* was chosen as the last successful retrieval set. If a subject never retrieved a successful retrieval set for a given question, the retrieval set with the highest number of relevant passages, as judged by the subject, was chosen. The *final query* is the query input by the user which resulted in the final retrieval set.

Variables

Total time per question is calculated from the entry of the subject's first query for the question until after the display, or decision not to display, of the final set of retrieved passages.

Number of queries per question is determined by counting the number of unique queries the subject entered for a given question.

Number of relevant passages (user) found per question is determined by counting the number of user indicated relevant passages in the final retrieval set for the question.

User precision is calculated for the final retrieval set using the standard formula of: number of relevant passages (user) retrieved / number of passages retrieved.

Number of relevant passages found per question is determined by counting the number of passages in the final retrieval set for the question that are members of the absolute retrieval set.

Precision is calculated for the final retrieval set using the standard formula of: number of relevant passages retrieved (absolute) / number of passages retrieved.

Recall is calculated for the final retrieval set using the standard formula of: number of relevant passages retrieved (absolute) / total number of relevant passages available.

The *ranking balance point* (R) for each retrieval set (not just the final one) is calculated by

$$\frac{\sum_{i=1}^{n} i * relevance_i}{\sum_{i=1}^{n} relevance_i}$$

where n = number of passages in the retrieval set

i = position of the passage in the retrieval set

$relevance_i$ = relevance weight of passage i

This calculates where the midpoint of the relevant passages lies, accounting for the relevance weight. The earlier in the retrieval set the relevant passages occur, the smaller their midpoint. For example, consider a retrieval set of five passages of which one is very relevant (weight = 2), two are somewhat relevant (weight = 1) and two are irrelevant. An example ranking might present the very relevant passage first, then a somewhat relevant passage, an irrelevant one, the other somewhatr relevant passage and then the final irrelevant passage, represented as (2, 1, 0, 1, 0). Using this formula, the example balance point would be

$$\frac{(1*2) + (2*1) + (3*0) + (4*1) + (5*0)}{2 + 1 + 1 + 0 + 0} = 2$$

Similarly, the worst-case balance point would be 4.25, and the example balance point 2.

The *best case balance point* (BC) for each retrieval set is calculated by applying the ranking balance point formula to the case where all very relevant passages preceded all somewhat relevant passages which in turn preceded all non-relevant passages in the set. In this case, the best case ranking would be (2, 1, 1, 0, 0), yielding a balance point of 1.75. For comparison, the worst case ranking, (0, 0, 1, 1, 2), would yield a balance point of 4.25.

The *midpoint* (M) for each retrieval set is calculated by (n+1)/2 where n is the number of passages in the retrieval set. The midpoint is used for comparison with the ranking balance point. A ranking algorithm which distributed the relevant passages evenly throughout the set, in our example (1, 0, 2, 0, 1), would yield a ranking balance point of 3. This is also the midpoint of the set. A good ranking algorithm would produce distributions with balance points less than the midpoint (i.e. relevant passages presented earlier).

The *normalized ranking balance points* were calculated from the ranking balance points by moving the midpoint to 0 and adjusting the range so that the best case balance point fell on 1, and the worst case balance point at -1. The normalization performed was:

Normalized ranking balance point (NR) = (M - R) / (M - BC).

For the example retrieval set, the normalized ranking balance point would be:

(3 - 2) / (3 - 1.75) = 0.8.

Note that this normalization is not possible when BC equals M, since a division by zero will be attempted. This can arise when all the retrieved passages receive the identical relevance weight (e.g. (2, 2, 2, 2, 2)). Since any order of presentation is as good as any other when the passages are all of equal relevance, it would be safe to ignore these cases. However, this situation never arose in the experiment.

Summaries Calculated for Each System

For each system the means calculated were:

- number of queries per question
- time per question (seconds)
- number of relevant passages (user) per question
- user precision
- number of relevant passages (from absolute retrieval set)
- precision
- recall

For each ranking algorithm (the expert system's, and randomness) the normalized balance points were calculated.

3.3 Results

The means were compared to determine if their differences were statistically significant. Pairwise two-tailed t-tests were performed. A difference was considered significant if its probability of occurring due to chance was less than 5% at the 95% confidence level (a 10% chance at the 95% confidence level was considered marginally significant). Pairs of means with statistically significant differences are flagged with asterisks.

3.3.1 Search Effectiveness

All three systems retrieved comparable numbers of relevant passages. Whereas there seemed to be higher recall with the thesaurus, shown by a mean of 7.688 compared to a mean of 7.292 with the expert system, this difference was not significant (p = 0.5333).

- number of relevant passages (user) per question

user alone	7.375
user and thesaurus	7.688
user and expert system	7.292

All three systems produced comparable precision, based on the subject's relevance judgments.

- user precision

user alone	0.763
user and thesaurus	0.786
user and expert system	0.761

All three systems retrieved approximately the same number of passages from the absolute retrieval set.

- number of passages from absolute retrieval set

user alone	5.521
user and thesaurus	5.708
user and expert system	5.729

Recall was comparable across all three systems. There was a slight improvement in recall for the user and expert system configuration, but the advantage over the user-alone configuration was not significant (p < 0.6988).

- recall

• user alone	0.364		
• user and thesaurus		0.368	
• user and expert system	0.379		

The user and expert system configuration produced marginally significant improvements in precision when compared with the user-alone configuration.

- precision

• user alone	0.530	*	(p < 0.0817)
• user and thesaurus		0.576	
• user and expert system	0.604	*	

3.3.2 Search Efficiency

The user was marginally significantly slower when using a thesaurus. The expert system was not significantly slower than the other two systems. However, MICROARRAS was being executed by a Sun 4 with the user-expert system configuration resulting in approximately a doubling of its speed.

- mean time per question (seconds)

user alone	474.5	*	($p < 0.101$)
user and thesaurus	571.5	*	
user and expert system	539.8		

The expert system improved search efficiency, as measured by number of user queries over both the user alone and user plus thesaurus.

- number of queries per question

user alone	4.833	*	($p < 0.0001$)
user and thesaurus	5.458	**	($p < 0.0001$)
user and expert system	2.354	*,**	

3.3.3 Ranking

The expert system ranked relevant documents more highly than would be predicted by randomness. The expert system's ranking was compared to a random distribution for 74 sets of retrieved passages.

- balance points
    - random 5.00 * ($p < 0.0165$)
    - expert system 4.53 *
- normalized balance points (on range of -1 to +1)
    - random 0.000 * ($p < 0.0025$)
    - expert system 0.195 *

3.4 Analysis

<u>Effectiveness</u>

The first hypothesis, that the expert system can improve the search effectiveness for a novice user was not supported by this study. However, the expert system produced marginally significant improvements in precision, and seemed to indicate improvements in recall. Providing the online thesaurus produced no improvement in search effectiveness. The suggested improvements in precision may result from the expert system applying better broadening techniques. The subjects, when searching unassisted, would often stop with a very broad query and examine a large set of retrieved passages (over fifteen) looking for relevant information. This type of strategy results in the lower precision observed when the subjects search on their own.

The subjects' browsing strategy may account for their ability to produce recall comparable to the expert system when there were a large number of relevant passages in the textbase. For example, in two questions with large absolute retrieval sets the subjects were able to retrieve, on average, 10 and 10.25 relevant passages on their own compared with the expert system's retrieval of 8 and 7.75 passages respectively. By using a target number of 10 for these broader questions, the expert system was operating at a disadvantage. More relevant information was easily found, judging by the high recall of the subjects, but the expert system did not even attempt to further broaden the query. Clearly, 10 is not the ideal target number for all queries.

Efficiency

The second hypothesis, that the expert system can improve the search efficiency of novice searchers, was supported. Using the expert system significantly reduced the number of queries subjects needed to answer a given question. Subjects required fewer than half as many queries per question on average versus systems in which the user queried without it, a substantial improvement. The expert system reduced the amount of user effort required by decreasing the number of queries a user needs to design to express their information needs. If efficiency is measured in terms of total user time the expert system fares less well. The expert system was not significantly slower than either of the other two systems but it was necessary to run MICROARRAS on a faster machine to achieve this. However, this version of the expert system was designed with correctness rather than efficiency in mind, and there are several ways that it could be sped up. In particular, when a stemgroup is added to a concept, the entire query is re-evaluated against the textbase. A large speed improvement could be gained by unioning the passages retrieved by the new stemgroup with those retrieved by the rest of the concept (which has already been calculated).

Allowing the subjects to access the online thesaurus actually decreased the subjects' efficiency. They took significantly more time than when they searched on their own and generated as many queries. This seems to indicate that the improvement in efficiency seen above was due to the expert system's searching knowledge base, not just access to an online thesaurus.

Ranking

The third hypothesis that the expert system could rank passages in decreasing order of relevance was supported. Although the expert system did present relevant passages significantly earlier than would be predicted by randomness, the improvement was not large enough to be considered truly successful. The current algorithm needs to be evaluated with different weights or a somewhat different algorithm needs to be tried in order to further improve the ranking function. Decreasing the query term weights more quickly as the query terms move farther from the original may improve the ranking by placing more emphasis on the user's original search terms. Using a more sophisticated closeness factor, one that took into account to how many words apart the search terms were in the passage, as well as sentence and paragraph measures considered in this version, could also lead to improved ranking.

Reformulations

Finally, some discussion of the number of reformulations performed by the expert system seems appropriate. The number of reformulations performed for a given user on a given question varies since some unsuccessful starting queries were reformulated before (and sometimes after) a successful starting query is found. The following statistics are given for the final query. The average number of reformulations performed on the starting query for the twelve questions, in order, were: 4.25, 5.75, 4.25, 2.75, 6, 5.75, 3.25, 3.25, 4.25, 2.75, 4, 1. This gives an average of 3.65 reformulations over all final retrieval set queries. It is interesting to note that the highest average number of reformulations is six. If the expert system is continually broadening the query (which is the most common case), this means that even on the question requiring the most reformulations it stops, on average, just after adding child stemgroups. In fact, examining the 48 final retrieval set queries reveals that only in 3 cases did the expert system go past this point. Twice it went one more step and adjusted the context, and once it performed all ten reformulations on the broadening side before the user was satisfied with the number of passages retrieved.

3.5 Questionnaire

The twelve subjects were asked which features of the expert system they liked best. The automatic addition of terms from the thesaurus was the most frequently mentioned (8 subjects), whereas the automatic context adjustment was the second most popular feature (3 subjects). Many subjects (8) mentioned the decreased amount of work needed to perform a search, with three of them specifically mentioning that they did not have to think as much. Other features mentioned which decreased the user effort were the simplified syntax, decreased typing, and the fewer queries to remember.

System slowness was the feature most disliked (6 subjects). Although the amount of time necessary to answer a question was no greater with the expert system (see Section 3.3.2),there was less work for the user so time seemed longer. The other main complaints concerned the user interface. The subjects were fairly evenly split between wanting the system to proceed more automatically, with less prompting from them (4 subjects), whereas others wanted the system to explain what it was doing and/or allow the user to direct it (5 subjects). These comments lead to the conclusion that if a usable system is to be built based on the success of this research prototype, the execution of the system must be sped up and more work on interface design is needed.

Almost all the subjects (10) found the user-expert system version the easiest to use, with the remaining two subjects split between the other two versions. Not surprisingly, given the comparable effectiveness of the three systems, the subjects were split on which system they felt gave the best results. Three voted for the user-alone version, two for the user-thesaurus, and three for the expert system. Three said it was a tie between the user-thesaurus and the expert system, and one abstained.

4 FUTURE WORK

Running the experiment suggested several possible refinements to the system. The experimental subjects had many useful comments, the bulk of which dealt with the desire for a more sophisticated user interface. Desirable changes include: provision of a non-Boolean query language; allowing users to adjust the amount of system interaction; having the user specify the type of search desired, rather than having him give a specific target number; and increasing the speed of the system by improving the way the expert system uses MICROARRAS.

Observing the expert system reformulate real queries gave invaluable insight into which types of queries it handled well, and which it did not. A common type of query that required broadening was one that contained the intersection of three or more concepts. In this case, broadening context and adding search terms to each concept fails to address the fundamental problem of intersecting too many concepts. The next step of replacing the ANDs with ORs is too drastic a change. It invariably leads to too broad a query. Instead, the expert system should take the original query and drop each of the concepts in turn.

The most common type of query requiring narrowing consisted of a single, high-frequency concept. None of the current reformulation techniques were of any use in this case. There were no operators to change, no context to adjust, and adding search terms just makes the query broader. This type of query should be treated as a special case. The concept's child concepts from the thesaurus should be presented as alternative, more specific, queries. The user could also be encouraged to AND this concept with another.

The treatment of multiword phrases entered by the user which do not appear in the thesaurus should be changed. Currently, the only expansions done are expanding each word to its stemgroup and loosening the context allowed between the words of the phrase. It would be preferable to treat the words of the phrase as separate concepts which are ANDed together with adjacent context. Each phrase word could then be expanded using the full range of thesaural relationships, as is the case with regular search terms.

Finally, more work is needed to improve the ranking of the retrieved passages. The current ranking algorithm should be tried with different weights for the query search terms and the closeness factor. It may be necessary to try entirely different algorithms, possibly incorporating syntactic or semantic information, to achieve high quality ranking.

In conclusion, we have demonstrated that an expert system can provide online search assistance to improve the efficiency of novice searchers. Whereas more research is necessary to develop a better search assistant, I have been able prove that a useful search assistant can be developed which separates the search strategies from the domain knowledge, and that implementation of such a system is feasible now.

REFERENCES

Al-Hawamdeh, S., de Vere, R., Smith, G., and Willett, P. (1991). Using nearest-neighbour searching techniques to access full-text documents. *Online Review, 15* (3/4), 173-191.

Bates, M.J. (1979). Information search tactics. *Journal of the ASIS*, *30*(4), 205-214.

Belkin, N.J. & Marchetti P.G. (1990). Determining the functionality and features of an intelligent interface to an information retrieval system. *Proceedings of the Thirteenth Annual International ACMSIGIR Conference on Research & Development in Information Retrieval,* Brussels, ACM Press, 151-174.

Blair, D.C., and Maron, M.E. (1985). An evaluation of retrieval effectiveness for a full-text document-retrieval system. *Communications of the ACM, 28*(3), 289-299.

Blaauw G.A. and Brooks, F.P., Jr. (Fall, 1986). Computer Architecture, Volume 1-Design Decisions, Draft.

Borgman, C.L. (1986a). Why are online catalogs hard to use? *Journal of the ASIS*, *37*(6), 387-400.

Borgman, C.L. (1986b). The user's mental model of an information retrieval system: an experiment on a prototype online catalog. *International Journal of Man-Machine Studies, 24(*1), 47-64.

Borgman, C.L. (1987). Individual differences in the use of information retrieval systems: Some issues and some data. *Proceedings of the Tenth Annual International ACMSIGIR Conference on Research & Development in Information Retrieval,* New Orleans, ACM Press, 61-69.

Brajnik, G., Guida, G., and Tasso, C. (1988). IR-NLI II: Applying man-machine interaction and artificial intelligence concepts to information retrieval. *Proceedings of the Eleventh Annual International ACMSIGIR Conference on Research & Development in Information Retrieval*; Grenoble, ACM Press, 387-399.

Chang, S. and Chow, A. (1988). Towards a friendly adaptable information retrieval system. *Proceedings of RIAO 88*, M.I.T., 172-182.

Chen, H. & Dhar V. (1990). Online query refinement on information retrieval systems: A process model of searcher/system interactions. *Proceedings of the Thirteenth Annual International ACMSIGIR Conference on Research & Development in Information Retrieval,* Brussels, ACM Press, 115-133.

Chiaramella, Y. and Defude, B. (1987). A prototype of an intelligent system for information retrieval: IOTA. *Information Processing & Management, 23*(4), 285-303.

Croft, W. B. and Thompson, R.H. (1987). I^{3}R: A new approach to the design of document retrieval systems. *Journal of the ASIS, 38*(6), 389-404.

Croft, W. B. and Turtle, H.R. (1992). Text retrieval and inference. In P.S. Jacobs (Ed.), *Text-Based Intelligent Systems: Current Rsearch and Practice in Information Extraction and Retrieval.* Hillsdale, N.J.: Lawrence Erlbaum Associates.

Crouch, C.J. (1988). A cluster-based approach to thesaurus construction. *Proceedings of the Eleventh Annual International ACMSIGIR Conference on Research & Development in Information Retrieval,* Grenoble, ACM Press, 309-320.

Fenichel, C.H. (1981). Online searching: measures that discriminate among users with different types of experience. *Journal of the ASIS, 32*(1), 23-32.

Fidel, R. (1991). Searcher's selection of search keys. *Journal of the ASIS, 43*(7), 490-500.

Gauch, S. (1991). Search improvement via automatic query reformulation. *ACM Transactions on Information Systems, 9*(3), 249-280.

Gauch, S. (1992). Intelligent information retrieval: An introduction. *Journal of the ASIS, 43*(2), 175-182.

Gauch, S. & Smith, J.B. (1993). An expert system for information retrieval. *Journal of the ASIS,* 44(3), 124-136.

Harman, D. (1986). Individual differences in the use of information retrieval systems: Some issues and some data. *Proceedings of the 1986 ACM Conference on Research & Development in Information Retrieval,* Pisa, ACM Press, 61-69.

Harman, D. (1988). Towards interactive query expansion. *Proceedings of the Eleventh Annual International ACMSIGIR Conference on Research & Development in Information Retrieval,* Grenoble, ACM Press, 321-331.

Jacobs, P.S. & Rau, L.F. (1990). SCISOR: Extracting information from on-line news. *Communications of the ACM, 33,* 88-97.

Krawcsak, D., Smith, P.J., & Shute, S.J. (1987). EP-X: A demonstration of semantically-based search of bibliographic databases. In C.T. Yu and C.J. van Rijsbergen (Ed.), *Proceedings of the Tenth Annual International ACM SIGIR Conference on Research & Development in Information Retrieval* . New Orleans, ACM Press, 263-271.

Marcus, R.S. (1994). Intelligent assistance for document retrieval based on contextual, structural, interactive Boolean models. *Proceedings of RIAO 94: Intelligent Multimedia Information Retrieval Systems and Management*. New York, C.I.D., Vol. 2, 27-43.

McCune, B.P., Tong, R.M., Dean, J.S., Shapiro, D.G. (1985). RUBRIC: A system for rule-based information retrieval. *IEEE Transactions on Software Engineering, SE-11*(9), 939-945.

Oldroyd, B.K. (1984). Study of strategies used in online searching 5: differences between the experienced and the inexperienced searcher. *Online Review, 8*(3), 233-244.

Pollitt, A.S. (1984). A 'front-end' system: An expert system as an online search intermediary. *ASLIB Proceedings, 36*(5), 229-234.

Pollitt, A.S. (1987). CANSEARCH: An expert systems approach to document retrieval. *Information Processing & Management, 23*(2), 119-136.

Ro, J.S. (1988). An evaluation of the applicability of ranking algorithms to improve the effectiveness of full-text retrieval. II. On the effectiveness of ranking algorithms on full-text retrieval. *Journal of the ASIS, 39* (3), 147-160.

Salton, G. (1986). Another look at automatic text-retrieval systems. *Communications of the ACM, 29* (7), 648-656.

Shoval, P. (1985). Principles, procedures and rules in an expert system for information retrieval. *Information Processing & Management*, *21* (6), 475-487.

Smeaton, A.F., and van Rijsbergen, C.J. (1983). The retrieval effects of query expansion on a feedback document retrieval system. *The Computer Journal, 26* (3), 239-246.

Smith, J.B., Weiss, S.F., and Ferguson, G.J. (1987). MICROARRAS: An advanced full-text retrieval and analysis system. *Proceedings of the Tenth Annual International ACMSIGIR Conference on Research & Development in Information Retrieval*, New Orleans, ACM Press, 187-195.

Smith, P.J., Shute, S.J., Galdes, D., and Chignell, M.H. (1989). Knowledge-based search tactics for an intelligent intermediary system. *ACM Transactions on Information Systems*, *7*(3), 246-270.

Tong, R.M., Applebaum, L.A., Askmann, V.N., and Cunningham, J.F. (1987). Conceptual information retrieval using RUBRIC. *Proceedings of the Tenth Annual International ACMSIGIR Conference on Research & Development in Information Retrieval*; New Orleans, ACM Press, 247-253.

Vickery, A., and Brooks, H.M. (1987). PLEXUS-The expert system for referral. *Information Processing & Management 23* (2), 99-117.

Zadeh, L.A. (1965). Fuzzy sets. *Information and Control*, *8*, 338-353.