Development of the CODER System: A Test-bed for Artificial Intelligence Methods in Information Retrieval

Edward A. Fox

TR-86-40

December 1986

Preprint of paper to appear in
"Information Processing and Management," 23(4), 1987.

Development of the CODER System: A Test-bed for Artificial Intelligence Methods in Information Retrieval †

Edward A. Fox

Department of Computer Science
Virginia Tech, Blacksburg VA 24061

ABSTRACT

The CODER (COmposite Document Expert/Extended/Effective Retrieval) system is a test-bed for investigating the application of artificial intelligence methods to increase the effectiveness of information retrieval systems. Particular attention is being given to analysis and representation of heterogeneous documents, such as electronic mail digests or messages, which vary widely in style, length, topic, and structure. Since handling passages of various types in these collections is difficult even for experimental systems like SMART, it is necessary to turn to other techniques being explored by information retrieval and artificial intelligence researchers. The CODER system architecture involves communities of experts around active blackboards, accessing knowledge bases that describe users, documents, or lexical items of various types. Most of the lexical knowledge base construction work is now complete, and experts for search and temporal reasoning can perform a variety of processing tasks. User information and queries are being gathered, and the first prototype is nearly complete. It appears that a number of artificial intelligence techniques are needed to best handle such common, but complex, document analysis and retrieval tasks.

CR Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; H.3.1 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software; H.4.3 [**Information Systems Applications**]: Communications Applications – *electronic mail*; I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems;

General Terms: Algorithms, Design

Additional Keywords and Phrases: blackboard, CODER project, composite documents, computer based message systems, document analysis, expert systems, knowledge base, Prolog

†Project funded in part by grants from the National Science Foundation (IST-8418877) and the Virginia Center for Information Technology (INF-85-016), and aided by an AT&T Equipment Donation.

1. Introduction

Online searching of bibliographic databases, originally an aid to scholarly activities, has become increasingly important in business and government as well. With the spread of word processing and electronic publishing, a greater proportion of written materials now being produced is available in machine readable form. The advent of full-text databases [TENO 84], originally important primarily for legal research, has helped push the total number of publicly accessible online databases above the three thousand mark. If one includes corporate and private text collections, such as can develop from office information systems with a text filing capability, tens of thousands of searchable collections already exist.

1.1 End user searching

Machine aided searching of early databases began during the 1950's, but was a cumbersome and expensive process. Today, with far-flung networks connected to mainframe computer systems that manage vast banks of online storage, or with powerful microcomputers controlling high capacity CD ROM (compact disc read only memory) optical drives, individuals have the hardware tools to perform their own searches, but still only have fairly primitive software support.

Even in the domain of bibliographic retrieval, many end users prefer to locate interesting items without having to involve a search intermediary [OJAL 86]. Not every user feels this need, but many search intermediaries also desire a simpler means of access to the multiple systems and databases involved. Furthermore, in the context of office automation, individual office workers usually **must** search on their own. Even though there are now commercially available systems aimed at making searching easier, there is still no truly helpful search methodology that can effectively meet the needs of end users [WILL 85].

1.2 Need for improvement in information retrieval systems

One recent study confirmed earlier findings that effectiveness and efficiency of searching by a single individual is surprisingly low [TRIV 86]. Another study showed the overlap between search results of different people to be small [KATZ 82]. Since it is typically not possible to follow the obvious suggestion derived from these studies, namely to have several different searchers work on the same interest statement and then pool their results, it seems appropriate to consider a different approach, such as making a computer help play the role of several (intelligent) searchers.

The opportunity of discovering how to dramatically improve retrieval effectiveness has challenged the body of researchers working on automatic indexing and retrieval systems, and has led to development of a variety of methods based primarily on statistical and probabilistic processing of text collections and user queries. While marrying these methods with powerful microcomputers and optical stores is now feasible and would benefit many users [FOXE 86b], some researchers feel that more "intelligent" approaches are needed to provide even greater effectiveness [VANR 86]. Since artificial intelligence methods have begun to provide assistance in a variety of other complex tasks, the information retrieval problem is being re-formulated as one involving "knowledge" bases [BISW 85].

1.3 Prolog for handling knowledge

The Japanese fifth generation effort has popularized the value of Prolog as an AI language for handling knowledge of various types. Database researchers find it appealing to bring together the flexibility and expressive power of Prolog and the efficiency of database management systems [SCIO 86]. Prolog can be easily learned [CLOC 84] and can be applied to a variety of problems [KOWA 79]. Advanced texts on Prolog programming are now available, so the elegance of logic programming and the efficiency and metamathematical expressiveness of Prolog can both be properly unleashed [STER 86].

Prolog typically manipulates integers, characters, and atoms (i.e., character strings that "name" some entity). Lists (e.g., [1,2,3]) and structures (e.g., "owns(tom, car)") are built up from simpler objects. Predicates (as in first order logic) are in the form of structures, where the relation head (or "functor" such as "owns" above) describes a relationship among the various arguments, whose number ("arity") is fixed. The simplest Prolog statement ("clause") is in the form of a fact, like "father(adam, cain)." Rules are more complex clauses, whose power derives in part from use of variables. Thus, to define the grandfather relationship one could state

```
grandfather( Grandpa, Grandchild ):- father( Grandpa, Child ),
                                     parent( Child, Grandchild ).
```

Then, by asking a question like

```
?- grandfather( adam, Grandchild ).
```

one requests a constructive proof and so can cause Prolog to (successively) generate each grandchild's name. Pattern matching ("unification"), recursion, automatic backtracking, searching through sets of facts or rules, and list manipulation are additional features of Prolog that allow it to be easily adapted to handle lists of keywords, perform natural language parsing, manage a relational database, or inference about complex knowledge structures.

1.4 CODER

The CODER (COmposite Document Expert/Extended/Effective Retrieval) project was proposed as a means of investigating the use of logic programming methods in general, and Prolog in particular, for handling the complex task of information analysis and retrieval of composite documents [FOXE 85]. A version of Prolog with a built-in database capability, MU-Prolog, was selected [NAIS 85]. Before explaining about the CODER system, however, it is appropriate to

explore the class of problems that were to be addressed, and the information retrieval and artificial intelligence research that relates.

2. Problem

During the last two decades the communication and computing technologies supporting computer based message systems have matured to the point that most large computer systems, and many small systems, are involved in some type of networking. Thousands of computers are on far flung networks such as ARPANET, BITNET, MILNET, NSFNET, or USENET [QUAR 86]. Some support remote logging on of users, or conferencing, but nearly all handle electronic mail.

2.1 Network mail

Following the lead of standards groups like the National Bureau of Standards and IFIP Working Group 6.5, CCITT developed and approved the X.400 Message Handling Standard for later submission to the International Standards Organization [MYER 83]. In terms of the established ISO/OSI framework for open systems interconnection, X.400 dealt with the topmost or application layer. As can be seen in Figure 1, users invoke the message handling system (MHS) by talking to a user agent (UA). The UA in turn communicates with the message transport system (MTS), which can connect together computers across the globe. Ultimately, one user's message follows the chain of UA-MTA-...-MTA-UA so that another user can receive it.

<Have Figure 1 around this point>

The X.400 model allows user agents to carry out other related tasks. Since many UAs will be part of office information systems, it seems appropriate for them to perform additional functions such as filing and retrieval of recent or archived messages or other office objects [CROF 82]. But

since any text file (or other file, such as an image or voice segment) can be sent as mail, it is necessary to understand the semantics of objects which are to be indexed, filed, and retrieved. Of particular interest are relationships of inclusion or reference among messages [BABA 85]. Clearly, there is a need for information retrieval support of mail handling systems, and that must include the ability to analyze the document structure and type [FOXE 86d].

2.2 AI message examples

Figure 2 illustrates some of the types of documents and document components present in messages sent out over the DARPA Internet as part of recent issues of AIList Digest. An archive of these messages was selected as the test collection for CODER.

<Have Figure 2 around this point>

Each (brief) example begins with the initial part of the body of a real message, and includes a minimal number of interesting later portions separated by "..." lines. Example A gives a bibliographic citation and an abstract that can only be identified by their format and indenting. Examples B, C, and D have references to messages, but use several different devices. Example E uses spacing and lines to separate the title, sections, and lists provided. A memo style seminar announcement is shown in example F, and a memo style is again used in part of the survey of example G. Example H uses a memo style, centering, and capitalization in a Call for Papers. A special bibliographic form is used in example I, while a different structure is used in the report list part of example J. A standard address form begins example K, but the list included is structured with separator lines, spacing, and left justified entries in a fixed order. Finally, example L shows a different address form, and uses block centering to separate the title and abstract.

While context free grammars are often used for parsing, it may be more appropriate to use more powerful notations like that of [KIMU 84]. After parsing, it is necessary to index: entire documents, parts of documents, and even low level structures like the bibliographic citation at the

top of example A of Figure 2. In the context of probabilistic retrieval, a scheme has been proposed for adapting to different components [KWOK 86]. Some of the passage retrieval techniques mentioned in [OCON 80] may also be applicable.

2.3 Limitations of SMART Handling

In the vector space retrieval model, each document is represented by a list of concepts and weights. This perspective has provided insight for a variety of studies undertaken with the SMART system [SALT 80]. SMART was extended in 1982 so that several kinds of concepts could be used to represent the different aspects of composite documents; instead of a single vector, multiple subvectors were allowed [FOXE 83b]. Based on the notion of multiple concept types, an AIList digest message can be indexed or inquired about according to the scheme shown in Figure 3.

<Have Figure 3 around this point>

The digest header is represented by concept types 0-2, parts of the message header are represented using concept types 3-5, and the subject line and body of the message are represented with type 6. The result of a document pre-parsing phase applied to the message used in the last example of Figure 2 is shown in Figure 4. While the heading is analyzed in a reasonable fashion, all structure present in the body of the message is lost by the current analysis scheme used in SMART. Though parsing the message body prior to SMART processing is possible, there is no easy way to represent the relationship among document components. If one large vector is built, the structural relationships are lost. On the other hand, if a different vector is build for each component, it is awkward in a vector scheme to relate the various vectors so that, for example, characteristics of an object are inherited by another object that is part of the first.

<Have Figure 4 around this point>

All in all, while SMART has served as a test-bed for statistical retrieval methods, it cannot easily be

adapted to the AIList Digest collection. It is hoped that CODER will aid in such processing and become a test-bed for artificial intelligence applications to information retrieval.

3. Related Work

In designing CODER so as to solve the problem of handling composite documents, a variety of related efforts were surveyed. Since CODER is to be a flexible test-bed, many of the best features of other research systems are or can be included. Of particular interest are efforts in information retrieval, artificial intelligence. Furthermore, it is especially useful to consider similar systems.

3.1 Information retrieval research

First, it was clear that CODER should be able to combine evidence of various sorts when comparing an interest statement representation with a document representation. In [BICH 80] the value of this approach was shown for using bibliographic coupling and cocitations. That work was extended in [FOXE 83a] and [FOXE 84a] to consider terms, bibliographic coupling, cocitations, direct citations, author names, etc., and again led to improvements beyond using terms alone.

Second, it was clear that a thesaurus would be of value. It is not obvious, however, how to (semi)automatically build a good thesaurus for a particular document collection, or how to use it effectively later [SVEN 86]. In the interest of generality it was suggested that a machine readable dictionary [AMSL 84] be used as a starting point. An initial experiment showed a small improvement in performance when queries were expanded by having terms added that were lexically or semantically related to low frequency terms in the original query. These results were confirmed in [WANG 85] and [EVEN 85], and replicated again in the context of expanding

extended Boolean queries [FOXE 84b]. Starting with these lexical/semantic relations, a more comprehensive lexicon could be produced [EVENS 79], and one is being developed by Evens et al. In [WILL 86] it was noted that associative aids are easy and effective to use in retrieval; such help could easily be obtained from this kind of lexicon. Indeed, there are many uses of such lexicons [KUCE 85], including to aid in natural language processing. [WALK 85] also illustrates the value of lexical and other knowledge resources.

Third, there have been a variety of retrieval algorithms where advanced search techniques and other heuristics should be of benefit. [BUCK 85] explores this in the context of improving efficiency by limiting the processing of an inverted file. [BOOK 83] suggests viewing probabilistic feedback as a sequential learning process, which might be implemented using heuristics so that documents are actually retrieved one by one with a reduced computational load at each step.

Fourth, several systems gave users other opportunities to interact beside supplying a Boolean query [ODDY 77]. CALIBAN made use of high quality raster displays with windows [FREI 84]. In [KORF 86] it was suggested that users could navigate through a document vector space. [MILL 85] also suggested browsing, but between words and word senses instead of documents.

Finally, it has become clear that to service end users it is important to understand more about how they interact with retrieval systems. Findings regarding human-computer interaction can be applied [BORG 84] to help with designing systems for easier use [BORG 85]. [BATE 86] suggests that the overall process must be considered, so that phases like "finding the barn door" and "docking" into a close connection are identified, and a more effective design is developed than what is currently employed. [DANI 86b] considers user modeling in the light of developments in cognitive science. Studies of human-human interaction suggest characteristics of the user modeling function [DANI 86a] in the overall context of human-computer problem solving [DANI 85].

Ultimately these findings can be used to develop an intelligent intermediary that emulates the behavior of a human aid [BROO 85].

3.2 Artificial intelligence research

Linguistics has long been thought to be of value for information retrieval [SPAR 73]. It is not clear, however, exactly what role should be played by each of the broad types of processing that are possible [KORF 84]. Though Prolog can be used to parse large subsets of natural language and to build parse trees and other structures [PERE 83], matching of such structures is problematic. On the other hand, conversion of text descriptions to more structured forms does seem particularly useful [SOMM 85]. Resolving anaphoric references might be of value, but that effect has still to be demonstrated [KATZ 86]. Some improvement may be possible when queries are parsed to obtain phrases that appear in documents [SPAR 84].

At the heart of artificial intelligence is the representation of knowledge. Many schemes have been proposed and several have clear application to information retrieval. Frames are useful for representing objects [FIKE 85], and can be manipulated in any one of several frame languages that thus support a variety of applications [FOXM 86]. Rules are probably the easiest form to work with, can be coded by experienced programmers in Prolog or in other special languages, and fit in well with expert systems [HAYE 85]. Networks are helpful in situations where multiple associations exist. Thus, [SIMM 83] describes early work with hand coding of propositional knowledge from the first fifty pages of the *Handbook of Artificial Intelligence* into semantic networks.

Since many retrieval questions ask for documents appearing during a known time interval or that relate to events occurring during a particular interval, it is important to have some temporal reasoning capability. [ALLE 83] presents an interval based temporal logic with efficiency that can

be improved through the user of a hierarchy of reference intervals. [MAYS 86] deals with temporal reasoning in a natural language front end to a database system. [ZARR 83] indicates how to apply temporal representation and reasoning to the analysis and retrieval of biographical information about French historical figures.

Expert systems are being constructed for a wide variety of tasks. To simplify development, it has been suggested that generic experts with special languages be developed for the small number of common tasks, such as classification or abductive assembly [CHAN 86]. Prolog can be used to develop expert systems if one is experienced with the language [HELM 85]. Uncertainty can be represented by adding confidence values to all parts of all rules [LEEN 86] and by having appropriate computation routines. Alternatively, a meta-interpreter can be developed that hides this processing and so can readily switch to a different reasoning under uncertainty method [LECO 86]. This flexibility is necessary since there are many different schemes, each with their own advocates [CHEE 85].

To simplify the construction of expert systems in very complex domains, it is convenient to have a central blackboard with a variety of areas that can be examined and written to by experts. Though blackboards were first used to help with speech recognition [ERMA 80], they have been used for many different applications [NIIH 86b]. The theory and practice of blackboard use are carefully discussed in [NIIH 86a].

3.3 Related systems

Several different systems relate to particular aspects of CODER. The FRUMP system could skim newspaper stories, by filling in sketchy scripts (which are similar to frames) [DEJO 82]. Its speed, which will be needed if CODER is to process moderate size text collections, is achieved because only a partial parse is done. As an extension of FRUMP, the FERRET system has been proposed, to be applied to electronic mail messages [MAUL 86]. However, the primary aim is to

demonstrate feasibility of the sketchy script approach to this type of information analysis and retrieval. TOPIC is another system where document analysis is involved, carried out by a word expert parser that helps determine the topical structure of a text. In all of these systems, there is no notion of test-bed, user modeling, or of applying expertise to searching.

ARGON and RUBRIC are artificial intelligence systems applied to information retrieval. ARGON uses frames in a computationally efficient manner to handle classification of objects and queries, and determines matches according to frame subsumption hierarchies [PATE 84]. While this capability is valuable to CODER, it need not be the only representation, classification, and matching scheme allowed. RUBRIC provides special tools for users to develop comprehensive queries that are in the form of tree structured knowledge bases [TONG 86]. At the base of a tree are terms connected with Boolean and proximity operators that can be matched against documents. Given a scheme for rule based inferencing under uncertainty scheme, the system can compute a certainty value indicating how the query can be inferred from the document.

In their work on knowledge assisted document retrieval, Biswas et al. consider both the natural language interface and the retrieval components. They have a modular structure and plan to carry out a variety of experiments with the system. The natural language interface can handle a restricted sub-language through its augmented transition network, and determines the number of documents desired, the time range of interest, and the subject matter or content [BISW 86a]. The retrieval component uses fuzzy set theory and one of several combination of evidence schemes, which are also of interest in CODER [BISW 86b]. Yet there is little in the way of document analysis, user modeling, or other expertise.

IRES is another intelligent retrieval system, with natural language query processing and user modeling capabilities [DEFU 85]. There is emphasis on an independent thesaurus and on using an expert system. After morphological and syntactic query analysis, the indexing terms are accessed, results are combined and assigned values, and re-formulation takes place as needed. Though

adaptable to users and state of processing, there is no blackboard, no document analyzer, and little reported emphasis on system evaluation.

Project Minstrel is a much broader effort than CODER, aimed at office information systems. Yet its use of knowledge representation schemes and special query forms is relevant to CODER. More closely related, however, is I^3R, which is a blackboard based system built as a community of experts [THOM 85]. It is being developed on one computer, in LISP, with an interface to a separate database system. There is a user model and other aspects are close to the scheme suggested by Belkin, Brooks, and Daniels.

4. Approach

CODER is to serve as a test-bed for artificial intelligence (AI) methods in information retrieval. For example, logic programming methods are involved through the user of MuProlog with its built-in database support. Expert systems are present in both the document analysis and the retrieval communities of experts. Natural language processing is supported through the lexicon, and can help with query and document analysis. Knowledge representation is supported by a knowledge administration complex which allows creation and manipulation of types for elementary objects, frames, and relations. Searching is involved in providing rapid feedback in the event of any clues, and in applying heuristics to make the process more efficient. Planning is embodied in the strategist which deals with handling resources on multiple computers. Temporal reasoning is involved for both documents and queries. User modeling relates to the human-computer interaction aspects, to dealing with term expansion, and to controlling the retrieval process.

CODER has a number of distinguishing characteristics. To serve as a test-bed, it is modular so as to be adaptable to different theories, and can be experimented with by controlling which

modules are changed. CODER is a stand alone system, to manipulate raw documents and communicate directly with users, without an external database or indexing facility. CODER is comprehensive: in dealing with document analysis that yields vector and AI representation schemes; in building and applying knowledge about users; in supporting natural language processing with a large lexicon; in handling documents or passages; and in applying expertise to search and other tasks. CODER is designed from the ground up to operate on multiple computers and to benefit from whatever level of concurrency is achievable. Finally, CODER is unique in focussing on composite documents, including their structure, basic data types, and interrelationships.

4.1 Evolution of CODER project

Originally proposed in 1984, the CODER effort began with studies of different document components and with methods for combining them [FOXE 85]. Some 1000 documents were examined to develop heuristics for determining document types. The lexicon was begun, and MuProlog was chosen as the main programming language (to be supplemented with C whenever necessary). Issues relating to the design are discussed in [FOXE 86c]. Encouraged by the findings of [BELK 84], a blackboard orientation was planned. Students in information retrieval courses in AY 85/86 were assigned small parts of the system. Specifications were completed and collected together in [FRAN 86a]. [FOXE 87a] provides an overview, and highlights the system architecture.

4.2 Blackboard based development

As mentioned, a blackboard based design was chosen, to provide maximum flexibility and to ease integration of components. A blackboard for the retrieval function and another blackboard for the document analysis task were both needed, surrounded by different groups of experts to address the tasks required. Each blackboard can be viewed as being made of two parts: the blackboard

proper, and the strategist. As can be seen in Figure 5, the blackboard portion has subject areas for each major class of information, and two priority areas. While experts may be restricted to examining and writing to certain subject areas, all experts can access the priority areas. Thus, questions and answer sets are present in the question/answer area when experts need or can offer information. In addition, pending hypotheses (e.g., that a particular document might be relevant, or that a user has a certain level of knowledge in a topical area) are posted in the pending hypothesis area when the strategist notices a hypothesis (or collection of related hypotheses) in some subject area, with a high confidence value associated. Relating this to the retrieval problem, it should be noted that since high levels of recall are often difficult to achieve, the possibility of having multiple interpretations and hypotheses should allow query splitting and other methods of expanding term sets and queries.

<Have Figure 5 about here.>

The strategist has five components: to manage the aforementioned posting areas, to identify experts and priorities relating to handling a new question, to maintain dependency chains between hypotheses (in case of later retraction), to select experts and priorities that are needed based on the current phase of processing, and to actually "wake" and otherwise control experts.

All user interaction is through the user interface manager. Special commands for analysis or retrieval can be given, and are handled by the command parser. The report expert can cause display or filing of results. Explanations are based on the current user and the blackboard state. Browsing is possible of both the document database and the lexicon. The user model builder updates the user model base as a result of events on the blackboard.

Document analysis begins when a command is received by the analysis blackboard. The document file manger affords access to incoming documents, which in raw form are handled by the text storage manager. Analysis of temporal references and document type are handled by specialists, while the document analyzer performs the general tokenizing, parsing, and knowledge

structure building (that leads to entry in the document knowledge base).

Retrieval is prompted by an explicit (or default, from the user model base) query. User model building, problem state transformation, and building of the problem description all proceed. When some terms are available, the lexicon can be accessed by the two term expanders to obtain other related terms that can be browsed or automatically used to help construct a query. Eventually a p-norm or other query is constructed, a search is made, and a report is made to the user.

The relationship between the strategist components, the experts, the blackboards, the external knowledge bases, and the various resource managers, can all be seen in Figure 6. The almost loop free structure simplifies the task of the strategist by reducing the likelihood of deadlock.

<Have Figure 6 about here.>

The design of CODER calls for specific types of behavior by each expert, and indicates what each part of the blackboard/strategist complex must do. Details of calls and the resulting processing are shown in Figure 7. It should be noted that the posting area manager is the strategists' primary window provided to the outside, and the task dispatcher is the communicator between the strategist and the experts. The question/answer handler and the domain task scheduler together provide the real knowledge-based control of system operation, which is most restrictive if only one processor is involved, and is minimal when many processors are available so that every expert can spot and perform work as soon as possible.

<Have Figure 7 about here.>

Since CODER is a message passing system that can be distributed across machines, the TCP/IP protocol suite is used for actual communications. A client/server model allows the server to queue up pending requests that can each be dealt with. Information flows between sockets, but to the developer all this is transparent, looking like another rule is being invoked. This facility is provided as a result of modifications made to the MuProlog interpreter; the way it is implemented can be seen in Figure 8.

<Have Figure 8 about here.>

4.3 Knowledge Engineering

A second key aspect of the approach taken in CODER is to use knowledge whenever possible. Behind the scenes, a knowledge engineer must develop a type system for each subject domain, so that objects/entities/events and relationships are properly described (by frames and relations, respectively) for matching and reasoning. For example, in the application domain of document retrieval, where citations to various objects are abundant, the frame hierarchy shown in Figure 9 is valuable for classification.

<Have Figure 9 about here.>

It should be noted that the hierarchy shows a kind of (AKO) relationships, so that a Proceedings is AKO Report which is AKO Publication ... What this means is that all slots of a parent frame are inherited by the child type. By using strict typing, the computational complexity of matching is drastically reduced [LEVE 84]. Since these operations may be performed on the large fact stores in the external knowledge bases, efficiency is an important consideration.

For document analysis, it is vital to rapidly determine the type of a given digest message, according to the breakdown in Figure 10. Each type can be identified by applying a set of heuristics (see [FOXE 86d] for an example), and frames for each can be filled in with distinguishing characteristics.

<Have Figure 10 about here.>

In [FRAN 86b] it is shown how knowledge of terms can allow CODER to select documents based on matches of conceptual clusters rather than term matches. The lexicon and domain specific knowledge available are very useful in this regard, as can be seen in the next section.

While the external knowledge bases store the factual information resident in the system, the various experts each have local rule bases that can be used for forward or backward chaining

purposes involved in their particular tasks. Some commonality among experts is possible, especially when similar tasks (e.g., classification) are involved.

5. Implementation

Since development of CODER involves research assistants, students working on M.S. projects, and students completing class projects, it is difficult to precisely characterize the status of implementation. The knowledge administration complex, the blackboard/strategist complex, the communications enhancements to MuProlog, and two versions of the user interface manager are all nearly complete. Initial versions of the document type expert and the user model builder are being further developed. The time reasoning expert and the document analyzer are partially complete. Details on some of the completed components are given below.

5.1 Knowledge Base Development

To make it easy for CODER to be ported to other computers (that support UNIX and TCP/IP), and for simplicity and flexibility, MuProlog and its built-in database package [NAIS 85] was selected for coding rules and storing facts. The database package is described in [RAMA 85] and is an implementation of superimposed coding [SACK 85].

A variety of types of data have been stored in the form of Prolog facts, as can be seen in Table 1. Part A shows the current document collection, which is already about as big as the largest document collection used in earlier studies (i.e., the INSPEC collection). Statistics are based on the SMART form, since searching and data collection routines are in use on that system.

<Have Table 1 about here>

Parts B and C relate to the two parts of the lexicon. The text of the *Handbook of Artificial Intelligence* is available online, and various kinds of information have been extracted from it. First, to help provide a hierarchical organization to the artificial intelligence discipline, the Table of Contents of the 3 volume work is shown in Table 2.

<Have Table 2 about here>

Second, the initial portion of a more detailed topical hierarchy is shown in Table 3. Part A illustrates the hierarchy by indenting, once for section, twice for subsection, three times for subsubsection, ... The numbers shown in Part A actually are subject numbers, which are paired with the relevant (title) word or phrase given in Part B. These numbers are in turn related back to the text of the *HAI*. Thus, a searcher can browse through the subject heading structure of the *HAI*, to become more familiar with its organization of knowledge about artificial intelligence.

<Have Table 3 about here>

Third, the back of the index entries are given in Table 4. There are three relations involved. Some index items refer to a single text line in a particular file, and are handled by the "index_ref" relation. Other index items refer to a range of lines, and are handled by the "index_rng" relation. For ease of use, the "person" relation is shown in part C, and is taken from the other two index relations whenever a person name was present.

<Have Table 4 about here>

Finally, Table 5 lists the initial portion of the very long file of italicized words/phrases, along with the related text pointer. A user can then go back and forth between a phrase and the section it appears in, and on up the hierarchy; alternatively the user can begin with a section and find phrases that are important. All in all having the machine readable form of the *HAI* can allow users to browse in the text and terminology of the artificial intelligence discipline.

<Have Table 5 about here>

In addition to the HAI knowledge and text base, the CODER lexicon includes a set of 21 relations extracted [WOHL 86] from the *Collins Dictionary of the English Language (CDEL)* [HANK79]. This very large (over 80,000 headwords) dictionary (see statistics in part C of Table 1) provides a wealth of information to help with term expansion and natural language processing. Initial efforts have been made to supplement the CDEL portion of the lexicon with information from three other machine readable dictionaries: [COWI 75], [COWI 83], and [HORN74]. Additional information can be obtained from these dictionaries through more detailed analysis. A scheme for this and a more thorough description of the current lexicon is given in [FOXE 86e].

5.2 P-norm search expert

The p-norm query notation, which extends Boolean expressions to allow relative weights to be attached to terms and clauses, and which allows "p-values" on the AND and OR operators to indicate the strictness of interpretation of the operation, was first discussed in [SALT 83]. While other schemes for "soft Boolean evaluation" have been proposed [PAIC 84], none has been shown to perform as effectively as the p-norm method [FOXE 86a]. P-norm query processing has been incorporated in both the SMART and SIRE systems [FOXE 87b].

Because of its expressive power, the p-norm query form has been adopted in CODER as one of the canonical query forms. As can be seen in Figure 11, a p-norm search expert has been developed that supports calls through the blackboard to attend to the pnorm_query area. The result of normal processing is to generate hypotheses indicating that documents which best satisfy the query expression may be relevant to the query.

<Have Figure 11 about here>

5.3 Time Reasoning

In order to support time references appearing in connection with a problem description, it is necessary to be able to parse queries and documents, and to carry out temporal reasoning as required. Since a separate time reasoning project (by J. Roach et al.) is already underway at Virginia Tech, and may be adaptable for our purposes, our focus has been on identification, parsing, and representation of time indicative expressions. The easiest to work with are the entries in digest and message headers.

In the "Date:" field of electronic mail messages, a variety of notations for date and time can be found. Figure 12 shows forms for dates and times that our parser can process, and some samples of such fields found in AIList messages. As a result of using the date/time parser, each digest and each message can be related to a (small) absolute time interval attached to a time line.

<Have Figure 12 about here>

Many different words and phrases, found in the body of documents and in queries, indicate that a time interval is being described [BENN 75],[FUNK 53]. Based on the *CDEL* lexicon, in part A of Figure 13 is shown a listing of some common time words/phrases, categorized by part of speech. The duration of an interval is important for temporal reasoning, and is conveyed by the class of word shown in part B of Figure 13. Further, a past/present/future aspect classification is given in part C.

<Have Figure 13 about here>

Given that a time word has been identified, it is common to find it as an element of a prepositional phrase (PP). The first part of Figure 14 gives the syntax and semantics of such PPs, indicating how a frame can be constructed to record some of the key aspects of the phrase's meaning. Part B goes on to outline the temporal reasoning requirements and plan.

<Have Figure 14 about here>

5.4 User interaction and information gathering

As mentioned earlier in connection with the blackboard, the work of Belkin, Brooks, and Daniels, adapted to our particular environment and collection, has informed our approach to user interaction. There are a number of phases or types of interaction between the user and system, which are listed in part A of Figure 15. The various system experts (shown in part in Figure 5) are actually in charge of these activities — e.g., the explanation expert handles tutorials, help and other forms of explanation.

<Have Figure 15 about here>

In the current implementation, background information as listed in part B of Figure 15 is gathered. Some initial work on the user model builder has taken place, and more is scheduled through mid 1987. At present, all data collected is logged. Problem state and description indicators are also requested, as shown in parts C and D of Figure 15, and will later be handled by the appropriate builder experts (shown at the top of Figure 5).

Finally, to gauge the user's feeling toward the system and its operation, evaluation questions are asked that relate to the various factors listed in part E of Figure 15. With this feedback, the system can be tuned as a whole and to individual users' needs, and should hopefully be shown to more effectively aid end user searching than would conventional approaches.

6. Conclusions

In light of new developments in computer and communications technology, there is a growing need for end users to search for bibliographic references, document passages, or other items with a textual component. In the case of composite documents, where structure and type interact, and where particular components or passages are to be retrieved, it is necessary to carry out a more complete analysis of input text and to use additional information besides counts of word (stem) matches.

The CODER system has been under development since 1985 to serve as a test bed for the application of artificial intelligence methods to information retrieval problems. Though organized in a flexible fashion to handle a variety of retrieval tasks, its initial testing will be with messages in an archive of AIList Digest issues. It is hoped that CODER will perform well in this difficult domain, demonstrating the feasibility of analyzing and search for passages in a composite document collection, and of applying artificial intelligence techniques.

While it may be true that certain types of knowledge based processing do not contribute to retrieval performance [SALT 86], there is evidence that the approach taken in the CODER project will be of use. The blackboard based construction allows modular development, and enforces clean separation between fact bases, resource managers, blackboard communication, strategist control, and expert processing. An expert for p-norm searching functions well in conjunction with the MuProlog database package, and some work has begun on time expression parsing and temporal reasoning. Data is being gathered that will aid with user model building. The emphasis on using knowledge has led to a large lexicon constructed from machine readable texts; the dictionary portion should be of general use and the domain specific portion could easily be replaced by similar information taken from reference books in another domain. It is expected that by the middle of 1987 a fairly complete prototype will demonstrate the utility of the CODER design.

Acknowledgements

Robert K. France completed his M.S. thesis on the design of the CODER system, and Robert C. Wohlwend completed his M.S. project working on building the *CDEL* Prolog fact base. Qi Fan Chen and Marybeth T. Weaver have worked on numerous parts of CODER in connection with class projects and as graduate research assistants. Joy Weiss has provided secretarial assistance. Numerous other students have helped develop other parts of the system in connection with course and M.S. project efforts.

Bibliography

[ALLE 83] Allen, James F. Maintaining Knowledge about Temporal Intervals. *Commun. ACM*, 26(11):832-843, Nov. 1983.

[AMSL 84] Amsler, R.A. Machine-Readable Dictionaries. *ARIST*, 19:161-209, 1984.

[BABA 85] Babatz, R. and M. Bogen. Semantic Relations in Message Handling Systems: Referable Documents. In *Proc. IFIP WG 6.5 Symposium*, Sept. 1985.

[BATE 86] Bates, Marcia J. Subject Access in Online Catalogs: A Design Model. *J. Am. Soc. Inf. Sci.*, 37(6):357-376, Nov. 1986.

[BELK 84] Belkin, N.J., Hennings, R.D., and T. Seeger. Simulation of a Distributed Expert-Based Information Provision Mechanism. *Inf. Tech.: Res. Dev. Applications*. 3(3): 122-141, 1984.

[BENN 75] Bennett, David C. *Spatial and Temporal Uses of English Prepositions: As Essay in Stratificational Semantics*. Longman: London, 1975.

[BICH 80] Bichteler, J. and Eaton III, E.A. The Combined Use of Bibliographic Coupling and Cocitation for Document Retrieval. *J. Am. Soc. Inf. Sci.*, 31(4):278-282, July 1980.

[BISW 85] Biswas, Gautam, Viswanath Subramanian and James C. Bezdek. A Knowledge Based System Approach to Document Retrieval. In *Proc. CAIA-85*, 455-460.

[BISW 86a] Biswas, Gautam, James C. Bezdek, Marisol Marques, and Viswanath Subramanian. Knowledge- Assisted Document Retrieval: I. The Natural Language Interface. *J. Am. Soc. Inf. Sci.*, (to appear).

[BISW 86b] Biswas, Gautam, James C. Bezdek, Viswanath Subramanian, and Marisol Marques. Knowledge- Assisted Document Retrieval: II. The Retrieval Process. *J. Am. Soc. Inf. Sci.*, (to appear).

[BOOK 83] Bookstein, A. Information Retrieval: A Sequential Learning Process. *J. Am. Soc. Inf. Sci.*, 34(5):331-342, Sept. 1983.

[BORG 84] Borgman, Christine L. Psychological Research in Human-Computer Interaction. *ARIST*, 19:33-64, 1984.

[BORG 85] Borgman, Christine L. Designing an Information Retrieval Interface Based on User Characteristics. In *Res. & Dev. in Inf. Ret., Eighth Annual Int. ACM SIGIR Conf.*, Montreal, 139-146, June 1985.

[BROO 85] Brooks, H.M., P.J. Daniels, and N.J. Belkin. Problem Descriptions and User Models: Developing an Intelligent Interface for Document Retrieval Systems. *Advances in Intelligent Retrieval. Proc. of Informatics 8*, London, ASLIB, 191-214, 1985.

[BUCK 85] Buckley, Chris and Alan F. Lewit. Optimization of Inverted Vector Searches. *Res. & Dev. in Inf. Ret., Eighth Annual Int. ACM SIGIR Conf.*, Montreal, 97-110, June 1985.

[CHAN 86] Chandrasekaran, B. Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design. *IEEE Expert* , 1(3):23-30, Fall 1986.

[CHEE 85] Cheeseman, Peter. In Defense of Probability. *Proc. AAAI-85*, 1002-1009, 1985.

[CLOC 84] Clocksin, W.F. and C.S. Mellish. *Programming in Prolog*. 2nd ed. Springer-Verlag, New York, 1984.

[COWI 75] Cowie, A.P. and R. Mackin. *Oxford Dictionary of Current Idiomatic English. Volume 1: Verbs with Prepositions & Particles*. Oxford Univ. Press, Oxford, 1975.

[COWI 83] Cowie, A.P., R. Mackin, and I.R. McCaig. *Oxford Dictionary of Current Idiomatic English. Volume 2: Phrase, Clause & Sentence Idioms*. Oxford Univ. Press, Oxford, 1983.

[CROF 82] Croft, W.B. and Pezarro, M.T. Text Retrieval Techniques for the Automated Office. In *Office Information Systems*, ed. by N. Naffah, North- Holland, Amsterdam, 565-576, 1982.

[DANI 85] Daniels, P.J., H.M.Brooks, and N.J. Belkin. Using Problem Structures for Driving Human-Computer Dialogues, In *RIAO '85*, IMAG, Grenoble, 1985, 645-660.

[DANI 86a] Daniels, Penny J. The User Modelling Function of an Intelligent Interface for Document Retrieval Systems, In Proc. of *IRFIS 6. Intelligent information systems for the information society*, Frascati, Sept. 1985, Amsterdam, North-Holland, 1986.

[DANI 86b] Daniels, P.J. Cognitive Models in Information Retrieval — an Evaluative Report. Final Report to the British Library Research and Development Department on Project Number SI/G/753, May 1986.

[DEFU 85] Defude, B. Different Levels of Expertise for an Expert System in Information Retrieval. In *Res. & Dev. in Inf. Ret., Eighth Annual Int. ACM SIGIR Conf.*, Montreal, 147-153, June 1985.

[DEJO 82] DeJong, G. An Overview of the FRUMP System. In *Strategies for Natural Language Processing*, ed. by Wendy G. Lehnert and Martin H. Ringle, Lawrence Erlbaum Assoc., Hillsdale NJ, 149-176, 1982.

[ERMA 80] Erman, L.D., Hayes-Roth, F., Lesser, V.R., and D.R. Reddy. The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty. *ACM Comp. Surveys*, 12:213-253, 1980.

[EVEN 79] Evens, M.W. and R.N. Smith. A Lexicon for a Computer Question-Answering System. *Am. J. Comp. Ling.*, Microfiche 83: 1-93, 1979.

[EVEN 85] Evens, Martha., J. Vandendorpe, and Yih-Chen Wang, Lexical Semantic Relations in Information Retrieval. In *Humans and Machines: The Interface Through Language*, Ablex, ed. S. Williams, 73-100, 1985.

[FIKE 85] Fikes, Richard and Tom Kehler. The Role of Frame-Based Representation in Reasoning. *Commun. ACM*, 28(9):904-920, Sept. 1985.

[FOXE 80] Fox, E.A. Lexical Relations: Enhancing Effectiveness of Information Retrieval Systems. *ACM SIGIR Forum*, 15(3):5-36, Winter 1980.

[FOXE 83a] Fox, E.A. Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types. Dissertation, Cornell University, University Microfilms Int., Ann Arbor MI, Aug. 1983.

[FOXE 83b] Fox, E.A. Some Considerations for Implementing the SMART Information Retrieval System under UNIX. TR 83-560, Cornell Univ., Dept. of Comp. Sci., Sept. 1983.

[FOXE 84a] Fox, E.A. Combining Information in an Extended Automatic Information Retrieval System for Agriculture. In *The Infrastructure of an Information Society* , ed. B. El-Hadidy and E.E. Horne, North-Holland, Amsterdam, 449-466, 1984.

[FOXE 84b] Fox, E.A. Improved Retrieval Using a Relational Thesaurus Expansion of Boolean Logic Queries. In Proc. *Workshop Relational Models of the Lexicon*, Martha W. Evens (ed.), Stanford, CA, July 1984 (to appear).

[FOXE 85] Fox, E.A. Composite Document Extended Retrieval: An Overview. In *Res. & Dev. in Inf. Ret., Eighth Annual Int. ACM SIGIR Conf.*, Montreal, 42-53, June 1985.

[FOXE 86a] Fox, E.A. and S. Sharat. A Comparison of Two Methods for Soft Boolean Operator Interpretation in Information Retrieval, TR-86-1, Virginia Tech Dept. of Comp. Sci., Jan. 1986.

[FOXE 86b] Fox, E.A. Information Retrieval: Research into New Capabilities. In *CD-ROM: The New Papyrus*, Steve Lambert and Suzanne Ropiequet (eds.), Microsoft Press, 1986, 143-174.

[FOXE 86c] Fox, E. A. A Design for Intelligent Retrieval: The CODER System. *The Second Conference on Computer Interfaces and Intermediaries for Information Retrieval,* 28-31 May 1986, Boston MA.

[FOXE 86d] Fox, E. A. Expert Retrieval for Users of Computer Based Message Systems. *Proceedings 49th Annual Meeting Amer. Soc. for Inf. Sci.,* Sept.28-Oct.2, 1986, Chicago, IL, 23:88-95.

[FOXE 86e] Fox, E.A., Robert C. Wohlwend, Phyllis R. Sheldon, Qi Fan Chen, and Robert K. France. Building the CODER Lexicon, Phase 1: The Collins English Dictionary and Its Adverb Definitions, TR-86-23, VPI&SU Computer Science Dept., Blacksburg, VA, October 1986.

[FOXE 87a] Fox, E.A. and Robert K. France. Architecture of an Expert System for Composite Document Analysis, Representation and Retrieval. *Int. J. of Approximate Reasoning*, 1(2), April 1987, to appear.

[FOXE 87b] Fox, E.A. and Matthew B. Koll. Practical Enhanced Boolean Retrieval: Experiences with the SMART and SIRE Systems. *Information Processing and Management,* to appear.

[FOXM 86] Fox, Mark S., J. Mark Wright and David Adam. Experiences with SRL: An Analysis of a Frame-based Knowledge Representation. In *Expert Database Systems: Proceeding from the First International Workshop,* Larry Kerschberg (ed.), Menlo Park CA, Benjamin/Cummings Pub. Co., Inc., 161-172, 1986.

[FRAN 86a] France, Robert K. An Artificial Intelligence Environment for Information Retrieval Research. MS Thesis, VPI&SU Dept. of Comp. Sci., Blacksburg VA, July 1986.

[FRAN 86b] France, R.K. and E.A. Fox. Knowledge Structures for Information Retrieval: Representation in the CODER Project. *Proceedings IEEE Expert Systems in Government Conference*, October 20-24, 1986, McLean VA, 135-141.

[FREI 84] Frei, H.P. and Jauslin, J.F. Two-Dimensional Representation of Information Retrieval Services. In *Representation and Exchange of Knowledge as a Basis of Information Processes*, ed. by Hans J. Dietschmann, North-Holland, New York, 383-396, 1984.

[FUNK 53] Funk & Wagnalls Editorial Staff. *Standard Handbook of Prepositions, Conjunctions, Relative Pronouns and Adverbs*. Funk&Wagnalls: New York, 1953.

[HAHN 84] Hahn, U. and Reimer, U. Heuristic Text Parsing in 'Topic': Methodological Issues in a Knowledge-based Text Condensation System. In *Representation and Exchange of Knowledge as a Basis of Information Processes*, ed. by Hans J. Dietschmann, North-Holland, New York, 143-163, 1984.

[HANK 79] Hanks, P. ed. *Collins Dictionary of the English Language,* William Collins Sons & Co., London, 1979.

[HARP 86] Harper, D.J., J. Dunnion, M. Sherwood-Smith, and C.J. van Rijsbergen. Minstrel-ODM: A Basic Office Data Model. *Inf. Proc. & Mgmt.*, 22(2):83-108, 1986.

[HAYE 85] Hayes-Roth, F. Rule-Based Systems. *Commun. ACM,* 28(9):921-932, Sept. 1985.

[HELM 85] Helm, A.R., Marriott, Kimbal, and Catherine Lassez. Prolog for Expert Systems: An Evaluation. *Proc. of Expert Systems in Government Symp.*, 284-293, October 1985.

[HORN 74] Hornby, A.S. ed. *Oxford Advanced Dictionary of Current English,* Oxford University Press, Oxford, 1974.

[KATZ 82] Katzer, J., et. al. A Study of the Overlap Among Document Representations."*Inf. Tech.: Res. & Dev.*, 1(4): 261-274, Oct. 1982.

[KATZ 86] Katzer, Jeffrey, Susan Bonzi and Elizabeth Liddy. The Effects of Anaphoric Resolution on Retrieval Performance: Preliminary Findings. *Proc. 49th Ann. Mtg. Amer. Soc. Inf. Sci.*, Sept. 28 - Oct. 2, 1986, Chicago, IL, 118-122.

[KIMU 84] Kimura, G.D. A Structure Editor and Model for Abstract Document Objects. Dissertation. Tech. Report No. 84-07-04, Dept. of Comp. Sci., Univ. Washington, July 1984.

[KORF 84] Korfhage, Robert R. and Charles Hemphill. Retrieval Linguistics. Tech. Report No. 84-CSE-12, Southern Methodist University Dept. of Comp. Science, Dallas Texas, 1984.

[KORF 86] Korfhage, Robert R. Browser: A Concept for Visual Navigation of a Database. Tech. Report No. 86-CSE-4, Southern Methodist University Dept. of Comp. Science, Dallas Texas, Feb. 1986.

[KOWA 79] Kowalski, R.A. *Logic for Problem Solving*. Elsevier North-Holland, New York, 1979.

[KUCE 85] Kucera, Henry. Uses of On-Line Lexicons. Proc. *First Conference of the UW Centre for the New Oxford English Dictionary: Information in Data*. Nov. 6-7, 1985, Waterloo, Canada, 7-10.

[KWOK 86] Kwok, K.L. The Concept of Document Components for Probabilistic Indexing. *Proc. 49th Ann. Mtg. Amer. Soc. Inf. Sci.*, Sept. 28 - Oct. 2, 1986, Chicago, IL, 158-162.

[LECO 86] Lecot, Koenraad. Logic Programs with Uncertainties: Dealing with Multiple Evidence. *Proceedings IEEE Expert Systems in Government Conference*, October 20-24, 1986, McLean VA, 234-242.

[LEEN 86] Lee, Newton S. Fuzzy Inference Engines in Prolog/P-Shell. *Proceedings IEEE Expert Systems in Government Conference*, October 20-24, 1986, McLean VA, 243-247.

[LEVE 84] Levesque, Hector J. A Fundamental Tradeoff in Knowledge Representation and Reasoning. *Proceedings of the Fifth CSCSI National Conference (London, ON, May 1984):* pp. 141-152.

[MAUL 86] Mauldin, Michael L. Thesis Proposal: Information Retrieval by Text Skimming, unpublished manuscript, Dept. of Computer Science, Carnegie-Mellon University, Pittsburgh PA, May 22, 1986.

[MAYS 86] Mays, Eric. A Temporal Logic for Reasoning About Changing Data Bases in the Context of Natural Language Question-Answering. In *Expert Database Systems: Proceeding from the First International Workshop,* Larry Kerschberg (ed.), Menlo Park CA, Benjamin/Cummings Pub. Co., Inc., 559-578, 1986.

[MILL 85] Miller, George A. Wordnet: A Dictionary Browser. Proc. *First Conference of the UW Centre for the New Oxford English Dictionary: Information in Data*. Nov. 6-7, 1985, Waterloo, Canada, 25-28.

[MYAE 86] Myaeng, Sung and Robert R. Korfhage. Towards an Intelligent and Personalized Information Retrieval System. Tech. Report No. 86-CSE-10, Southern Methodist University Dept. of Comp. Science, Dallas Texas, March. 1986.

[MYER 83] Myer, T.A. Standards for Global Messaging: A Progress Report. *J. Telecommunication Networks,* 2(4) (Winter 1983).

[NAIS 85] Naish, Lee. *MU-Prolog 3.2db Reference Manual*. Melbourne Univ., July 1985.

[NIIH 86a] Nii, H. Penny. Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures. *The AI Magazine*, 7(1):38-53, Summer 1986.

[NIIH 86b] Nii, H. Penny. Blackboard Systems: Blackboard Application Systems, Blackboard Systems from a Knowledge Engineering Perspective. *The AI Magazine*, 7(3):82-106, August 1986.

[OCON 80] O'Connor, J. Answer-Passage Retrieval by Text Searching. *J. Am. Soc. Inf. Sci.*, 31(4):227-239, 1980.

[ODDY 77] Oddy, R.N. Information Retrieval Through Man-Machine Dialogue. *J. Doc.*, 33(1):1-14, March 1977.

[OJAL 86] Ojala, Marydee. Views on End-User Searching. *J. Am. Soc. Inf. Sci.*, 37(4), 197-203, 1986.

[PAIC 84] Paice, C.D. Soft Evaluation of Boolean Search Queries in Information Retrieval. *Inf. Tech.: Res. Dev. Applications*, 3(1):33-42, 1984.

[PATE 84] Patel-Schneider, P.F., R.J. Brachman, and H.J. Levesque. ARGON: Knowledge Representation meets Information Retrieval. Fairchild Technical Report No. 654, FLAIR Technical Report No. 29, Sept. 1984.

[PERE 83] Pereira, F. Logic for Natural Language Analysis. Tech. Note 275, SRI Int., Jan. 1983.

[QUAR 86] Quarterman, John S. and Josiah C. Hoskins. Notable Computer Networks. *Commun. ACM*, 29(10):932-971, Oct. 1986.

[RAMA 85] Ramamohanaro, Kotagiri and John Shepherd. A Superimposed Codeword Indexing Scheme for Very Large Prolog Databases. Tech. Report 85/17, Dept. of Comp. Sci., Univ. of Melbourne, 1985.

[SACK 85] Sacks-Davis, Ron. Performance of a multi-key access method based on descriptors and superimposed coding techniques. *Inform. Systems,* 10(4), 391-403, 1985.

[SALT 80] Salton, G. The SMART System 1961-1976: Experiments in Dynamic Document Processing. In *Encyclopedia of Library and Information Science*, 1-36, 1980.

[SALT 83] Salton, G., Fox, E.A., and Wu. H. Extended Boolean Information Retrieval. *Commun. ACM*, 26(11):1022-1036, Nov. 1983.

[SALT 86] Salton, G. On the Use of Knowledge-Based Processing in Automatic Text Retrieval. *Proceedings 49th Annual Meeting Amer. Soc. for Inf. Sci.,* Sept.28-Oct.2, 1986, Chicago, IL, 23:277-287.

[SCIO 86] Sciore, Edward and David Scott Warren. Towards An Integrated Database-Prolog System. In *Expert Database Systems: Proceeding from the First International Workshop,* Larry Kerschberg (ed.), Menlo Park CA, Benjamin/Cummings Pub. Co., Inc., 293-305, 1986.

[SIMM 83] Simmons, Robert F. A Text Knowledge Base for the AI Handbook. Univ. of Texas at Austin Dept. of Comp. Sci., Technical Report TR-83-24, Dec. 1983.

[SOMM 85] DSA — A Tool for Descriptive Text Analysis. Univ. of Strathclyde Software Technology Research Group, Research Report CS/ST/2/85, 1985.

[SPAR 73] Sparck Jones, K. and Martin Kay. *Linguistics and Information Science*, Academic Press, New York, 1973.

[SPAR 84] Sparck Jones, K. and J.I. Tait. Automatic Search Term Variant Generation. *J. Doc.*, 40(1):50-66, March 1984.

[STER 86] Sterling, Leon and Ehud Shapiro. *The Art of Prolog*. MIT Press: Cambridge MA, 1986.

[SVEN 86] Svenonius, Elaine. Unanswered Questions in the Design of Controlled Vocabularies, *J. Am. Soc. Inf. Sci.*, 37(5):331-340, Sept. 1986.

[TENO 84] Tenopir, Carol. Full-Text Databases.*ARIST*, 19:215-246, 1984.

[THOM 85] Thompson, R.H. and W.B. Croft. An Expert System for Document Retrieval. *Proc. Expert Systems in Gov. Symp.*, IEEE, 448-456, Oct. 1985.

[TONG 86] Tong, Richard M et al. RUBRIC III: An Object-Oriented Expert System for Information Retrieval. *Proceedings IEEE Expert Systems in Government Conference*, October 20-24, 1986, McLean VA, 106-115.

[TRIV 86] Trivison, Donna, Alice Y. Chamis, Tefko Saracevic, and Paul Kantor. Effectiveness and Efficiency of Searchers in Online Searching: Preliminary Results from a Study of Information Seeking and Retrieving. In *ASIS '86, Proc. 49th ASIS Ann.Mtg.*, Chicago, IL, 341-349, Sept. 28 - Oct. 2, 1986.

[VANR 86] Van Rijsbergen, C.J. A New Theoretical Framework for Information Retrieval. *Proc. 9th Annual Int'l SIGIR Conf. on Research & Devel. in Inf. Ret.*, Pisa, Italy, 194-200, Sept. 1986.

[WALK 85] Walker, Donald E. Knowledge Resource Tools for Accessing Large Text Files. Proc. *First Conference of the UW Centre for the New Oxford English Dictionary: Information in Data.* Nov. 6-7, 1985, Waterloo, Canada, 11-24.

[WANG 85] Wang, Y.-C., J. Vandendorpe, and M. Evens. Relational Thesauri in Information Retrieval. *J. Am. Soc. Inf. Sci.*, 36(1): 15-27, Jan. 1985.

[WILL 85] Williams, Phil W. How Do We Help the End User? Proc. *6th National Online Meeting*, April 30-May 2, 1985, 495-505.

[WILL 86] Williams, Martha E. et al. Comparative Analysis of Online Retrieval Interfaces. In *ASIS '86, Proc. 49th ASIS Ann.Mtg.*, Chicago, IL, 365-370, Sept. 28 - Oct. 2, 1986.

[WOHL 86] Wohlwend, Robert C. Creation of a Prolog Fact Base from the Collins English Dictionary. MS Report, VPI&SU Computer Science Dept., Blacksburg, VA, March 1986.

[ZARR 83] Zarri, G.P. An Outline of the Representation and Use of Temporal Data in the RESEDA System. *Inf. Tech.: Res. Dev. Applications*, 2(2/3):89-108, July 1983.

Figure 1. Functional Perspective of the X.400 Model

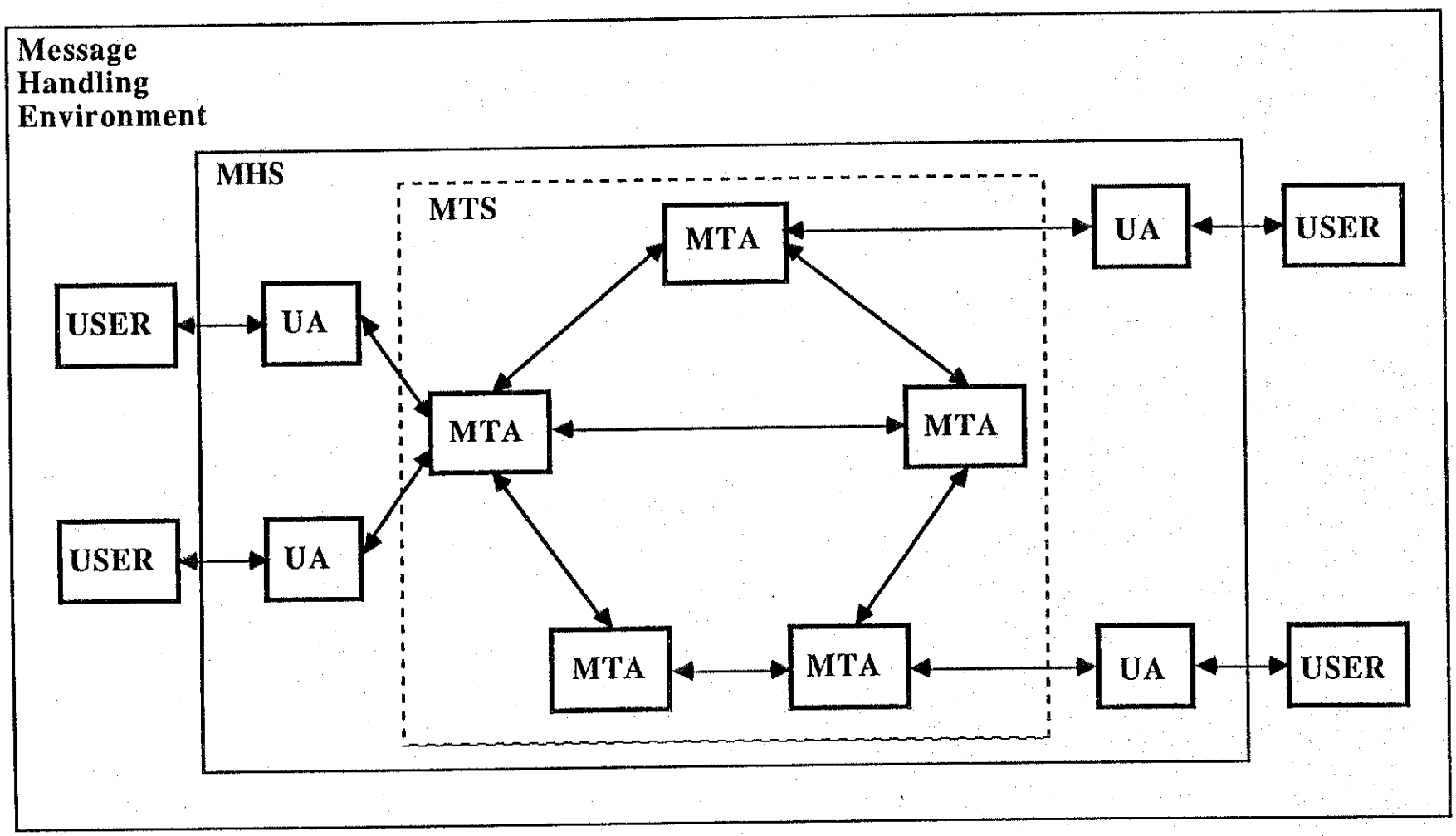

Figure 2. Examples of AIList message contents

A) Citation, abstract, question about the article.

RIT Researchers Find Way to Reduce Transmission Errors,
Communications of the ACM, Vol. 29, No. 7, July 1986, p. 702:

Donald Kreher and Stanislaw Radziszowski at Rochester Institute of Technology
...
Integer (Diophantine) equations are notoriously difficult to solve. Is this ...

B) Reference to earlier msg, using author name

Grethe Tangen asked about using mathematical models of gas turbines ...

C) Reference to earlier msg, using "Re" and ">>>" indented extract

>>> instead of a computer trying to fool you in ASCII,
...

The purpose of scientific inquiry is not just to better the human
condition. It is also to understand nature, including human nature. ...

D) Embedded prior msg, followed by commentary

Date: Mon 29 Sep 86 09:55:11-PDT
From: Pat Hayes <PHayes@SRI-KL.ARPA>
Subject: Searle's logic

Look, I also don't think there's any real difference between a human's
...
At one end of the human knowledge spectrum we have that knowledge of a ...

E) Summary of a report, divided into sections, with lists

Summary of Volume 2 No 10

Discussion of S.1, ART, KEE discussing user interface, performance, features.
A common problem was done in all three applications. The person who did
...

Japan Watch

MITI has budgeted the following for AI type products

$400,000 diagnosis support systems
2.4 million robotics
1.1 million language translation systems
$234,000 factory automation R&D

F) Seminar, with memo style header, abstract

Title: Learning Apprentice Systems
Speaker: Prof. Tom Mitchell, Carnegie-Mellon University
Location: Rm. 2324 Dept of CS, U of MD, College Park
Time: 4:00pm
We consider a class of knowledge-based systems called Learning
Apprentices: systems that provide interactive aid in solving some problem, ...

Figure 2. Examples of AIList message contents -- cont'd

G) Forwarded survey, with address first, indented memo style

[Forwarded from the AI-Ed digest by Laws@SRI-STRIPE.]

Here I present a survey of Intelligent Tutoring systems which,
...
You can't use REPLY to get to me so you need to SEND me Email to
YAZDANI%UK.AC.EXETER.PC@UCL-CS.arpa
or post to
...
ACE
Subject: Nuclear Magnetic Spectroscopy
Aim: Monitor Deductive Reasonsing
Features: Problem solving monitor, accepts natural language input
System: MODULAR ONE
Reference:
Sleeman, D.H., and Hendley, R. J. (1982)
ACE: a system which analyses complex explanations
in Sleeman and Brown (eds.) ...

H) Call for Papers, with centering, lists, addresses

FINAL CALL FOR PAPERS:
Optical Society Topical Meeting on
MACHINE VISION
March 18-20, 1987
Hyatt Lake Tahoe, Incline Village, Nevada

Topics will include: 3-D vision algorithms, image understanding,
...
Invited speakers include: Bob Bolles (SRI), Peter Burt (RCA),
...
Program committe: Alex Pentland, Glenn Sincerbox (co-chairs),
...
WHAT TO SUBMIT: 25 WORD abstract and separate 4 PAGE camera-ready
...
Optical Society of America
Machine Vision
1816 Jefferson Place, N.W.
Washington, D.C. 20036
DEADLINE: Nov. 3, 1986

I) Bibliography in UNIX roffbib form

...
%A Richard Forsyth
%A Roy Rada
%T Machine Learning Applications in Expert Systems and Information Retrieval
%I John Wiley and Sons
%C New York
%D 1986
%K AT15 AA15 AI01 AI04
%X ISBN 0-20309-9 Cloth $49.95 , ISBN 0-20318-18 $24.95 paper 277 pages ...

Figure 2. Examples of AIList message contents -- cont'd

J) Technical report explanation, centered title&address, list

[Forwarded from the UTexas-20 bboard by Laws@SRI-STRIPE.]
Following is a listing of the reports available from the AI Lab.

...

TECHNICAL REPORT LISTING
Artificial Intelligence Laboratory
University of Texas at Austin
Taylor Hall 2.124
Austin, Texas 78712
(512) 471-9562
September 1986
All reports furnished free of charge

...

AI84-05 A Text Knowledge Base for the AI Handbook, Robert F. Simmons, December 1983.

K) Grant awards, with title, explanation, address, list

Fiscal Year 1986 Research Projects
Funded by the Information Science Program
(now Knowledge and Database Systems Program)

A complete listing of these awards, including short descriptive abstracts of the research is available by writing to:

Joseph Deken, Director
Knowledge and Database Systems Program
National Science Foundation
1800 G Street NW
Washington, DC 20550

...

IST-8518307
$15,750 - 12 mos.
Donald H. Kraft
Louisiana State University

Travel to the ACM Conference on Research and Development in Information Retrieval: Pisa, Italy; September 8-10, 1986 ...

L) Header, explanation, address, dissertation title and abstract

Date: Thu, 9 Oct 86 10:21:18 EDT
From: "Charles W. Anderson" <cwa0%gte-labs.csnet@CSNET-RELAY.ARPA>
Subject: Dissertation - Multilayer Connectionist Learning

The following is the abstract from my Ph.D. dissertation completed in August, 1986, at the University of Massachusetts, Amherst. Members of my committee are Andrew Barto, Michael Arbib, Paul Utgoff, and William Kilmer. I welcome all comments and questions.

Chuck Anderson
GTE Laboratories Inc.
40 Sylvan Road
Waltham, MA 02254
617-466-4157
cwa0@gte-labs

Learning and Problem Solving
with Multilayer Connectionist Systems

The difficulties of learning in multilayered networks of computational units has limited the use of connectionist

Figure 3. Query input form, annotated with indexing instructions

FIELD LETTER (& explanation)	EXAMPLES / Indexing Instructions
.WHEN DIGEST ISSUED	(any format - Jan. 1, 1986, or 1/01/86, etc) Concept type: 0 Parsing: full Token-type: date
.VOLUME NUMBER	(a known volume number- 1, 2, 3, 4,) Concept type: 1 Parsing: full Token-type: number
.U (ISSUE NUMBER)	(a known issue number- 34, 87, etc) Concept type: 2 Parsing: full Token-type: number
.DATE MESSAGE SENT	(any format - Jan. 1, 1986, or 1/01/86, etc) Concept type: 3 Parsing: full Token-type: date
.NAME OF SENDER	(first and/or last name(s)- Roger Schank) Concept type: 4 Parsing: full Token-type: name
.A (network ADDRESS of sender)	(e-mail address- benda@usc-isi.arpa) Concept type: 5 Parsing: token Token-type: token
.SUBJECT	(term(s) in message heading- case grammar) Concept type: 6 Parsing: full Token-type: word,p-nouns
.BODY	(term(s) in message body- computational linguistics) Concept type: 6 Parsing: full Token-type: word,p-nouns

Figure 4. Output of SMART pre-parser, ready to be indexed

.I 5495
.W **<When was digest sent?>**
Friday, 10 Oct 1986
.V **<Volume of AIList>**
Volume 4
.U **<issUe of AIList in current volume>**
Issue 211
.D **<Date author sent in message to digest editor>**
Thu, 9 Oct 86 10:21:18 EDT
.N **<Name of message author>**
Charles W. Anderson
.A **<Address of message author>**
cwa0%gte-labs.csnet@CSNET-RELAY.ARPA
.S **<Subject field of message>**
Dissertation - Multilayer Connectionist Learning
.B **<Body of message>**
The following is the abstract from my Ph.D. dissertation completed in August, 1986, at the University of Massachusetts, Amherst. Members of my committee are Andrew Barto, Michael Arbib, Paul Utgoff, and William Kilmer. I welcome all comments and questions.

Chuck Anderson
GTE Laboratories Inc.
40 Sylvan Road
Waltham, MA 02254
617-466-4157
cwa0@gte-labs

Learning and Problem Solving
with Multilayer Connectionist Systems

The difficulties of learning in multilayered networks of computational units has limited the use of connectionist systems in
...

Figure 5. Overview of CODER System

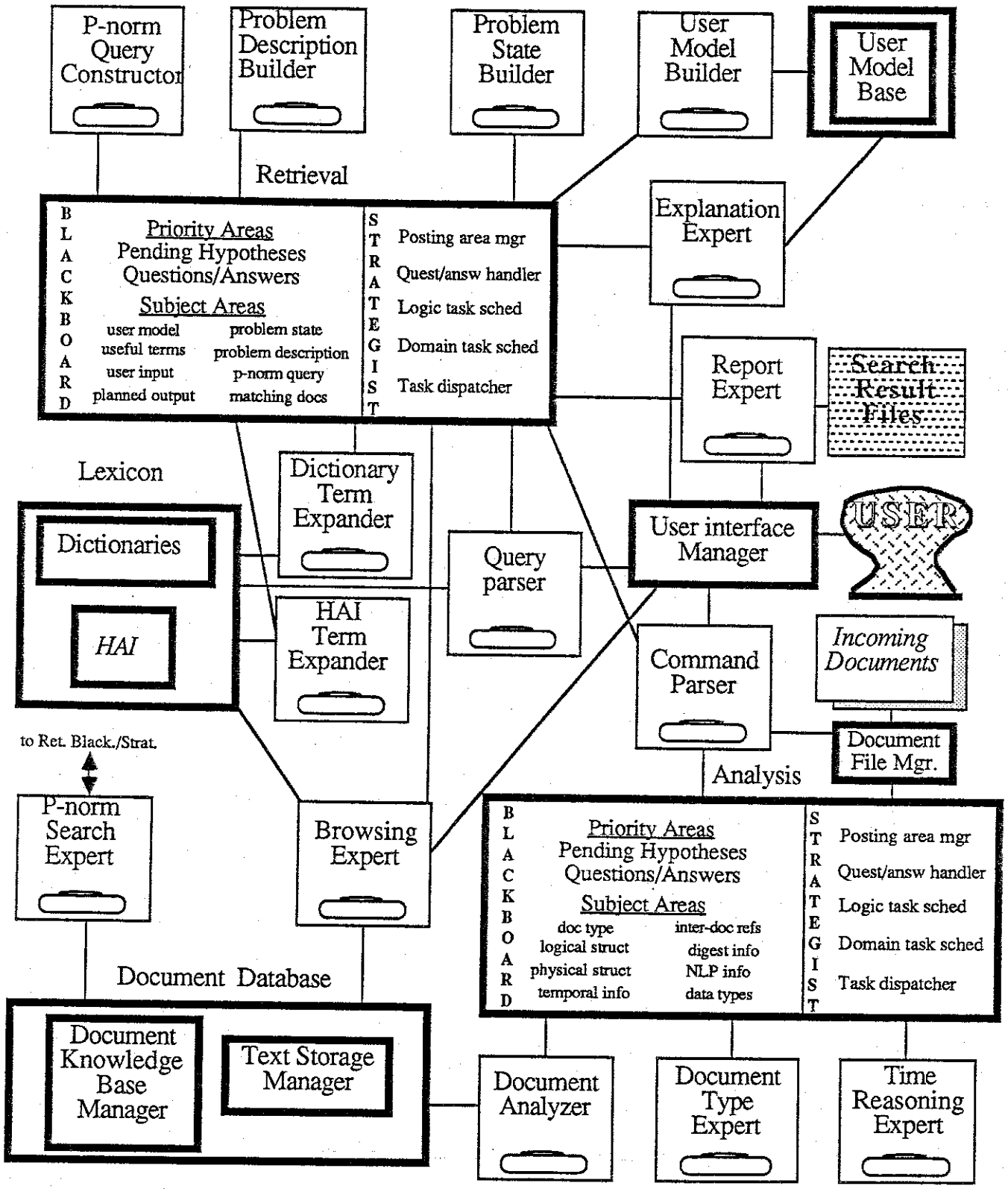

Figure 6. Detailed calling hierarchy for a CODER Community of experts.

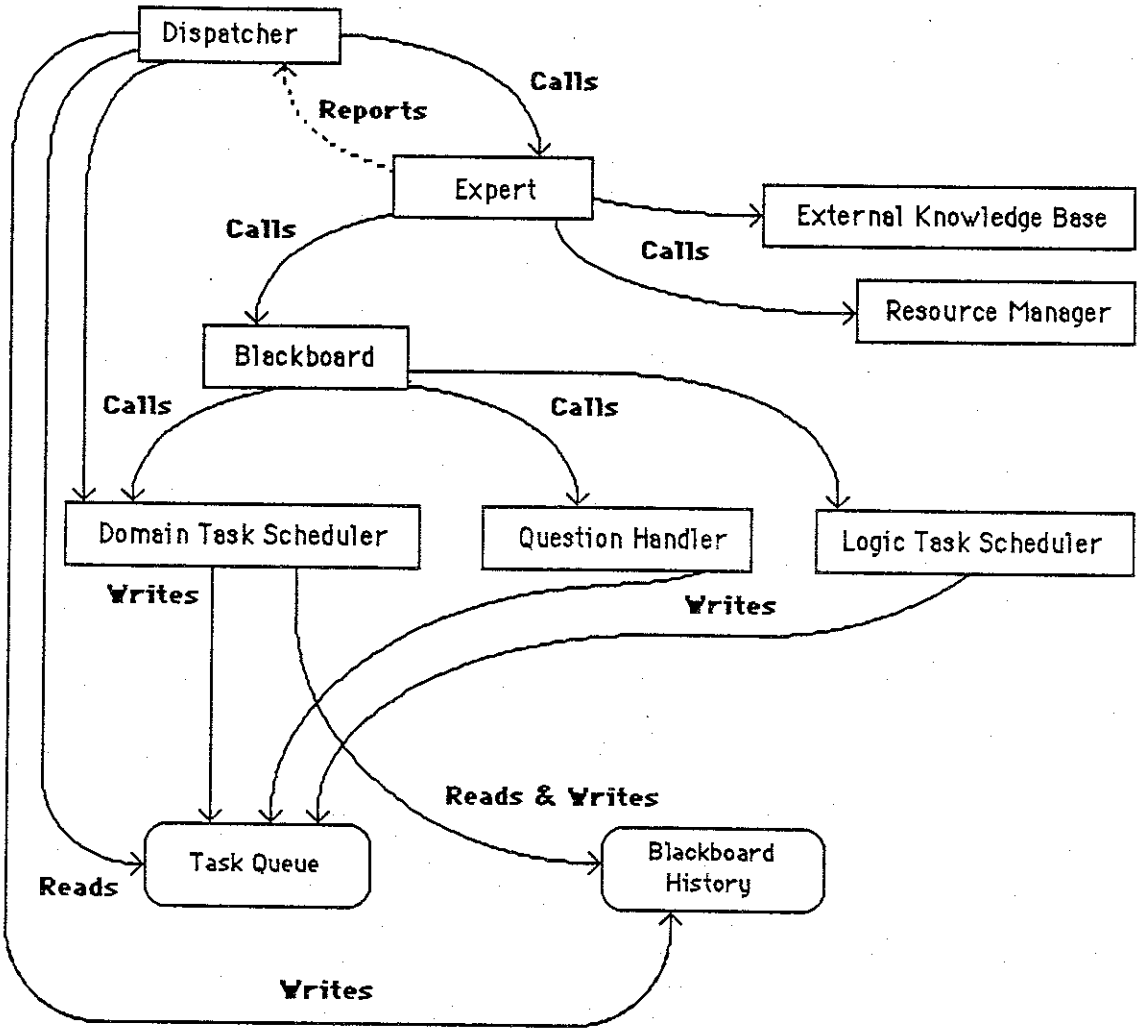

Figure 7. Calls for the Blackboard / Strategist, Experts

I. Blackboard/Strategist Complex

KEY: DTS = domain task scheduler
PAM = posting area manager
TD = task dispatcher
LTS = logic task scheduler
QAH = question/answer handler

A) PAM: maintain the integrity of the posting areas

Calls Available to All Modules	Calls Resulting To:
post_hypothesis(Hypothesis, Area).	LTS, DTS
retract_hypothesis(Hypothesis-id).	LTS
post_question(Question).	QAH
post_answer(Question-id, Answer).	QAH
retract_answer(Question-id, Answer-id).	
view_area(Area, Hypothesis-set).	
view_questions(Question-set).	
view_answers(Question-id, Answer-set).	
view_pending(Hypothesis-set).	
Calls Available to Strategist Modules	**Calls Resulting To:**
post_pending(Hypothesis).	LTS
retract_pending(Hypothesis-id).	LTS
retract_question(Question-id, Answer-set).	

B) LTS: maintain consistency of deduction trees for hypotheses

Calls Available to Posting Area Manager	Calls Resulting To:
new_dependencies(Hyp-id, Rel-head-of-hyp, Deps).	
hyp_retracted(Hypothesis-id, Change-in-confidence).	TD
hyp_replaced(Hypothesis-id, Change-in-confidence).	TD

Processing

Inputs: dependency information from posting area manager

Rules: based on age of hypotheses, number of dependencies, absolute values of confidence, relative change in confidence

Actions: call TD by "new_task(Task)." for dependent-hyp HYP-D where Task is [Expert-id-of-HYP-D, attempt_hyp(head(HYP-D)), LTS, Priority, Timestamp]

C) QAH: schedule follow-up action when a new question or answer arrives

Calls Available to Posting Area Manager	Calls Resulting To:
new_qustion(Question-id, Relation-head).	TD
new_answer(Question-id, Answer-id, Confidence).	TD

Processing

Questions:

Use facts of form

can_answer(Area, Relation-head, Expert, Order-to-call-this-expert).

to decide for a new question what expert(s) to ask the LTS to send calls to

attend_quest(...).

Answers:

Use prior posted answers and facts of form "can_answer(...)." and

answer_conf-no_thresh(Area, Relation-head, Min-conf, Min-no).

to decide if should ask LTS to get more answers, or if should ask PAM to retract the question and LTS to have original questioner "view_answers."

Figure 7. Calls for the Blackboard / Strategist, Experts - continued

D) DTS: schedule new tasks based on mix of hypotheses on blackboard, session history

Calls Available to Posting Area Manager	Calls Resulting To:
new_hyp(Hypothesis-id, Relation-head).	LTS

Processing

Stimuli for action

Arrival of a new hypothesis

Task queue empty or all tasks of very low priority

Posting to pending hypothesis area

When have refinement of former pending hypothesis

When satisfy rules of form

post_pend(Area, Rel-head, Min-conf, Max-already-posted).

Schedule appropriate tasks, in order based on specific rules as well as:

1) amount of user waiting, e.g., examine

user_need(Max_user_wait, Min_user_cert, Last_user_input, Expert-id).

2) overall state of processing, e.g.,

next_phase(Current_session_state, New_session_state, Expert_id, Priority).

2) who handles each area, as given by

expert_area(Expert, Area, Priority).

E) TD: funnels tasks from other scheduler units to experts based on priorities and resources

Call Available to Scheduling Units	Actions Taken
new_task(Task).	Adds task to queue.
Calls Available to Experts	
done(Expert-id).	Log task complete.
checkpoint(Expert-id, Checkpoint-id).	Log - possibly preempt.

Processing

Maintains a queue at each priority level.

Maintains history file of progress of all tasks of current session.

Maintains record of availability of all resources — ex. experts, machines, fact bases.

Passes work to experts based on priorities and resources.

II. Canonical Expert

Calls Available to Strategist	Actions Taken: Try
attempt_hyp(Rel-head).	to produce hypotheses with this Rel-head.
attend_to_area(Area).	any processing relating to Area.
attend_to_quest(Rel-head).	to answer questions with this Rel-head.
answers(Question-id, Answer-set).	complete task leading to the question.
wake.	all tasks, with "checkpoint"s between, and then send "done".
abort.	stop all processing.

Calls to the Strategist To Convey Status

done(Expert-id).

checkpoint(Expert-id, Checkpoint-id).

Figure 8. Implementation levels of intermodule communication.

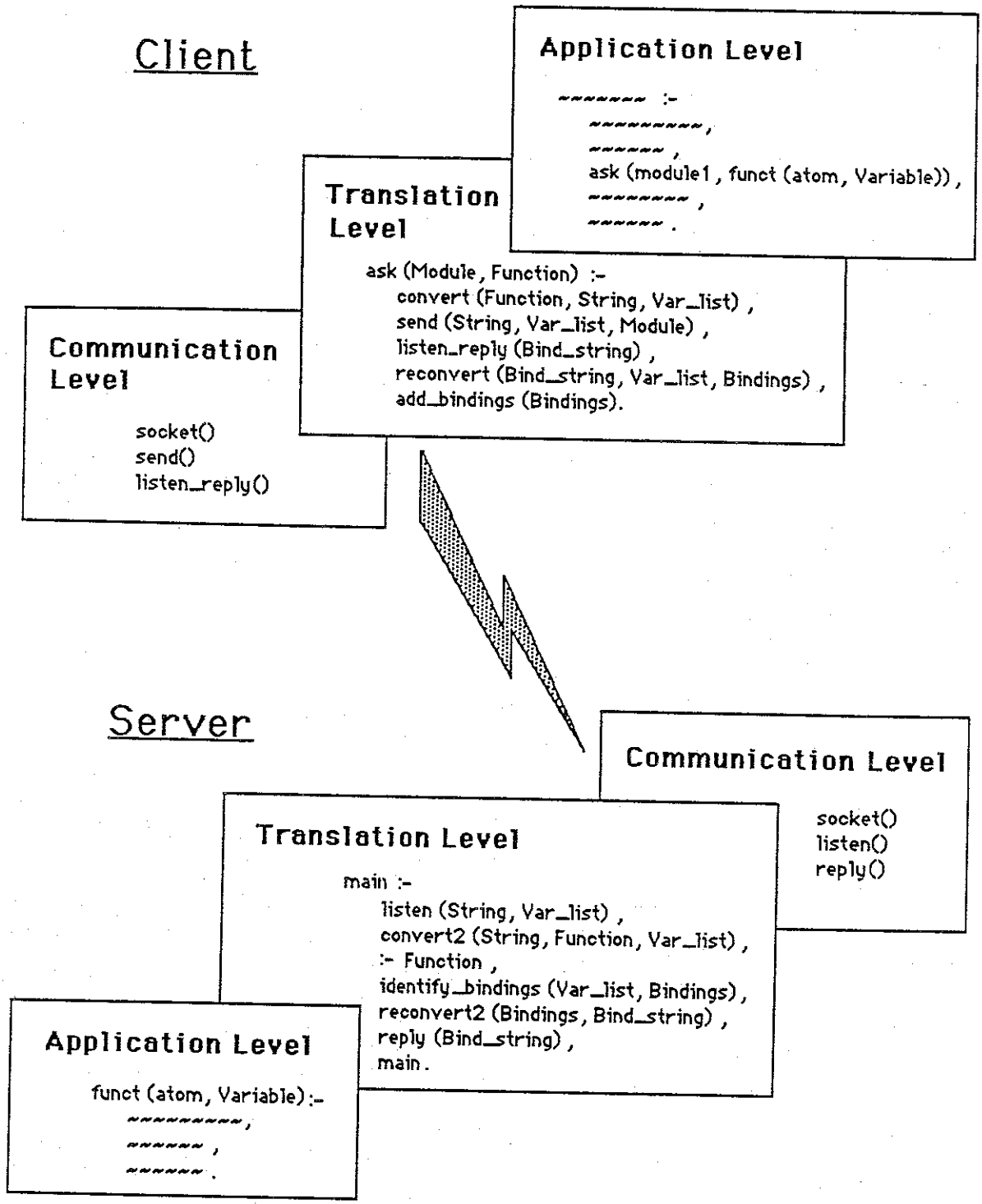

Figure 9. Type hierarchy for common classes of documents.

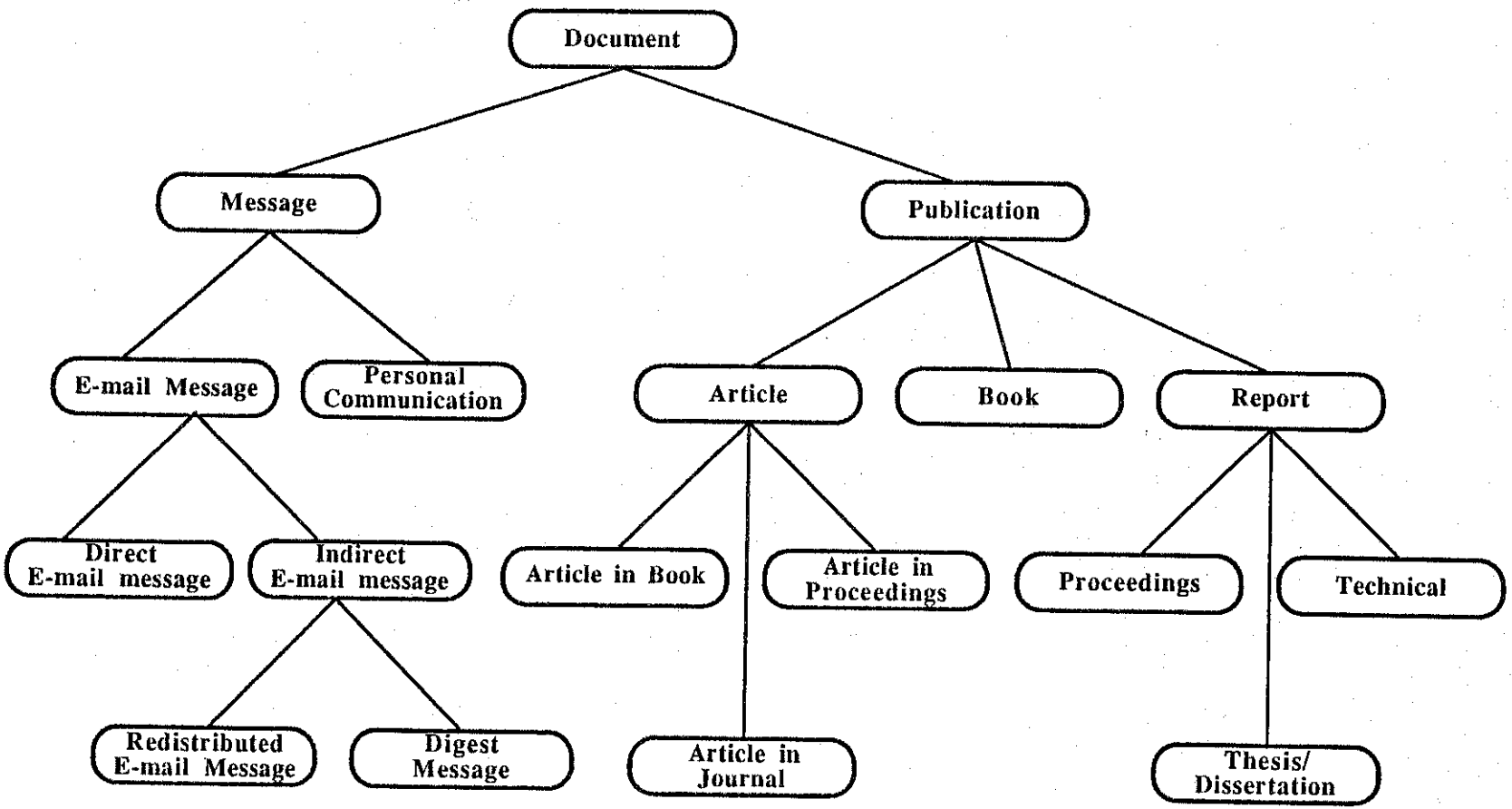

Figure 10. Topical hierarchy for digest documents

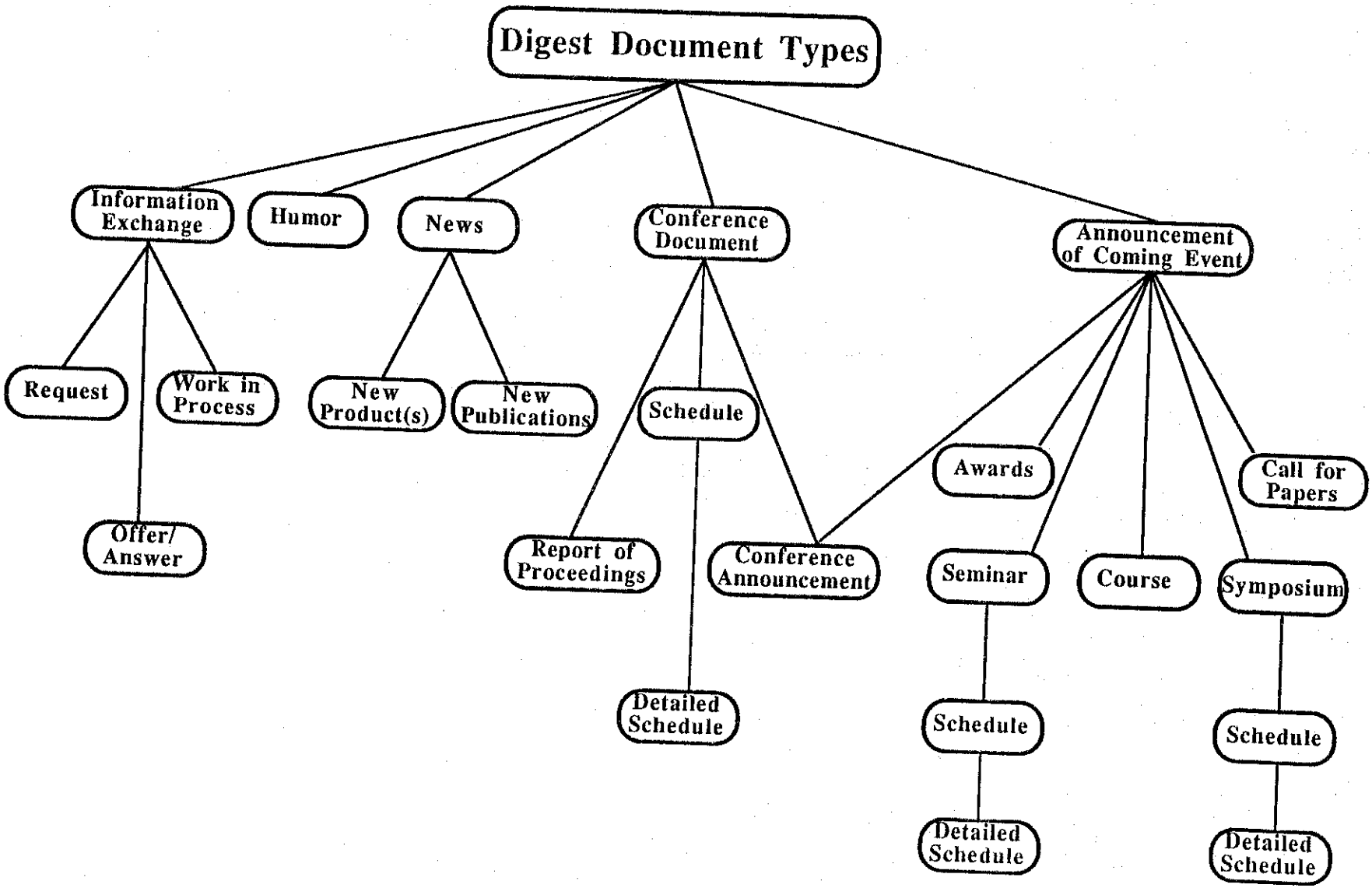

Figure 11. P-norm Search Expert

A) Output: Hypothesis for Relevant-Doc Area

The hypothesis data structure is a 5-tuple:

<fact,confidence,search-expert-id,Id,dependencies>

where

fact = a relevant document id
confidence = the similarity value (real value in unit interval)
Id = a variable to which a hypothesis id will be assigned by blackboard
dependencies = the dependent hypotheses on which this posting is based

B) Input: Document Information from Inverted File

The search expert obtains doc-id/wt pairs for a term by using the ASK predicate:

ask(doc_base,facts_with_rel(rel_id(Qterm),Id_wgt_list))

where

doc_base: identifies the document data base;
Qterm: is instantiated to the concept-type/number representing a query term
facts_with_rel: is a function of each external knowledge base; this form asks for a list of facts from the "rel-id" relation, to retrieve doc ids and weights for the given query term
Id_wgt_list: is a variable to which the list of ids and weights will be bound

C) Input: P-norm Query

The search expert is awakened, perhaps as a result of the retrieval strategist command

attend_to_area(pnorm_query)

and can then ask the posting area manager of the blackboard to provide it with all pending p-norm queries by ASKing to

view_area(pnorm_query,Hyp_set) predicate.

The hypothesis set contains a hypothesis-fact in form:

[pnorm_search,Num_docs,Query]

where Num_docs bounds the number of documents desired, and Query is of form

[weight,AND-or-OR,p_value,clause_list]

where

weight = relative weight for this clause if embedded in outer clause
clause_list = a list containing [term,wt] pairs or other (embedded) queries

D) P-norm Processing: Partial Calling Hierarchy

ATTEND_TO_AREA - call for processing that must be provided for use by stategist
VIEW_AREA - to retrieve hypotheses from blackboard
PROCESS_HYP - to recursively select applicable hypotheses
PROCESS_SEARCH - to process a search goal, if found
LOOKUP - access inverted file
...
ORDER_SIM - to sort the document, similarity (confidence) list
DOCS_TO_POST - select best num_doc documents as specified in query
POST_HYPOTHESES - posts each selected doc-id and similarity value
DONE - notifies the strategist that search task is complete

Figure 12. Time and date forms in message headers

A) Partial list of date patterns recognizable by current parser
1. Month Day, Year
2. Month Day, Year
3. Month / Day Year
4. Day - Month Year
5. Day - Month - Year
6. Day, Day_of_month
7. Year
8. Month Day Year
9. Month / Day / Year
10. Day_of_week Month / Day /Year

B) Partial list of time patterns recognizable by current parser
1. hour o'clock
2. hour o'clock a-pm
3. hour a-pm
4. hour : minute : second
5. hour : minute : second a-pm
6. hour a-pm zone

C) Sample of date-line facts from the AIList messages.
Monday, April 25, 1983 5:27PM
Mon 25 Apr 83 14:51:42-PDT
Mon 25 Apr 83 09:34:04-PDT
22 Apr 1983 0227-EST
Thursday, 21-Apr-83 15:23:45-BST
Sun 24 Apr 83 20:41:46-PDT
Sunday, May 1, 1983 11:00AM
Thu 28 Apr 83 14:40:26-PDT

Figure 13. Time word/phrase lists

A) By part of speech (in *CDEL*)

1. Adverb, proposition and conjunction

after	any	around	at	before	between
by	coming	during	early	following	for
from	from...to...	in	later	through	throughout
till	to	towards	since	until	within

2. Noun

<day>	<month>	<season>	<unit-of-time>		
A.D.	B.C.	afternoon	anteriority	century	date
duration	dusk	epoch	era	evening	fortnight
future	moment	morning	night	noon	period
posteriority	season	temporal	time	today	tomorrow
twilight	yesterday				

B) By duration

1. Definite duration or portion of time

age	decade	era	lifetime	period	semester

2. Indefinite duration

flow	lapse	march	progress		

3. Long duration

durability	longevity	permanence	persistence	stability	standing

4. Short duration

brief	fleeting	momentary	passing	soon	temporaily

5. Endless duration

always	ceaseless	deathless	endless	eternal	forever

6. Point of time

abrupt	at once	in no time	sudden		

C) Past, present, future

1. Past

days of old	days of yore	history	past time	remote time	yesterday

2. Present

actual	at this moment	instant	time being	twentieth century	

3. Simultaneousness

coexist	coincident	concurrent	contemporary	together	synchronous

4. Future

future	hereafter	morrow	time to come	tomorrow	

Figure 14. Parsing prepositional phrases, and temporal reasoning

A) Prepositional phrases can be parsed using the grammer:

PP --> (PREP) (DET) (ADJ)* N

N ::= Nouns determine the BASE UNIT of the meaning. N should be a time noun, or an explicit date or time representation. Events implicitly indicate time intervals.

ADJ ::= Counting adj. like " first, second, ..., middle, last". They contribute the OFFSET; or
::= Present, past adj. like "last", "present", "coming". They contribute the DIRECTION.
Future time has + direction. Past time has - direction. Present time has no direction.
Past: antiquated, bygone, elapsed, expired, extinct, forgotten, gone, lapsed, obsolete, outworn, over, passed, past
Present: actual, current, existing, instant, latest, present
Future: coming, eventual, following, future, imminent, impending, near, next, prospective, ultimate

DET ::= Determiners indicate definiteness, quantification. They may allow determination of the exact interval involved; thus "this year" currently means 1986.

PREP ::= Prepositions indicate how an interval should be built from the PP elements. When a PP occurs alone, the PREP indicates before, during, or after RELATIONSHIPs.
Before: before,of,through,to,toward,until
During: at,during,in,through,throughout,within
After: after, from, since
Fuzziness can lead to ambiguity among these, as in "about noon."

Therefore, a frame representing a PP must have slots, such as for " in the following three hours":
Relationship: during
Base Unit: hour
Direction: +
Offset: 3
Determiner: definite

B) Reasoning with intervals

1. Document processing
   Use header field time stamps to relate digests, messages to time line
   Use explicit dates in body of message to relate nearby text objects to time line
   Use inter-message references to build lattice of time relationships
2. Query processing
   Build frames for all intervals mentioned
   Determine fuzziness allowed in matching intervals
   Search using: time line, reference intervals, or relationship lattices
3. Refinements
   To support more complex query forms, it is necessary to deal with reference intervals for sequences of PPs. The preposition "of" often leads to a reference interval, as can be seen from "in the last session of the conference of 1986." Methods for analyzing the frames produced from such PP sequences are under development.

Figure 15. User interaction and user information

A) Phases of system aid to user
- Logging on
- Welcoming
- Offering
  - Tutorials
  - Help
  - Explanations
- Gathering information on
  - User model
  - Problem state
  - Problem description
- Query
  - constructing
  - revising
- Browsing (and getting feedback on) text and knowledge bases
  - *HAI*
  - Dictionaries
  - Documents
  - Users
- Saving/printing results
- Requesting assessment and suggestions on system behavior

B) Background information collected for user model
- Reason for search
- Academic level
- Linguistic ability
  - English is native language
- Experience with
  - Computer's operating system
  - Information retrieval systems
- Courses
  - Information retrieval
  - AI

C) Problem state
- Whether general or specific topic
- Whether browsing, searching, or re-locating known object
- Whether continuing earlier search

D) Problem description
- Topic, according to *HAI* contents (Table 2)
- Type of document/passage desired, according to digest document type hierarchy (Fig. 10)
- No. of items desired, as related to recall/precision needs
- English prose description of document/passage desired

E) Evaluation
- Recall, precision
- Satistifaction with document parts or wholes
- Frustrations
- Reasons for stopping search

Table 1. Approximate Statistics on External Knowledge Bases

A) AIList Document Collection	
Number of messages:	5750
Number of authors:	300
Number of digest issues:	750
Dates covered:	4/83-11/86
Characters of text:	10Mbytes
Concepts (in SMART)—	
Word stems,proper names:	25K
Total:	32K
SMART collection sizes —	
Document vectors:	4Mbytes
Inverted file:	5Mbytes

B) Handbook of Artificial Intelligence	
Number of files:	106
Characters of text:	4Mbytes
Number of Table of Contents subjects:	218
Number of Index entries:	853
Number of Index range-entries:	158
Number of Index person names:	138
Number of italicized words/phrases:	5009

C) Collins Dictionary of the English Language	
Number of relations produced:	21
Number of headwords:	85K
Number of different parts of speech:	46
Number of categories (used >=30 times):	120
Number of definitions:	165K
Number of morph. variants:	28K
Number of usage samples:	17K
Number of comparisons:	8K
Numbers for parts of speech—	
Nouns:	63K
Verbs:	15K
Adjectives:	13K
Adverbs:	1300

Table 2. Volume/Chapter Structure of *HAI*

HANDBOOK OF ARTIFICIAL INTELLIGENCE

VOLUME I by Avron Barr and Edward A. Feigenbaum
- I. Introduction
- II. Search
- III. Knowledge Representation
- IV. Understanding Natural Language
- V. Understanding Spoken Language

VOLUME II by Avron Barr and Edward A. Feigenbaum
- VI. Programming Languages for AI Research
- VII. Applications-oriented AI Research: Science
- VIII. Applications-oriented AI Research: Medicine
- IX. Applications-oriented AI Research: Education
- X. Automatic Programming

VOLUME III by Paul R. Cohen and Edward A. Feigenbaum
- XI. Models of Cognition
- XII. Automatic Deduction
- XIII. Vision
- XIV. Learning and Inductive Inference
- XV. Planning and Problem Solving

Table 3. *HAI* Subject Hierarchy

A) List representation

```
aih_hier(
[0,
        [1,
                [2,3,4],
        6,
                [7,8,
                        [9,10,11],
                12,
                        [13,14,15,
                                [16,17,18,19],
                        20,21,
                                [22,23,24]],
                25,
                        [26,27,28,29,30,31]], ...
```

B) Subject facts

```
% Subject facts provide a number for each 'subject' in HAI.  A subject is an entry in the table of
%   contents outline; it includes chapter titles, subheadings, etc.  It is of form:
%               subject(subject-number,subject-title).
% There are about 220 subjects; numbers are skipped and are used in the hierarchy structure above.

subject(0,'handbook of artificial intelligence').
subject(1,'introduction').
subject(2,'artificial intelligence').
subject(3,'the ai handbook').
subject(4,'the ai literature').
subject(6,'search').
subject(7,'overview 1').
subject(8,'problem representation').
subject(9,'state-space representation').
subject(10,'problem-reduction representation').
subject(11,'game trees').
subject(12,'search methods').
subject(13,'blind state-space search').
subject(14,'blind and/or graph search').
subject(15,'heuristic state-space search').
subject(16,'basic concepts in heuristic search').
subject(17,'a*--optimal search for an optimal solution').
subject(18,'relaxing the optimality requirement').
subject(19,'bidirectional search').
subject(20,'heuristic search of an and/or graph').
subject(21,'game tree search').
subject(22,'minimax procedure').
subject(23,'alpha-beta pruning').
subject(24,'heuristics in game tree search').
subject(25,'sample search programs').
subject(26,'logic theorist').
subject(27,'general problem solver 1').
subject(28,'gelernters geometry theorem-proving machine').
subject(29,'symbolic integration programs').
subject(30,'strips').
subject(31,'abstrips').
subject(33,'knowledge representation').
```

Table 4. Relations derived from the *HAI* back-of-the-book indexes

A) References to line of text in *HAI* for word/phrase in index

```
% index_ref([file_number,line_number],index_entry).
index_ref([4,33],'natural language').
index_ref([4,92],'logic').
index_ref([4,93],'computation').
index_ref([4,108],'turing, a.').
index_ref([4,125],'cybernetics').
index_ref([4,146],'computers').
index_ref([4,147],'computational complexity').
index_ref([4,211],'chess').
index_ref([4,212],'search').
index_ref([4,286],'problem solving').
index_ref([4,298],'problem representation').
```

B) References to range of lines of text in *HAI* for word/phrase in index

```
% index_rng([file_number_1,line_number_1],
%          [file_number_2,line_number_2],index_entry).
index_rng([4,4],[4,485],'intelligence').
index_rng([4,91],[4,233],'early ai').
index_rng([8,75],[8,444],'problem representation').
index_rng([8,155],[8,276],'forward reasoning').
index_rng([8,156],[8,277],'backward reasoning').
index_rng([8,318],[8,443],'search space').
index_rng([8,451],[8,582],'heuristics').
index_rng([8,521],[8,581],'heuristic search').
index_rng([8,522],[8,580],'blind search').
index_rng([8,537],[8,562],'generate-and-test').
```

C) Person names extracted from above index relations

```
% aih_person('person_name')
aih_person('abelson, robert').
aih_person('adelson-velskiy, g. m.').
aih_person('amarel, s.').
aih_person('anderson, j.').
aih_person('artsouni, g. b.').
aih_person('atkin, l. r.').
```

Table 5. Italicized words/phrases in text of *HAI*

```
% italics_ref([file_number,line_number],'italicized phrase').
italics_ref([70,73],'environmental').
italics_ref([70,137],'courseware author').
italics_ref([70,141],'individualization of instruction').
italics_ref([70,145],'frame').
italics_ref([70,235],'teacher').
italics_ref([70,244],'less powerful').
italics_ref([70,250],'production rules').
italics_ref([70,274],'tutoring strategies').
italics_ref([70,283],'international journal of man-machine studies').
```