I^3R: A New Approach to the Design of Document Retrieval Systems

W.B. Croft
R.H. Thompson

COINS Technical Report 87-58
July 23, 1986

Abstract

The most effective method of improving the retrieval performance of a document retrieval system is to acquire a detailed specification of the user's information need. The system described in this paper, I^3R , provides a number of facilities and search strategies based on this approach. The system uses a novel architecture to allow more than one system facility to be used at a given stage of a search session. Users influence the system actions by stating goals they wish to achieve, by evaluating system output, and by choosing particular facilities directly. The other main features of I^3R are an emphasis on domain knowledge used for refining the model of the information need, and the provision of a browsing mechanism that allows the user to navigate through the knowledge base.

1 Introduction

Document retrieval systems have typically provided three main facilities to their users; a language for specifying a query, an algorithm for retrieving documents based on a comparison of the document contents with the query, and a file organization that supports an efficient implementation of the retrieval algorithm. These facilities are similar to those provided in database systems for data records with well-defined attributes. For example, records that describe attributes of employees such as name, age and salary can be efficiently stored and retrieved in a relational database system [1]. In fact, document retrieval is often seen as a special case of database retrieval where the major difference is that the file organization must support large text attributes [2,3]. The approach to document retrieval that uses queries expressed as Boolean combinations of keywords and retrieves documents that contain exactly that combination of keywords is firmly based on database retrieval. This view, however, obscures the features of document retrieval that make it a challenging and difficult research area.

The users of a document retrieval system have a wide range of information needs. In some cases, a user may be able to provide a query that accurately describes an information need. More typically, the query will not accurately specify the type of documents required. For example, there is a large difference between a database query such as

Find all employees with age > 30 *and salary* $< 20,000$

and a document retrieval query such as

Find all documents about database systems and retrieval techniques

even though these queries may be expressed in very similar languages. The database query provides an exact specification of the content of the desired records whereas the document retrieval query provides only an indication of the content of the desired documents. The actual content of the documents identified as relevant by the user may vary considerably from the phrases provided in the query.

This brings us to another feature of document retrieval systems. The goal of such a system is to retrieve documents that are relevant to a particular user's query. However, the property of relevance cannot be established by a simple comparison of the contents of documents with the query. Given a particular document citation, a user will identify its relevance to the information need by a process of inference. That is, the content of the document abstract (say), together with the user's knowledge of the subject area, is used to infer that the document will be a valuable source of information. For a document retrieval system to be effective, it has to support this process of inference. The success of a system in achieving the goal of retrieving relevant documents can be measured in terms of *recall* and *precision* [4].

Much of the research in information retrieval has concentrated on indexing techniques for representing the contents of documents and retrieval techniques that compare documents to queries [4,5]. These techniques have supported the retrieval inference process and the resolution of uncertainty associated with that process by a variety of methods. The most common method has been to base retrieval and indexing techniques on the statistical properties of text. Probabilistic models of document retrieval are the best example of this approach [5]. Given a query and a set of documents represented by index terms derived from the text, the retrieval algorithms that are based on the probabilistic model infer a *probability of relevance* for individual documents and rank them accordingly. The probability of relevance is based on the frequencies of occurrences of index terms in the relevant and non-relevant sets of documents. The top-ranked documents are shown to the user and the relevance judgments that are made provide information that enables the information need to be more precisely specified. This process of *relevance feedback* is one way of addressing the problem of ill-defined queries. The vector model [4] of document retrieval provides essentially the same form of inference. The advantages of using any of these statistical techniques include the following;

- They are efficient to implement [6].
- They are more effective in terms of finding relevant documents than searches based on Boolean queries/exact matching [7].
- They have a sound theoretical basis.
- They are independent of any particular domain. That is, different types of documents (journal articles, office memos) from different domains (medicine, law) can be handled using the same techniques.

The major disadvantage of using these techniques is that absolute performance is low. Although statistical techniques provide better performance than conventional database techniques, there is still a lot of room for improvement.

Given that statistical techniques appear to have reached a ceiling of performance, researchers are investigating the application of techniques for natural language processing and knowledge representation [8,9,10]. There are, however, inherent difficulties in applying artificial intelligence (AI) techniques to the information retrieval task. In particular, the amount of domain-specific knowledge used in even a small collection of documents is very large. In many applications, the number of documents is also very large and dynamic. A natural way of limiting the amount of domain knowledge that must be considered is to concentrate on acquiring knowledge from individual users about specific queries. This domain knowledge can then be used to improve retrieval effectiveness by supporting inference based on the domain *concepts* (important words or phrases) in the queries and documents. The process of identifying concepts using domain knowledge can be seen as an extension of the process of identifying important index terms using limited statistical information. When domain knowledge is not available, the document retrieval system should be able to use the well-established statistical techniques to guarantee a reasonable level of performance.

The use of domain knowledge raises a number of questions about the design of document retrieval systems. Some of these are;

- How is domain knowledge acquired, represented and used?
- Can an inference process based on domain knowledge be combined with statistical techniques?
- How are the techniques used by the system explained to the users?

In this paper, we describe a document retrieval system design (I^3R) that addresses these issues. In contrast to document retrieval systems that provide only a query language and a single retrieval strategy, the I^3R system provides a range of facilities for query formulation, domain knowledge acquisition, explanation, browsing, retrieval and evaluation. It supports inference based on statistical techniques and domain knowledge. New system facilities can easily be added.

I^3R is designed to act as an *expert intermediary* (hence the name *I*ntelligent *I*ntermediary for *I*nformation *R*etrieval). That is, the types of activities undertaken by the system during a search session are similar to those done by a human intermediary. The facilities that are available to I^3R, however, are superior to those available to the typical intermediary. For example, I^3R can provide direct access to sections of the database for browsing and can select different retrieval algorithms. The notion of the expert intermediary should be viewed, therefore, as a means of structuring a search session rather than as an attempt to do an accurate simulation of human intermediaries. The incorporation of the intermediary concept into an entire system design and the integration with advanced probabilistic strategies distinguishes I^3R from other "intermediary" systems that provide access to conventional bibliographic retrieval systems [11,12]. Belkin *et al* [20] describe a system architecture similar to I^3R that associates the various actions performed by intermediaries with independent system modules.

The basis of the I^3R approach is presented in the next section. This section includes a discussion of the research that contributed directly to the design choices that were made. The third section describes the general I^3R software architecture which is based on the Hearsay-II system [13]. This architecture provides a flexible framework for implementing system facilities and controlling their use. An example of the system's operation follows. The next two sections discuss the representation and use of domain knowledge, and the browsing facility. Section 7 discusses the implementation and evaluation of the system. Evaluation is made particularly difficult by the heavily interactive nature of the search sessions. A number of issues brought up during the design of I^3R remain to be investigated. The most important of these is the use of more powerful natural language processing techniques. This is discussed in the last section.

2 The Basis for I^3R

A typical session with I^3R will involve three major phases; query formulation and refinement, search, and user evaluation. A major design feature of I^3R is to provide a variety of facilities or system experts that can contribute to these phases and to select experts that are appropriate for a particular user and session. For example, there is a system expert for acquiring domain knowledge and one for processing natural language queries. If a user's query provides enough information, domain knowledge acquisition may not be initiated. The basis for the operation of I^3R can be expressed in one simple principle;

> The Quality-In Quality-Out Principle: *A query that more accurately reflects the user's information need will produce better results.*

This principle, although intuitively obvious, is a succint way of summarizing a range of experimental results. A more formal way of expressing it comes from the view that document retrieval is a process of inference [14]. If the contents of a document representative (such as the title and abstract) can be used to infer the statements made in the query, the document is retrieved. For example, a user may read an abstract and infer that the document will address the information need. A retrieval system provides a query language for describing the information need and a particular inference mechanism, such as that used in the probabilistic model. This inference mechanism is limited in comparison to the inferences that would be made by the users of the system. The probabilistic model, for example, typically uses only information about the frequencies of occurrence of single words in documents to infer that a document is relevant (i.e. matches the query). Since the system's inference mechanism is limited in comparison to that of the users, the documents it retrieves will not all be relevant. The other source of error in retrieval is that the query only approximates the actual information need. A query that is a poor representation of the information need will lead to poor retrieval results. The two ways to improve retrieval performance are, therefore, to enhance the inference process used by the system and to acquire better descriptions of the information need. The first issue is addressed by research in natural language processing techniques that is described in the last section. The second issue is a major theme of this paper and the I^3R architecture.

The Quality-In Quality-Out principle applies to systems that use statistical techniques, exact matching of text strings, or matching of domain concepts. Using statistical techniques based on the probabilistic model as an example, the early retrieval experiments were done with queries represented as unweighted sets of index terms. Other experiments showed that when users were able to specify important dependencies or groups of terms in the query and the relative importance of query terms, retrieval performance improved significantly [15]. For example, the document retrieval query used in the first section could be represented as a set of terms {`database, systems, retrieval, techniques`}. The query more accurately reflects the information need when `database systems` and `retrieval techniques` are identified as important groups of terms. These dependencies can be identified in a variety of ways, including analyzing the query term combinations in Boolean queries or the structure of natural language queries.

The I^3R system, during the query formulation and refinement phase, attempts to build an accurate description of the information need. This description is referred to as the *request model*. The request model contains, amongst other things, a list of concepts and their relative importance (see Section 4 for details). These concepts are derived from the user's query and the domain knowledge. The domain knowledge itself may be acquired during this phase. The knowledge that is specific to an individual user is stored as part of a *user model*.

The information in the request model is used by the retrieval strategies to retrieve ranked lists of documents. When retrieval is viewed as an inference process, the ranking is produced by the certainty of the inference for each document. The current version of the system provides two statistical retrieval strategies. The main strategy is based on the probabilistic model and uses information about dependencies between terms and their relative importance derived from the concepts in the request model. For example, the request model may list `retrieval technique` as a concept with an associated importance of 0.7. This is used to specify that the words `retrieval` and `technique` should occur together in relevant documents. The documents are ranked according to the score [15,16]:

$$\sum_i T(x_i)W(x_i)x_i + A$$

where x_i is the ith term in the set of terms describing a document, $T(x_i)$ is the weight related to the term's importance, $W(x_i)$ is a weight related to the collection frequency of term i, and A is a factor that depends on the presence of dependent groups of words in a particular document. The summation is done over all the terms in the query.

The second strategy used is a cluster-based search [6]. This additional strategy is provided because retrieval experiments have indicated that the two strategies tend to retrieve different sets of documents and, in particular, the cluster search may locate relevant documents when the probabilistic search fails. The cluster search is invoked when I^3R decides that it is appropriate. This essentially means it will be invoked when the probabilistic search fails to find relevant documents or when high recall results are required.

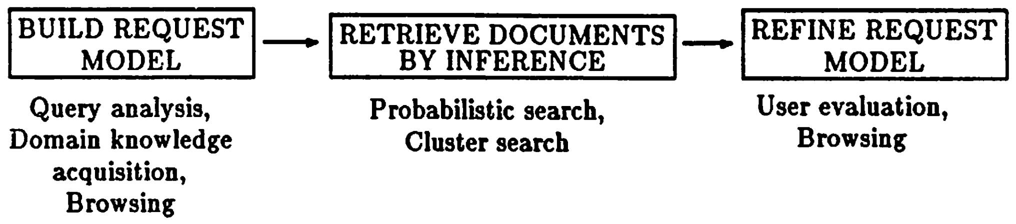

BUILD REQUEST MODEL
RETRIEVE DOCUMENTS BY INFERENCE
REFINE REQUEST MODEL
Query analysis, Domain knowledge acquisition, Browsing
Probabilistic search, Cluster search
User evaluation, Browsing

Figure 1: Basic Processes.

The documents in the I^{3}R system are indexed by conventional statistical techniques that produce index terms based on single words [4]. Statistical retrieval techniques can easily be extended to deal with multi-word concepts. In future versions of the system, we intend to use natural language processing (NLP) techniques to identify these concepts for document indexing. Some research has been done in this area, but many problems remain to be solved [17,18]. Another research direction being investigated is the use of NLP to compare the request model to the text in the abstracts and titles of documents retrieved using statistical techniques. This would provide a more powerful inference mechanism that could be used as an additional filter in the retrieval process. The use of NLP techniques is discussed further in the last section.

The evaluation phase involves the user looking at lists of retrieved documents. The user is able to identify relevant documents and parts of documents such as phrases that are important. This information is used to update the request model. Evaluation is done in conjunction with the browsing facility. Browsing has been shown to be an important adjunct to the document retrieval process [19]. The I^{3}R browsing facility provides access to the documents and other information in the database, along with commands to navigate along links to related information. Browsing can also be an important part of the query formulation and refinement phase. For example, if a user specifies a particular document instead of a conventional query, this document can be used to start browsing and building up the request model.

Figure 1 summarizes the basic processes in the I^{3}R system.

3 The I^{3}R Architecture

The processes described in the previous section place strong requirements on the architecture of the system. In particular, the system must provide a means of monitoring the progress of a session and controlling the use of the system facilities during the session. These facilities, such as statistical search strategies, browsing, and query analysis, work on different aspects of the common problem of characterizing a user's information need and locating documents that satisfy that need. In some cases, the system facilities may be competing for system resources and user attention. Human intermediaries also must be able to satisfy a number of potentially conflicting goals during their interviews with users [20,21]. These goals, which deal with different types of knowledge acquisition, could be satisfied by following a rigid plan or sequence of actions. Analysis of typical interviews, however, show that the intermediary is able to both respond to the requirements of individual users and to structure the dialogue appropriately [21].

The same type of flexibility should be demonstrated by document retrieval systems. In the case of the I^3R system, the flexibility lies in determining which system facilities are appropriate and the order of using these facilities during a session. For example, some users may require a very simple retrieval session where they specify a query as a Boolean combination of index terms, the system retrieves documents, and the retrieved documents are displayed for inspection. Other users may be prepared to spend more time during the query formulation and refinement phase doing browsing and providing domain knowledge in order to find more relevant documents. Supporting these different types of sessions places additional constraints on the possible architectures for I^3R.

The Hearsay-II system architecture [13] was designed to support multiple, independent *knowledge sources* that are cooperating on the solution of a common problem. A knowledge source can be thought of as an expert in some aspect of problem-solving. The sequence of activation of the knowledge sources is determined at run-time by a *scheduler*. This enables the system to respond flexibly to different situations that arise during the solution of the problem. The knowledge sources communicate indirectly through a shared data structure called a *blackboard*. These system characteristics match very closely the requirements for the I^3R system, and this led to I^3R being implemented using a variation of the Hearsay-II architecture. It is also interesting to note that an "information provision" (or intermediary) system architecture based on the Hearsay-II model of independent experts limited to data-directed interactions through the blackboard has successfully been simulated using human experts [20].

The major system components of I^3R are shown in Figure 2. The system experts handle different aspects of the operation of I^3R. Examples of these experts are the request model builder, the user model builder, and the browser. The experts are capable of operating independently and communicate through the working memory section of the blackboard. The scheduler controls the activation of the experts and makes use of a *plan* and an *agenda* stored in the blackboard. The agenda records the actions that system experts could carry out in a given situation. The plan records information about the state of a session and which actions are preferred in a given state. Using the plan, the scheduler determines which action on the agenda has the highest priority and activates the corresponding expert. The *knowledge base* contains a range of information about documents, users, and domain concepts. In particular, it contains the representations of document contents that are used by the retrieval strategies. The *interface manager* is responsible for the entire user interface of the system. All output to the user and input from the user goes through the interface manager via the blackboard. The following discussion gives more details on the structure and operation of these system components.

System experts

Each system expert is made up of a number of *rules*. A rule consists of two parts; a condition part and an action part. The condition part specifies the state of the blackboard that indicates the applicability of the rule. The condition part is evaluated by comparing the specified blackboard state to the current state. It can be thought of as a Boolean expression that evaluates to either true or false. The action part specifies the actions that are carried out when the condition part is true and the rule is activated. Any valid program can be an action

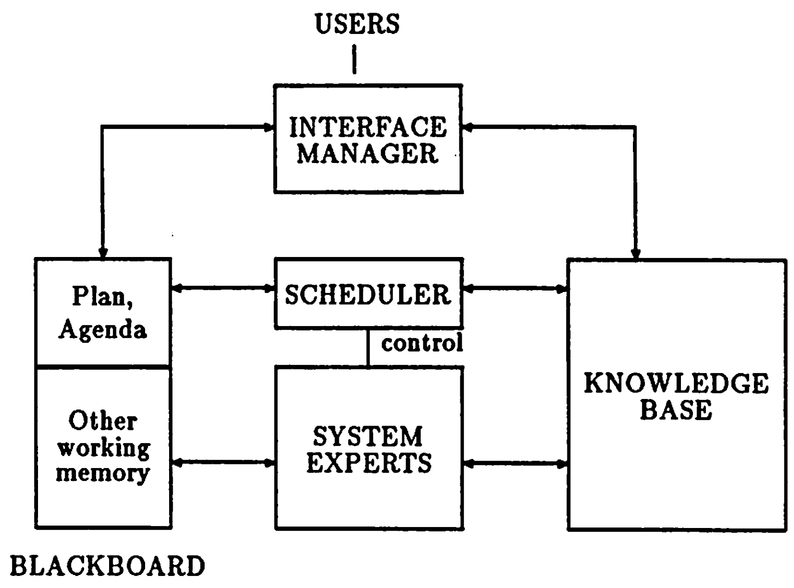


Figure 2: Major System Components.

part of a rule. Examples of rules are (in simplified form);

1. *Condition:* Request model contains more than three index terms and it contains dependencies.
   Action: Perform probabilistic retrieval, place retrieved documents on blackboard in request model.

2. *Condition:* Request model contains unevaluated documents.
   Action: Obtain user evaluations of documents, update request model.

The first rule forms part of the search controller expert, the second is part of the request model builder expert.

In the standard Hearsay architecture, a rule would be a single knowledge source. In the I^3R system, a knowledge source or expert can contain several related rules, thereby simplifying the task of the scheduler. To make this point clearer, consider a search controller expert that contains four rules. If these rules were individual knowledge sources, they may compete for resources and the scheduler would have to decide between them. On the other hand, if the four rules were made into one "super" rule by combining the condition and action parts, the situations that cause particular actions to take place are hidden in the condition and action part code and could not be easily used to explain the system's actions to the users. When system experts are allowed to contain multiple rules, situations may occur that would activate more than one of these rules. In I^3R a simple strategy is provided for choosing amongst rules in this situation. This "conflict resolution" strategy is to choose the rule with the most specific condition. If the strategy for choosing becomes complex, then the rules should be made into individual experts.

The experts in the current I^3R system are as follows;

- The *User Model Builder* (UMB) collects information about the user in order to determine if a particular *stereotype* applies [22]. Stereotypes are used to determine the style of interaction, goals of the session and other information. Identification of a stereotype is done with user-supplied information such as an interest in high recall. In I^3R stereotypes are simply global parameters. The other major function of the user model builder is to acquire knowledge about the user's domain of interest. This domain knowledge is used to refine the request model.

- The *Request Model Builder* (RMB) constructs a model of the user's information need. The major part of this task is to obtain an initial query from the user and to perform simple statistical indexing on this query to obtain index terms and weights. A number of different forms of query specification are allowed including natural language, words connected with Boolean operators, and words or phrases with associated weights. The request model builder also obtains relevance judgments on retrieved documents and information about the content of individual relevant documents.

- The *Domain Knowledge Expert* (DKE) uses the domain knowledge from the user model and the knowledge base to infer concepts that are related to those in the initial query. These concepts are presented to the user for evaluation and eventual inclusion into the request model. The inference process used by this expert is discussed further in section 5.

- The *Search Controller* (SC) selects and executes formal search strategies. The strategies are based on the probabilistic model and clustering. Retrieved documents are placed in the request model.

- The *Browsing Expert* (BE) provides the user with an informal method of finding relevant documents by navigating through the knowledge base. The browsing process can start from a given document, author or index term and follows links to other items in the knowledge base. The knowledge acquired during browsing is used to update the request model and may trigger a formal search strategy. Browsing is essentially a user-directed activity but the system provides recommendations of interesting paths. This expert is discussed in more detail in section 5.

- The *Explainer* is used to explain system actions in response to user demands. The technique used is similar to other rule-based systems where an action is justified by displaying the sequence of rules that led to it. A record of the rules that are activated in the system (and the rules used in the domain knowledge expert) is stored in the blackboard.

Interface manager

The interface manager is separate from the rest of the I^3R system. It looks at a particular part of the blackboard to find requests for information from the user and information to display to the user. Details of how particular types of information are displayed or requested are the responsibility of the interface manager. Examples are the layout of a browsing window, how to display a list of documents, and how to obtain relevance judgments. The requests on the blackboard for input or output are in the form of coded messages. For example, to display a retrieved list of documents and allow input from the user on relevance, a message need only contain a code and a pointer to the retrieved list of documents on the blackboard. By placing input and output on the blackboard, each expert is given an opportunity to examine and evaluate it.

Scheduler

The task of the scheduler is to select, from an agenda of possible activities, the activity that is most appropriate for the current state of the search session. The selection is based on assigning priorities to system experts and the rules they contain. The explanation expert, for example, always has the highest priority because it will only be invoked by the user. The priorities of the other experts are determined by a plan that also specifies goals that must be met during a session. These goals are based on those of intermediaries and have definite relationships to each other. For example, the system cannot search for documents before getting a description of the information need. The goals, in turn, have subgoals and the terminal nodes of this goal tree determine the current state of the session and the priorities of the system experts. Figure 3 shows the default plan for the system. The terminal nodes for this plan specify not only the priorities of system experts (shown by the ordering), but also whether a particular expert is appropriate at a given stage of the session. For example, the Search Controller is not considered for activation during the session phase called *Characterize User*.

The plan is represented by a network of goal states and transitions. The transitions can be either default or exception. The default transitions give the normal path through the goal states, while the exception transitions allow alternative paths that may be taken when the session is not producing the expected results. Associated with each transition are conditions that determine when that transition is taken. The conditions are similar to the condition part of rules. For example, the condition for the transition from the state *Characterize User* to *Characterize Information Need* is that a user model containing at least one stereotype has been created. An exception transition from *Evaluate Results* to *Search for Documents* occurs when no relevant documents have been found and the user has finished evaluating documents. Exception transitions can easily be added to the plan to increase the flexibility of the system. Plans could also be set up for individual users.

We are now able to describe the basic cycle of the system in more detail. Apart from the interface manager which operates as an independent process, the cycle is as follows;

1. The scheduler examines the current state of the plan in the blackboard and determines if a transition should be made. This determines the session state, the experts that may run, and their priorities for the next cycle.

2. The experts that are enabled by the scheduler examine the blackboard to evaluate the condition parts of their rules. A *blackboard monitor* simplifies this process by recording which states have changed and the rules that are affected. Agenda records are created for rules whose condition parts are true. If the condition part of more than one rule in an expert evaluates to true, one rule is selected.

3. The rules mentioned in agenda records are activated in order of the priority determined by the scheduler. As the action parts of rules may modify the blackboard state, the

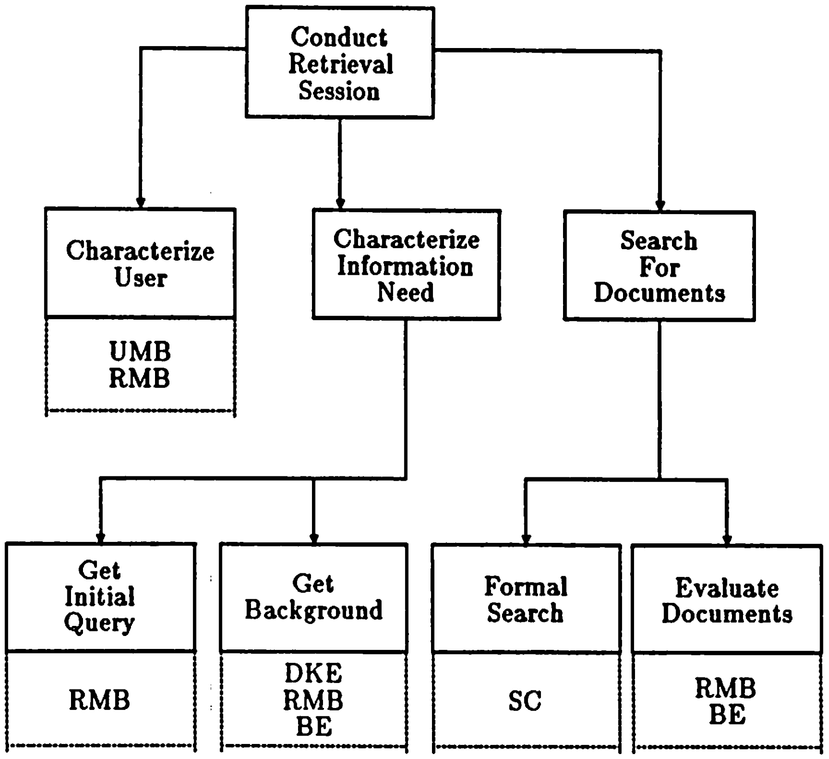


Figure 3: Default Scheduler Plan

monitor determines if a change may have "poisoned" the condition part of an agenda record waiting for activation.

4. Go back to step 1.

Blackboard and knowledge base

The blackboard and knowledge base provide the means of describing and storing the data used in the I^3R system. The blackboard contains information that relates to a particular session and is highly dynamic. The knowledge base provides more permanent storage. Unlike the Hearsay blackboard, the I^3R blackboard is not divided into different levels of abstraction. It simply has six areas that are modified by the system experts, the scheduler and the interface manager. It may be possible to create abstraction levels for the request model, in particular, but this is not currently done. The following is a list of the areas of the blackboard;

- *Plan:* Instantiated for a particular session, records current goal state and transitions.

- *Agenda:* Activation records. Each record lists an expert and a rule in that expert.
- *User Model:* Stereotypes and domain knowledge.
- *Request model:* Initial query specification, indexed terms, dependencies, concepts, associated weights, retrieved documents, relevant documents, non-relevant documents, relevant authors.
- *I/O Requests:* Messages to/from the interface manager.
- *System Journal:* Record of rules invoked during session.

The knowledge base can be described in terms of entities and relationships between these entities. The entities represent important objects in the I^{3}R system and the relationships capture information about the connections between these objects. The entities are documents, terms, concepts, authors, users and sessions. These entities have a number of properties such as the title and abstract for a document. There are a number of types of relationships such as statistical (e.g. document-document similarity measure), bibliographic (authors and documents), or semantic (synonyms). Figure 4 is a description of the current knowledge base.

The statistical relationships between document pairs and term pairs connect the *nearest neighbors* according to a similarity measure. This means that documents (terms) that are described by very similar sets of terms (documents) will be connected. The nearest neighbor connections form the basis of cluster searches and probabilistic searches based on term dependencies [6]. The relationships between concepts form part of the domain knowledge and can be of many different types. Concepts and their relationships are described further in section 5. Relationships between users and concepts identify the domain knowledge supplied by the user.

The knowledge base can also be thought of in terms of a frame-based representation where each entity description is stored as a frame and the relationships are links between the frames [23]. Note that the system expert rules are usually regarded as part of the knowledge base.

4 An Example Session

To further clarify the operation of the I^{3}R system, a brief example of a session will be given. This session will intentionally be kept simple.

The user is initially presented with an "identification" window by the interface manager. This window is used to enter identification data, such as an account name and password. The interface manager then posts a message on the blackboard that identifies the start of a session and the user. The scheduler instantiates the plan on the blackboard with a marker indicating that the session is in the first state (*Characterize User*). The basic cycle of the system then begins.

The condition for transition from *Characterize User* is that a user model containing at least one stereotype exists. There is also the general condition that no transition can be made when there are I/O messages remaining on the blackboard. The scheduler does not change the session state and enables the UMB and the RMB (see abbreviatons in last secton). The message on the blackboard corresponds to the condition part of a rule in the UMB. This rule is entered in the agenda. The RMB would only be appropriate when a user is restarting a previous session.

The UMB rule is then activated by the scheduler. This rule removes the interface manager message from the blackboard and then attempts to find a user model for this user in the

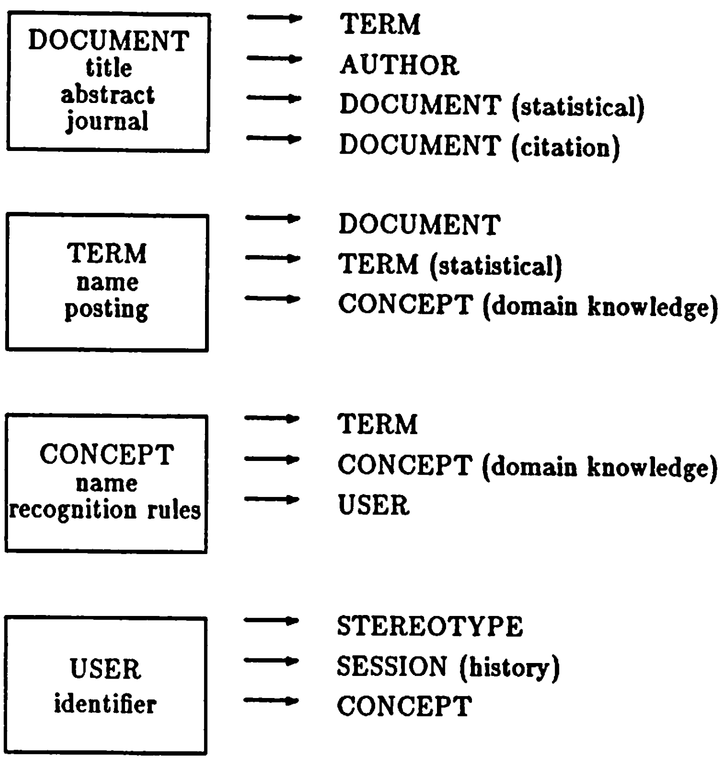


Figure 4: The Knowledge Base

knowledge base. In this case, a model does exist and is placed on the blackboard. This model contains a stereotype for an experienced user. This information can be used by the interface manager to change the style of interaction. The UMB then posts a message for the interface manager to ask the user for session-specific information. At this point, control is returned to the scheduler. Since there is an outstanding message on the blackboard, the session state cannot change. The changes to the blackboard do not affect any other rules and the system waits for the return message.

The interface manager puts a menu of stereotypes on the screen. The user indicates an interest in high recall results and this information is placed in a message on the blackboard. A UMB rule is then activated by the scheduler. This rule updates the user model by including a high-recall stereotype and updates the condition for the termination of the *Formal Search* plan state. This condition normally specifies that a list of retrieved documents must exist. It is modified to specify that more than one list must exist. This will ensure that multiple search strategies are used.

As there are no outstanding messages and the user model exists, the scheduler changes the plan state to *Get Initial Query*. In this state, the RMB is the only enabled expert. It instructs the interface manager to display a screen that allows the user to specify an initial query in a variety of ways. Figure 5(a) shows the screen at this stage of the session. All windows on the screen have two menus associated with them. The "Window" menu contains commands that relate to placing the window, scrolling it, etc. The "Content" menu contains commands that relate to the content of the window. These menu commands are selected using the mouse pointing device.

In this session, the user gives a short natural language query (Figure 5(b)). This is processed by the RMB and placed in the request model. If the user had given a different form of query, such as indicating a relevant document, further dialogue would take place. In our example, however, the condition for transition is that a request model with index terms exists and as there are no outstanding messages, the transition to *Get Background* is taken.

The *Get Background* stage of the session is potentially the most complex, as a variety of techniques can be used to refine the request model and acquire domain knowledge. Specifically, the system can call on the DKE to attempt to infer related concepts to the query and confirm them with the user. If the knowledge base does not contain any information relating to the current query, the DKE may request the user to supply domain knowledge as described in the next section. The RMB may request additional keywords to expand the query if the initial query contained too many high-frequency terms. The initial query can also be used as an entry point for the browsing expert and additional information for the request model will be obtained as the user browses the knowledge base.

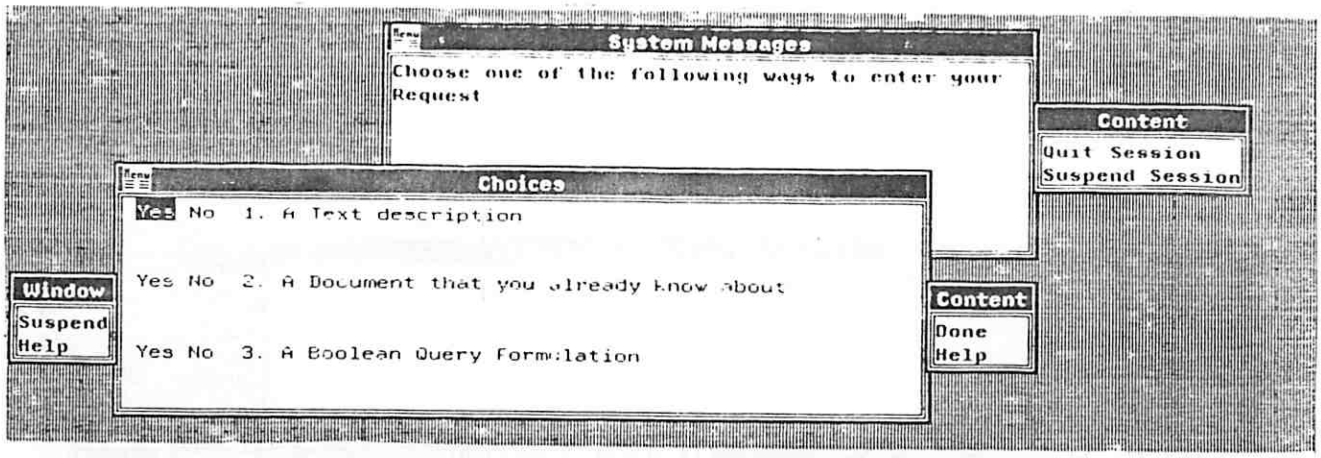
The system, however, also provides a facility that allows the user to "skip" a session stage. This facility is simply a menu command that results in a message being placed on the blackboard. This message is recognized by the scheduler and causes a transition to the next state. This is what happens in our example session.

The next state is *Formal Search* where only the SC is enabled. The first rule that is activated uses the probabilistic retrieval strategy to retrieve a ranked list of documents. The transition condition calls for multiple lists of retrieved documents so the (state exit) transition is not made. The rule that invokes the cluster search includes a condition that a retrieved list of documents must already exist. This rule is activated and an additional list of retrieved documents is created. The transition to *Evaluate Documents* is made.

In this state, both the BE and the RMB can be used to obtain relevance judgments and refine the request model. Rules from both experts will appear on the agenda. The RMB rule has priority and interacts with the user. Figure 5(c) shows the screen with the results of both searches and some documents already identified by the user as being relevant. The bar charts indicate the probable relevance of the documents according to their scores calculated using the retrieval algorithms. Figure 5(d) shows the screen after the user has chosen to "Show" a document's contents. The highlighted portions of the document text indicate terms that were considered interesting by the user. The Content menu for this window contains commands that allow the user to start browsing.

The final step is to ask if the user is satisfied with the results. If the answer is negative, the session cannot finish and the scheduler looks for exception transitions. Exception transitions to *Get Background* and *Formal Search* exist. The first transition will acquire more details for the request model, the second is the means of implementing relevance feedback. The transition that is taken will depend on the history of the session and, in this case, the transition will be to *Get Background* because this state was "skipped".

The example session shows that, from the point of view of the user, the interaction with I^3R can be as straightforward as a conventional system. The difference lies in the variety of facilities and the flexibility of their application in I^3R.



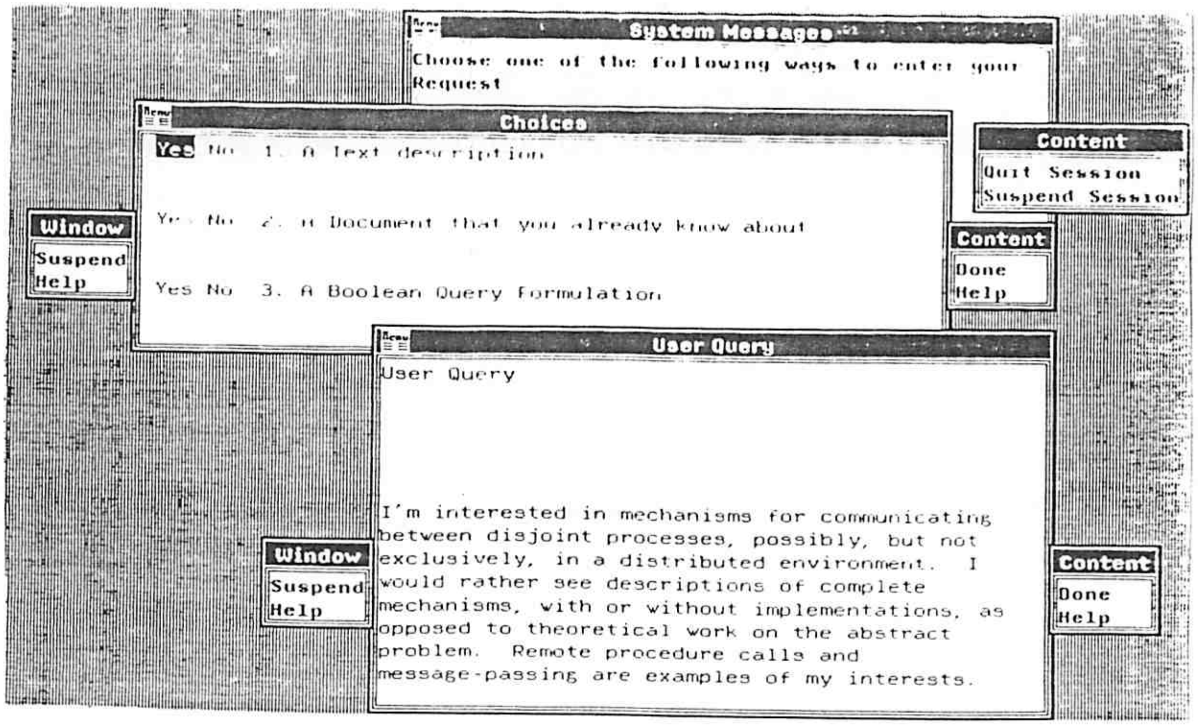
Choose one of the following ways to enter your Request

Yes No 1. A Text description

Yes No 2. A Document that you already know about

Yes No 3. A Boolean Query Formulation

5(a)



Choose one of the following ways to enter your Request

Yes No 1. A Text description

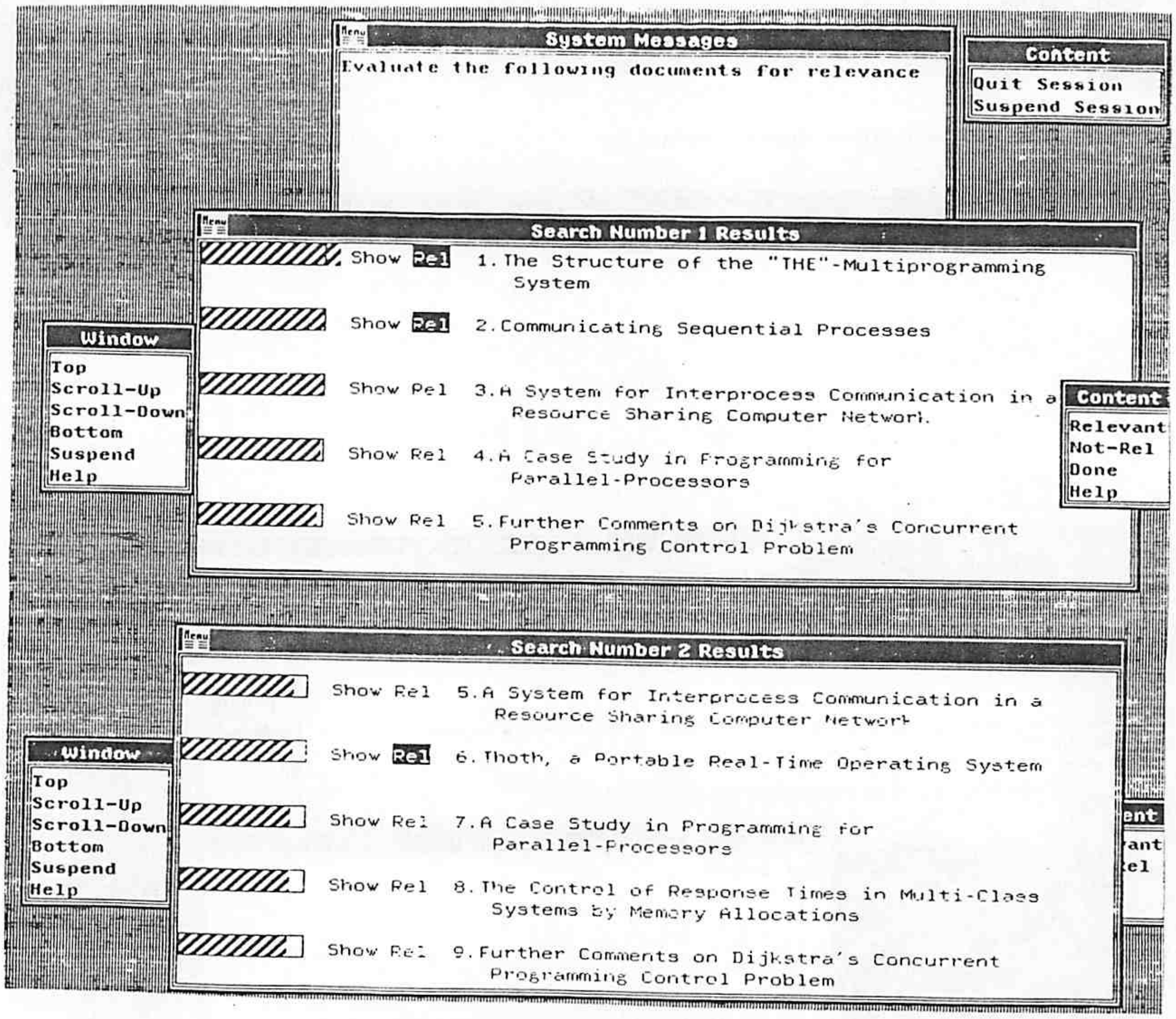
Yes No 2. A Document that you already know about

Yes No 3. A Boolean Query Formulation

User Query

I'm interested in mechanisms for communicating between disjoint processes, possibly, but not exclusively, in a distributed environment. I would rather see descriptions of complete mechanisms, with or without implementations, as opposed to theoretical work on the abstract problem. Remote procedure calls and message-passing are examples of my interests.

5(b)



System Messages

Evaluate the following documents for relevance

Search Number 1 Results

Show Rel 1. The Structure of the "THE"-Multiprogramming System

Show Rel 2. Communicating Sequential Processes

Show Rel 3. A System for Interprocess Communication in a Resource Sharing Computer Network.

Show Rel 4. A Case Study in Programming for Parallel-Processors

Show Rel 5. Further Comments on Dijkstra's Concurrent Programming Control Problem

Search Number 2 Results

Show Rel 5. A System for Interprocess Communication in a Resource Sharing Computer Network

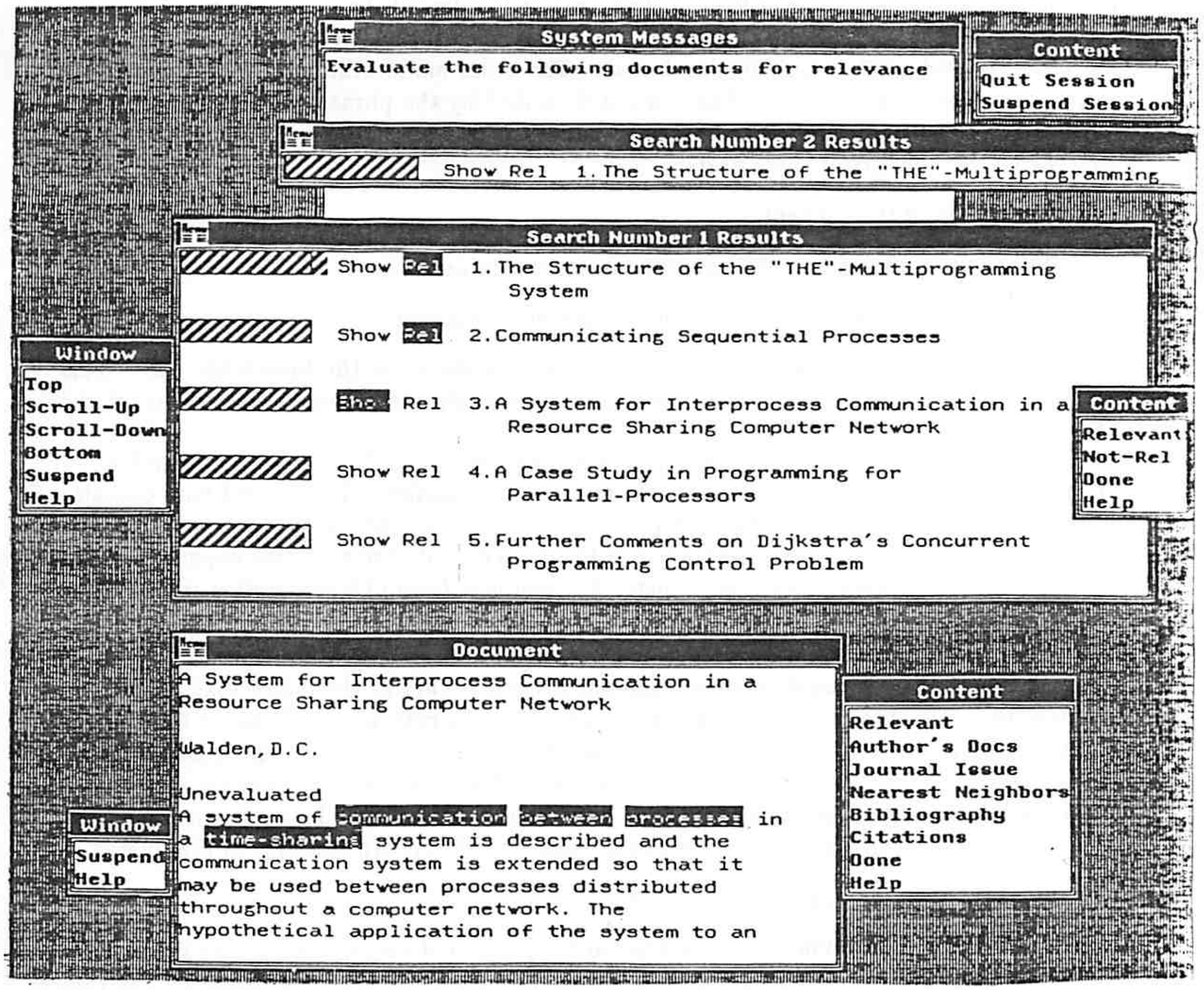
Show Rel 6. Thoth, a Portable Real-Time Operating System

Show Rel 7. A Case Study in Programming for Parallel-Processors

Show Rel 8. The Control of Response Times in Multi-Class Systems by Memory Allocations

Show Rel 9. Further Comments on Dijkstra's Concurrent Programming Control Problem

5(c)



Evaluate the following documents for relevance

Show Rel 1. The Structure of the "THE"-Multiprogramming

Show Rel 1. The Structure of the "THE"-Multiprogramming System

Show Rel 2. Communicating Sequential Processes

Show Rel 3. A System for Interprocess Communication in a Resource Sharing Computer Network

Show Rel 4. A Case Study in Programming for Parallel-Processors

Show Rel 5. Further Comments on Dijkstra's Concurrent Programming Control Problem

A System for Interprocess Communication in a Resource Sharing Computer Network

Walden, D.C.

Unevaluated
A system of communication between processes in a time-sharing system is described and the communication system is extended so that it may be used between processes distributed throughout a computer network. The hypothetical application of the system to an

5(d)

Figure 5: Screens from the example session

5 Domain Knowledge

The representation and use of domain knowledge is a significant part of I^3R [16]. The following is a summary of the main features. The domain knowledge represented in I^3R is very similar to the knowledge provided in a thesaurus. The novel features of this system are the inference process used to infer concepts from queries, the use that is made of those concepts, and the emphasis on acquiring knowledge from users. The same inference process could be applied to domain knowledge acquired from a thesaurus, if this knowledge were represented in the form described here. The definition of a concept is necessarily vague in that it represents an important "chunk" of a particular domain. The usual manifestation of a concept in text will be as a keyword or phrase. The concept described by the phrase "document retrieval", however, may actually appear in text in a variety of ways. The description of a domain concept in I^3R consists of three parts;

1. The name of the concept.

2. Information about how the concept can be recognized in text.

3. The relationships between the concept and other concepts.

These parts are contained in a *concept frame* which is stored in the knowledge base. This representation combines and generalizes aspects of the representations used by Shoval [24] and in the Rubric system [25].

The first part of a concept description is self-explanatory. The second and third parts are more complicated and are used by the inference mechanism. The second part contains recognition rules that describe the different ways in which a concept can be recognized in a text passage. These rules are different than those used to describe system experts and are used only by the domain knowledge expert. The simplest form of a recognition rule is

If `stem` *then* `Concept (C)`.

This rule recognizes `Concept` with certainty C if `stem` occurs in the query text. Certainty values express the degree of confidence the user has in the rule and are between 0 and 1. A certainty value can be thought of as a conditional probability, which in this case would be of the form P(`Concept|stem`). Recognition rules of this type are used in most single-word concepts and they form the main connection between the stem-oriented index terms generated in I^3R and the concepts represented in the domain knowledge. An example of this rule is

If `'informat'` *then* `Information (.9)`.

More complex rules involving stem endings can be specified. In general, there can be multiple rules for recognizing a concept specified in a concept frame. Another important type of rule refers to the presence of other concepts. For example,

If `Information` *and* `Retrieval` *then* `Information_Retrieval`.

Again, more complex forms of this rule can be specified.

The third part of the concept frame describes the relationships of a concept to other concepts. The types of relationship that are used are;

- Synonym.

- Generalization. This represents the broader/narrower term hierarchy found in a thesaurus.

- Instantiation. The concept is an instance of another concept. For example, Vax is an instance of Computer.

- Part-of. The concept is a component of another concept. This is not the same as the recognition rule that refers to other concepts. An example is that Processor is part of Computer.

- Cross-reference. This is a catch-all relationship to indicate that a concept has some similarity to another concept.

Each of these relationships is bi-directional. Note that recognition rules can also be thought of as defining a type of relationship. Figure 6 presents some domain knowledge represented as concept frames. In this figure, only some frames are shown in detail.

It is important to stress that the domain knowledge is intended to be derived mainly from interaction with the users. This means that during the *Get Background* stage of the session, the system enters into a dialogue with the user to verify concepts that have been inferred and to request more information on domains that are not well-specified in the current knowledge base. To facilitate the domain knowledge acquisition, the system will provide a simple interface for specifying concepts and the relationships between them. The system can also supply default certainty values. Domain knowledge acquisition is optional, but it is an important part of improving the effectiveness of retrieval. It is hoped that by emphasising the benefits, users will be encouraged to provide this information. Another point is that domain knowledge is acquired gradually through individual search sessions. This is a very different approach than trying to construct a thesaurus for an entire subject area before searching. Although the system can operate with no domain knowledge, it seems reasonable for a domain expert to "prime" the knowledge base with some key concepts. Research concerned with automatically identifying domain concepts is mentioned in the last section.

Given a particular query and assuming that some domain knowledge is available, the domain knowledge expert will attempt to infer concepts that are related to the query. The recognition rules and relationships in the concept frames define an AND/OR tree of concepts [23]. The AND nodes in the tree represent the recognition rules in that a concept represented by an AND node is recognized (or inferred) if each of the concepts represented by the children is recognized. More complex recognition rules put additional constraints on the recognition but every child node must be recognized. The OR nodes, which represent the relationship rules, can be inferred whenever any child concept is inferred. Figure 7 gives an example of the AND/OR tree for the concept frames of Figure 6. The inferences represented by the relationship rules are bi-directional and, therefore, the inferences only form a tree from the point of view of a particular concept. For example, if Computer_Vision was inferred from the concepts Computer and Vision, the concept Pattern_Recognition could be the next inference.

The certainty values associated with rules are used to derive certainty values for inferred concepts. These values are provided by users or by defaults that are related to the type of rule. For example, the default certainty for a synonym relationship is .9, for a cross-reference it is .6. Certainty values can be combined in a variety of ways [25]. If the certainty values are viewed as probabilities, then this combination can be difficult. A simpler approach results from treating the certainty values as fuzzy set membership values. There are two cases; inferences involving AND nodes and OR nodes. We shall illustrate the combination of certainty values using simple examples of these inferences. A simple rule that is associated with an AND node is

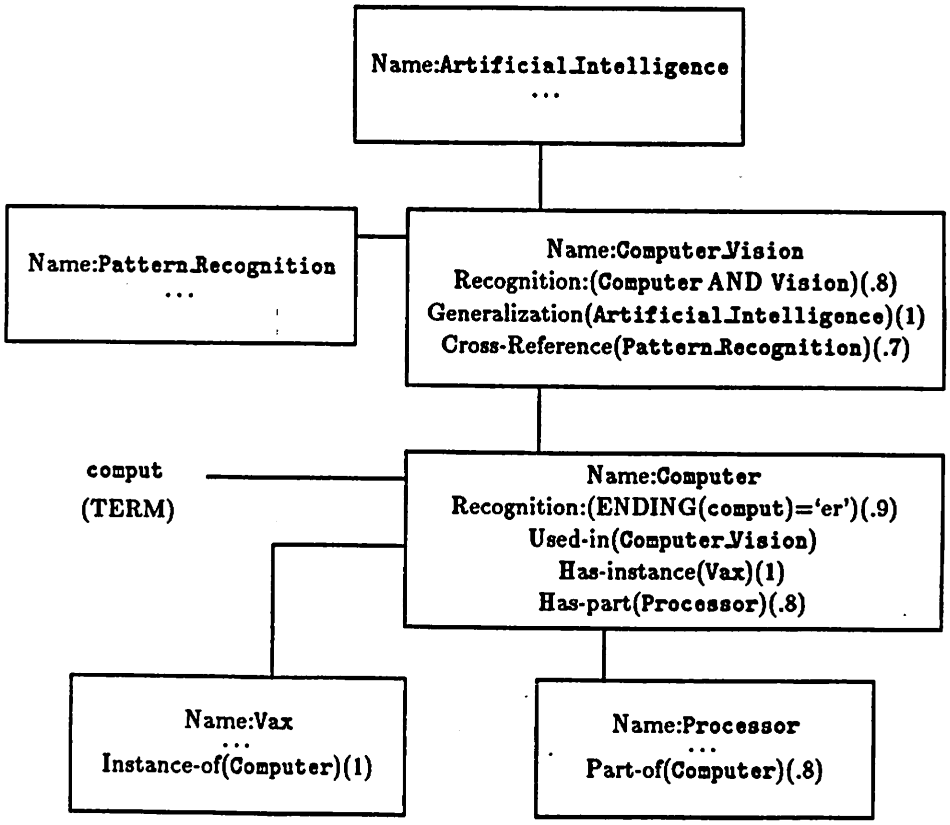


Figure 6: Domain knowledge as concept frames

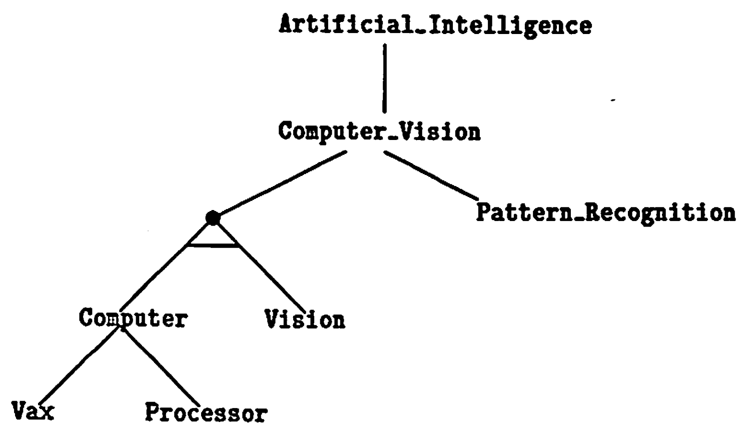


Figure 7: An AND/OR tree of concepts.

If A *and* B *then* C (X).

In this rule, A,B, and C are concepts and X is the certainty associated with the rule. A and B can have their own certainty values associated with them, represented as c(A) and c(B). The overall certainty associated with the inferred concept C is defined as

$$c(C) = \min[c(A), c(B)].X$$

For the corresponding OR rule, the certainty is

$$c(C) = \max[c(A), c(B)].X$$

The combination of certainty values when any number of concepts are involved in the inference takes the same form.

Using the AND/OR tree as a basis, we can describe the inference process used in I^3R. The process uses two lists; a list of the concepts inferred so far (concept-list) and a list of inferences that remain to be done (pending-list). These lists are data structures that are local to the domain knowledge expert. Each entry on the concept-list contains the name of the concept that has been inferred, the certainty value and the chain of inferences that was followed to that concept. The latter information is used to explain the inference of a concept to the user.

The first step in the inference process is to use the stems from the query text to locate the concept frames that can be directly recognized from them. This set of concepts is placed on the concept-list. The domain knowledge expert then tries to infer all concepts that are directly related to this initial set. In terms of the AND/OR tree, this means that the system attempts to move one level up the tree from the initial nodes (which are usually terminal nodes).

The first inferences that are done are those involving the OR nodes (concept relationships). These inferences can be made directly and are put on the concept list. The only relationship types that are used at this stage are synonym, cross-reference and instantiation in the direction of the general concept. As each concept frame on the initial concept list is examined, the inferences that involve AND nodes are put on the pending-list. From the example in Figure 7, if `Computer` was a concept established from the query, the inferences involving `Vax` and `Processor` are *not* done and the inference involving `Computer_Vision` is put on the pending-list.

After examining each of the initial set of concepts, the Domain Knowledge Expert attempts to infer the concepts on the pending-list. This involves looking at the child concepts of the AND node and attempting to infer them. This process of working backwards from a goal concept is called backward chaining. By combining forward and backward chaining, the system directs the inference process towards establishing the concepts that are closely related to the query. In our example, the domain knowledge expert would attempt to infer `Computer_Vision` by inferring `Computer` and `Vision`. `Computer` is present on the concept-list and has been inferred. The `Vision` concept frame is then examined to find what concepts can be used to infer `Vision`.

After attempting to infer the concepts on the pending-list, the system presents the concept-list to the user in order of decreasing certainty (and eliminates duplicates). A threshold value can be used to prevent showing concepts that are not strongly related to the query. The concepts identified as being of interest, together with their certainty values, are added to the request model and used in subsequent searches with the probabilistic retrieval algorithm. If the user requires more suggestions or if the system considers that not enough concepts have been identified, the inference process can be carried further.

6 Browser

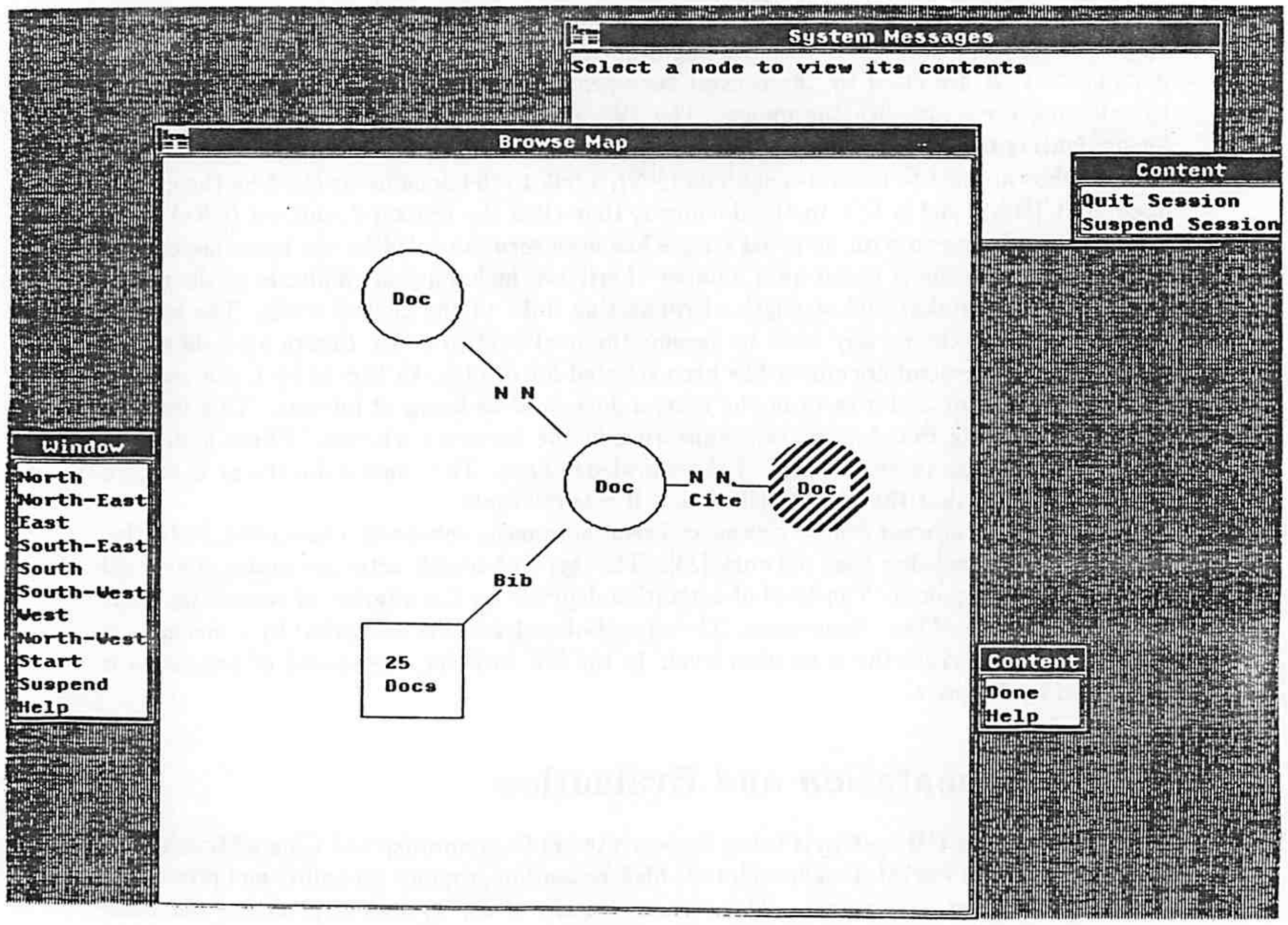
The browsing expert provides the user with a means of navigating through the knowledge base from given entry points. While browsing, the user identifies documents, terms or concepts that can be used to refine the request model. The knowledge base can be displayed graphically as a network of nodes and links. Nodes represent entities such as documents and links represent relationships between the entities.

A browsing window displays the network in the neighborhood of a central node. Figure 8(a) shows a browsing window containing 3 document nodes and a square representing 25 documents that are cited by the central document. The contents of a node are examined by selecting the node with the mouse. The "Window" menu provides direction commands for navigating through the knowledge base without examining node contents. Three types of link are shown; links to nearest-neighbors (NN), a link to the documents cited by the central document (Bib), and a link to the document that cites the central document (Cite). The document node shown with diagonal stripes has been recommended by the browsing expert. The recommendation is based on a number of criteria, including an emphasis on document nodes and the number and strength of connecting links to the central node. The user is, of course, free to choose any node to become the next central node. Figure 5(b) shows the screen after the central document has been selected for display. In Figure 5(c), the user has selected the author and a term in the central document as being of interest. This leads to nodes representing that information appearing in the browsing window. These nodes can then be used to go to other parts of the knowledge base. The central document is shaded black to indicate that the user specified that it was relevant.

The browsing process can be viewed as a semi-automatic version of a spreading activation search of the knowledge base network [23]. This type of search activates nodes connected to given starting points. The level of activation depends on the number of connecting links and the strengths of the connections. The spread of activation is controlled by a mechanism such as a threshold on the activation level. In the I^{3}R browser, the spread of activation is controlled by the user.

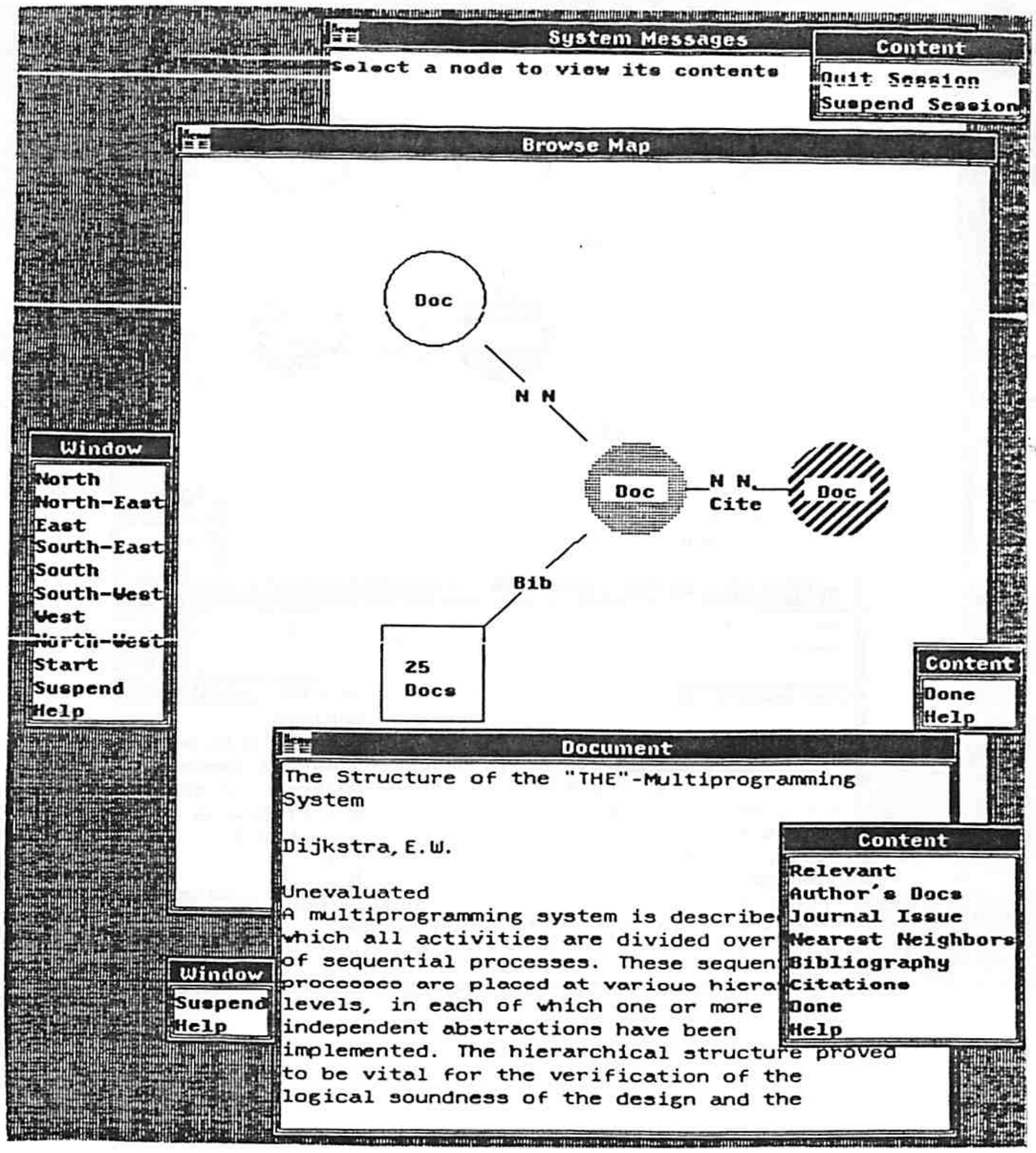
7 Implementation and Evaluation

A prototype of the I^{3}R system is being implemented in CommonLisp and C on a MicroVax-II workstation. This workstation provides the high-resolution graphics capability and processing power required for the system. The current version of the system implements the basic architecture of I^{3}R as shown in Figure 2. Apart from the search controller, the experts are currently very simple. The allows a complete search session to be carried out, but the options available at each stage of the session are limited. The retrieval algorithms have been implemented in C and the indexing algorithms in Lisp. The current system contains over 200 CommonLisp functions and 200 lines of C. The knowledge base for the CACM collection of documents [7] has been partially implemented using the RDB relational database system [26]. The database system provides a flexible means of storing the statistical part of the knowledge base and algorithms for the probabilistic search, the cluster search and nearest neighbor generation have been designed to take advantage of the access methods provided [6]. RDB also supports the storage of variable-length text fields such as the abstract. Currently, only simple domain knowledge is stored in the database. In particular, synonym and cross-reference relationships together with the associated certainty values are stored as relations. Recognition rules are not stored. The requirements for representing domain concept frames



Select a node to view its contents

8(a)



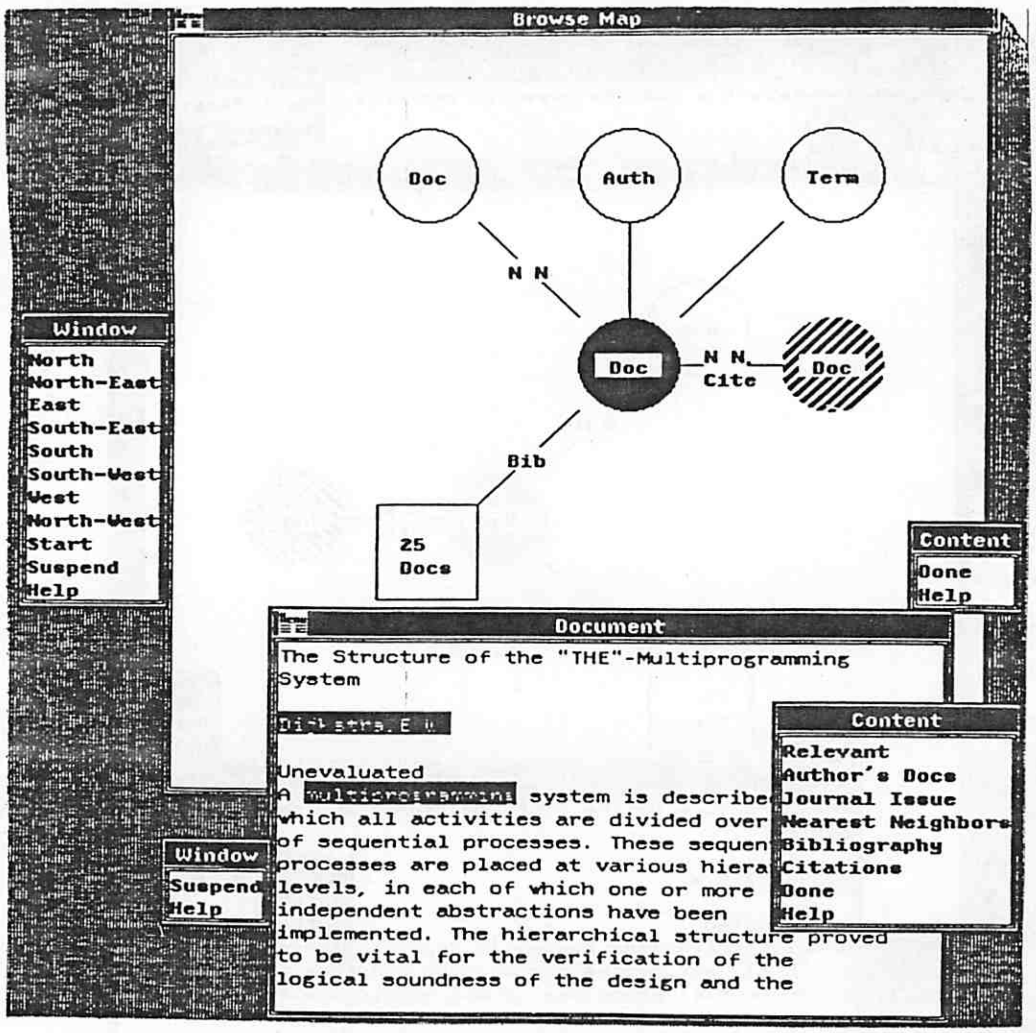
Select a node to view its contents

The Structure of the "THE"-Multiprogramming System

Dijkstra, E.W.

Unevaluated
A multiprogramming system is describe
which all activities are divided over
of sequential processes. These sequen
processes are placed at various hiera
levels, in each of which one or more
independent abstractions have been
implemented. The hierarchical structure proved
to be vital for the verification of the
logical soundness of the design and the

8(b)



The Structure of the "THE"-Multiprogramming
System

Dijkstra, E. W.

Unevaluated
A multiprogramming system is describe
which all activities are divided over
of sequential processes. These sequen
processes are placed at various hiera
levels, in each of which one or more
independent abstractions have been
implemented. The hierarchical structure proved
to be vital for the verification of the
logical soundness of the design and the

8(c)

Figure 8: A browsing display.

and efficiency problems with the current database implementation (response times for queries vary from 45 seconds to 4 minutes) have led us to consider implementing the next version of the knowledge base using a frame-based representation language.

A major issue in the development of a new system such as I^3R is evaluation. Conventional retrieval systems are based on a single search strategy that can be compared to other search strategies using effectiveness measures such as recall and precision. The situation for I^3R is complicated by the highly interactive and flexible nature of a search session. Even comparing just one facility, the browser, to conventional search strategies is a difficult task [19]. Another complication is that the current state of the prototype system prevents us from undertaking user studies.

The evaluation of the retrieval strategies used in I^3R has, however, already been done in previous research. The formal strategies used have been tested extensively. The results have shown them to be effective and, given additional information in the request model, the effectiveness definitely improves. Relevance feedback, another well-tested and effective technique, is incorporated in the system. In other words, the basis for the simple search sessions supported by the current I^3R system has been shown to be the most effective way of using statistical strategies. The parts of the system that have not been adequately tested involve the domain knowledge expert and the browser. In particular, the levels of improvement that can be obtained using concepts found by the DKE have not been established. Salton's experiments with a thesaurus in which he obtained significant retrieval improvements indicate the type of results that can be expected [27]. Some evidence also exists that browsing is a useful tool to provide to users of a document retrieval system.

The I^3R system can place more demands on the users than a conventional system. It can, however, provide more assistance and explanation than a conventional system. The effectiveness of the system (i.e. the underlying retrieval strategies) depend heavily on the amount of effort made by the users in specifying their information needs (the Quality-In Quality-Out Principle). It should be remembered that, at worst, the system will function in exactly the same way as a system that simply takes a query and retrieves documents. It is our opinion that users who are prepared to discuss their information needs with intermediaries will be willing to provide I^3R with the domain knowledge that would lead to improved retrieval performance.

8 Future Research

The main area of future research involves the use of NLP techniques. In particular, we are interested in techniques for identifying concepts in text. An obvious use of these techniques is to identify concepts in query texts and some work has been done in this area. For example, Sparck Jones and Tait [9] built a system that identified concepts in query texts and then generated variants that were used for searching. A facility for indentifying concepts using NLP could be incorporated into the domain knowledge expert.

The document texts of titles and abstracts present different challenges. In the current system, the concepts in the request model are matched with the document texts only through the index terms derived from them. A better approach would be to use NLP to infer that a given query concept occurs in a document text. The matching or inference is done at the concept rather than the single word level. This detailed matching process would, however, probably only be used on the documents already identified by the statistical strategies as potentially relevant. This would significantly reduce the amount of processing required.

The most difficult task is to identify concepts in new document texts that can be used for indexing. The document representatives would then contain concepts and single words derived from those concepts. It is important to keep some form of simple indexing because individual users may specify concepts in different ways. Statistical retrieval techniques can be used with concepts as well as words and the documents retrieved could still be analyzed in more depth. The principle advantages of this approach are the potential reduction in the size of document representatives and the identification of concepts that may be included in the domain knowledge. This is the general approach of some indexing techniques based on the identification of important syntactic units [17,18].

Many difficult research problems remain to be solved in order to apply NLP to document retrieval. One of the major objections has been the amount of lexical or dictionary information required for individual words in order to do effective analysis. This objection appears to have been addressed by recent research on the processing of machine-readable dictionaries [28]. The limited nature of the document retrieval task, the fact that there are ways to limit the role of NLP in these systems, and the continued progress in NLP research are all reasons to be optimistic about the use of NLP for document retrieval. Our aim is to use the I^{3}R system as a vehicle for this research.

Acknowledgments

This research was supported by NSF grant IST-8414486. Additional facilities were provided by University College, Dublin where the first author was on leave. The referees provided valuable comments.

References

[1] Date, C.J. *An Introduction to Database Systems.* Third Edition. Massachusetts: Addison-Wesley; 1981.

[2] Christodoulakis, S. "Office filing" in *Office Automation*, ed. D. Tsichritzis. Berlin: Springer-Verlag; 1985.

[3] Stonebraker, M. *et al.* "Document processing in a relational database system." *ACM Transactions on Office Information Systems.* 1:143-158; 1983.

[4] Salton, G.; McGill, M. *Introduction to Modern Information Retrieval.* New York: McGraw-Hill; 1983.

[5] Van Rijsbergen, C.J. *Information Retrieval.* Second Edition. London: Butterworths; 1979.

[6] Croft, W,B.; Parenty,T. "A comparison of a network structure and a database system used for document retrieval." *Information Systems.* 10:377-390; 1985.

[7] Salton, G.; Fox, E.A.; Wu, H. "Extended Boolean information retrieval." *Communications of the ACM.* 26:1022-1036; 1983.

[8] Lebowitz, M. "Intelligent information systems." *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR Forum.* 17:25-30; 1983.

[9] Sparck Jones, K.; Tait, J.I. "Automatic search term variant generation." *Journal of Documentation.* 40:50-66; 1984.

[10] Walker, D.E. "Natural language access systems and the organization and use of information." *Proceedings of the Ninth International Conference on Computational Linguistics, COLING-82.* Amsterdam: North-Holland; 1982, 407-412.

[11] Marcus, R.S. "An automated expert assistant for information retrieval in the information community." *Proceedings of the 44th ASIS Annual Meeting.* 1981, 270-273.

[12] Brajnik, G.; Guida, G.; Tasso, C. "An expert interface for effective man-machine interaction." in *Cooperative Interactive Systems*, Ed. L. Bolc. Springer-Verlag; 1985.

[13] Erman, L.D.; Hayes-Roth, F.; Lesser, V.R.; Reddy, D.R. "The Hearsay-II speech understanding system: Integrating knowledge to resolve uncertainty." *ACM Computing Surveys.* 12:213-253; 1980.

[14] Van Rijsbergen, C.J. "A non-classical logic for information retrieval." *Proceedings of the International Conference on Research and Development in Information Retrieval*, Pisa (to appear).

[15] Croft, W.B. "Boolean queries and term dependencies in probabilistic retrieval models." *Journal of the American Society for Information Science.* 37:71-77; 1986.

[16] Croft, W.B. "User-specified domain knowledge for document retrieval." *Proceedings of the International Conference on Research and Development in Information Retrieval*, Pisa (to appear).

[17] Klingbiel, P.H. "A technique for machine-aided indexing." *Information Processing and Management.* 9:477-494; 1973.

[18] Dillon, M.; Gray, A.S. "FASIT: A fully automatic syntactically based indexing system." *Journal of the American Society for Information Science.* 34:99-108; 1983.

[19] Oddy, R.N. "Information retrieval through man-machine dialogue." *Journal of Documentation.* 33:1-14; 1977.

[20] Belkin, N.J.; Hennings, R.D.; Seeger, T. "Simulation of a distributed expert-based information provision mechanism." *Information Technology.* 3:122-141; 1984.

[21] Brooks, H.M.; Daniels, P.J.; Belkin, N.J. "Using problem structures for driving human-computer dialogues." *Proceedings of RIAO-85, Recherche d'Informations Assistee par Ordinateur.* Grenoble; 1985, 645-660.

[22] Rich, E. "Building and exploiting user models." Ph.D. Thesis, Carnegie-Mellon University, Technical Report CMU-CS-79-119, 1979.

[23] Charniak, E.; McDermott, D. *Introduction to Artificial Intelligence.* Massachusetts: Addison-Wesley; 1985.

[24] Shoval, P. "Expert/Consultation system for a retrieval database with semantic network of concepts." *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR Forum.* 16:145-149; 1981.