Natural Language Techniques for Intelligent Information Retrieval

Paul S. Jacobs and **Lisa F. Rau**
Artificial Intelligence Program
GE Research and Development Center
Schenectady, NY 12301 USA

Abstract

Neither natural language processing nor information retrieval is any longer a young field, but the two areas have yet to achieve a graceful interaction. Mainly, the reason for this incompatibility is that information retrieval technology depends upon relatively simple but robust methods, while natural language processing involves complex knowledge-based systems that have never approached robustness. We provide an analysis of areas in which natural language and information retrieval come together, and describe a system that joins the two fields by combining technology, choice of application area, and knowledge acquisition techniques.

1 Introduction

Natural language processing (NLP), the use of computers to extract information from input in everyday language, has begun to come of age. Commercial natural language interfaces are available in many price ranges [HARRIS84], other software systems (from Semantec's Q&A to Lotus' HAL) use embedded natural language technology, and large government and industrial contracts combine basic research in natural language with real applications [SONDHEIMER86]. At the same time, the meteoric increase in the availability of on-line text motivates better methods for accessing information in text form. But the use of natural language for intelligent access to this information is still not practical, primarily because of the limitations of current natural language systems. Overcoming these limitations demands a combination of technology, selection of application, and knowledge acquisition.

Two types of natural language technology can be useful in information retrieval: (1) *Natural Language Interfaces* make the task of communicating with the information source easier, allowing a system to respond to a range of inputs, possibly from inexperienced users, and to produce more customized output. (2) *Natural Language Text Processing* allows a system to scan the source texts, either to retrieve particular information or to derive knowledge structures that may be used in accessing information from the texts.

Of these two technologies, interfaces are far more mature than text processing systems. However, it may not be possible to take complete advantage of natural language interfaces in information retrieval without some text processing, because the amount of information that can be derived from a natural language input is richer than that which can be used for conventional retrieval. In other words, understanding natural language input and producing natural language output may be of little value without understanding the source text.

The nature of an application is important for using natural language in information retrieval. Especially relevant is whether the task is to locate a particular document, or to find some piece of information that may be contained in the document. For example, literature search and legal reference are problems in document retrieval, while a help facility for a computer system or a decision support system are information systems in a broader sense. Text processing is more important, and thus natural language is more relevant, to these more general information needs.

A major difference between a natural language based approach and more conventional information retrieval (IR) methods is that natural language systems require considerable knowledge acquisition. While text scanning can conceivably be done simply by comparing words in an input query to words in a document, natural language understanding requires an existing vocabulary of words and their meanings, even in a simple application. In order for such a vocabulary to be useful, there must also be knowledge about how the concepts described by the words relate to one another.

The three main considerations in applying NLP to IR are thus (1) the selection of NL technologies, (2) the choice of application, and (3) the approach to knowledge acquisition. The sections that follow will consider some of the NL and text processing technologies that are available. Section 4 describes our approach in all three categories, and presents some of the tasks that can be accomplished by applying natural language techniques to a constrained IR problem.

2 Natural Language Interfaces

The more polished, commercial applications of natural language currently translate user input into a database query or other internal form. This performs two useful functions: (1) It reduces training time and makes systems accessible to users who would not learn a specialized language, and (2) If there are multiple information sources in the system, possibly using different query languages or access methods, the user is insulated from the particulars of each knowledge source.

Translation of natural language input into system queries is only one aspect of the interface problem. In environments where the user asks questions of the system, a cooperative response involves not only comprehension of the question, but understanding the user's intent and identifying misconceptions the user may have. If the response is to be expressed in natural language, the production of this response is yet another natural language problem. Three main areas of natural language research, therefore, are relevant for intelligent access to information systems: (1) *Input translation* is the basic parsing problem—processing the input to produce a system query, (2) *Question interpretation* is the understanding of the user's knowledge and goals in asking a particular question, and (3) *Response formation* is the problem of answering a question in natural language. Each of these will be considered in the sections that follow.

2.1 Input Translation

Most natural language database query systems [GROSZ85,GINSPARG83,BALLARD84] map natural language into an internal representation, such as predicate logic, and use this representation to structure the query. The advantage of this approach over mapping directly from natural language to query language is that the language interpretation mechanism can be *transportable*; that is, it can be used in multiple domains with different query languages. They are not, however, fully transportable, because each database domain has its own vocabulary that must be related to the system's knowledge about language. To illustrate the difficulties in relating English to system queries, consider the following two hypothetical inputs:

```
(1) User: Give me the sales managers in West Coast operations.
    System interpretation:
         Find ( MANAGERS with domain SALES )
         with division ( OPERATIONS with location WEST-COAST )

(2) User: Give me the senior managers in alphabetical order.
    System interpretation:
         List alphabetically ( MANAGERS with level >14 )
```

The system interpretations given above represent some of what must be extracted from the input in order to structure a query. The two queries have virtually identical linguistic structures, but have very different interpretations. In the first case, *in West Coast operations* describes *managers*, while in the second *in alphabetical order* describes how the results of the query should be given. In order to distinguish the two, the natural language analyzer must understand what *West Coast operations* and *alphabetical order* mean. Other phrases, such as *senior manager* and *sales manager*, must also be defined. No natural language system has a broad enough vocabulary to understand inputs such as these without being specifically trained, thus each interface application currently requires considerable training. Trying to reduce the amount of domain-specific vocabulary building is one aspect of the *knowledge acquisition* problem, and is probably the largest hurdle for current natural language interfaces [JACOBS86].

These examples are drawn from database interactions, but the problems are common to queries in information systems in general. The examples above illustrate the difficulty in finding a correct literal interpretation of a query; the next section considers some of the increased complexity of indirect or inaccurate requests.

2.2 Question Interpretation

Directly interpreting a user's input does not always produce an adequate response, particularly when the input (1) requires summarization, (2) reflects user misconceptions or mistakes, or (3) does not directly match the information source. For example, here are some exchanges, with the direct answer marked by - and the preferable response indicated by +:

```
(3) User: What managers earn more than $20,000?
    - : Abel, Ackerman, Albert, Allen, ...
    + : All of them.
```

```
(4) User: Who sold over $1M in 986?
    - : No one.
    + : I have no records for 986.  Did you mean 1986?

(5) User: What defense companies have the greatest earnings?
    - : General Dynamics, Rockwell International, ...
    + : Including defense earnings of non-defense companies, the
        greatest earnings are General Dynamics, General Electric
        (Electronics), Lockheed (Aerospace), Rockwell, ...
```

In these cases, the preferable responses require assumptions about the user's intentions in asking a question. A system can be engineered to tolerate inaccurate questions. However, the general issue of responding to the *intended* meaning, rather than the *literal* meaning of a question, is a difficult research problem.

2.3 Response Formation

Information systems generally produce responses in a rigid form, such as a table constructed from a database or a text accessed by a document retrieval program. When these systems become able to respond to a broader range of questions, they must be able to produce more natural responses, and it becomes necessary to use natural language generation as the output of the system. For example, inaccurate questions such as those above demand a natural explanation. Similarly, a user may ask a question *about* the database, rather than asking for a piece of information from the database [MCKEOWN82]. In asking questions about textual databases, a user may request information that demands a summary or inference from the text. Natural language seems necessary for expressing this sort of response. In all these cases, much of the power of an information system depends on the flexibility of the system's response.

This discussion has covered some of the general problems in accessing information using natural language. The next section covers the use of natural language techniques to extract information from texts.

3 Text Processing

Unlike understanding inputs to an interface, text processing requires the analysis of extended texts in a less constrained context. In natural language interfaces, the user is almost always looking for a particular piece of information. Free-form texts are generally more complex: a text can describe a process, tell a story, or analyze data. Extended texts also make continual references to previous mentions of objects and events, as well as using longer sentences and a broader vocabulary. For these reasons, full-scale automatic text understanding will not be accomplished in the near future.

Of course, information can be *extracted* from texts without full understanding, as evidenced by the partial success of certain statistical methods as well as some text-skimming work [DEJONG79a]. There are several ways in which natural language techniques may prove worthwhile in retrieving information from texts: (1) Text scanning can be used to create databases (or knowledge bases) from the texts, assuming the texts and the databases are sufficiently constrained. (2) The texts may be preprocessed and certain features or keywords extracted can then be used in information access. (3) At the time

when a question is asked, the text can be skimmed to try to find the response in the text (without preprocessing of the texts).

Each of these three approaches will be described briefly in the discussion that follows.

3.1 Database Generation

The appealing characteristic of database generation is that, if a database can be constructed automatically from text, a wealth of tools can then be used to access the database. Also, a database that is manually created can be updated automatically from text sources. The problem does not appear unmanageable because databases contain only well-structured information, and much of the extraneous material in the texts can be ignored.

Database generation is practical with constrained texts in constrained domains, and where only certain information must be extracted. For example, it has been applied to the automatic processing of radiology reports [SAGER78], banking telexes [LYTINEN86] and weather reports [KITTREDGE82]. Of course, if the database is a highly structured one (as opposed to an AI knowledge base, which can contain loosely structured information), only the information that fits the database can be effectively obtained.

The most common way of performing this operation is using *case frames*, structures that relate a key concept to other related concepts and their semantic properties, to fit the information from a text into a database structure. Case frames organize information by constraining the roles through which a knowledge structure is related to other structures. For example, in a banking telex, the roles in a transaction case frame may be SOURCE, DESTINATION, AMOUNT, and TYPE-OF-TRANSACTION. *Expecting* this structure makes the task of processing a text much easier.

3.2 Index Generation

Natural language can be used for automatically creating indices for accessing texts by extracting critical information from the texts [DILLON83,JONAK84], or by extracting indices from a user request [JONES84]. In the first case, texts are typically pre-indexed. These indices may range from simple words or keywords to complex conceptual hierarchies. In the second case, terms are extracted from a natural language query after it has been made, and these terms may be compared directly to the texts, or to indices created for the texts.

In general, these techniques have shown only a marginal improvement over finely-tuned statistical methods of traditional document retrieval [SALTON86]. However many of the limitations of automatic indexing techniques are due to limitations in language processing methods, and there is some evidence that more complete linguistic analysis, especially using a combination of syntactic and semantic methods, can produce potentially large improvements over statistical techniques [LEWIS88].

As an example, Fagan [FAGAN87] takes selected terms and produces English phrases by using some knowledge of syntax, and various heuristics to limit the phrases generated to contentful ones. However his experiments did not illustrate improved performance over statistically generated phrases. Although there was greatly improved performance for certain phrases generated, this was offset by poor performance for other phrases. This behavior indicates that improvements to the phrase construction process could yield consistent good performance with this type of approach.

3.3 Text Skimming

A variety of strategies are useful for text processing, depending on the type of application, the novelty of the input text, and the robustness of the system grammar and lexicon. Most of these strategies may be loosely assigned to one of the following three categories, although some systems contain elements of more than one category.

- *Full Parsing.* If the application domain is highly constrained, and the system knowledge base is extensive, it may be possible to process every word in the text and find its place in the meaning interpretation of the input. This type of approach may be practical in reading weather reports, technical abstracts, and certain other confined applications, as described in Section 3.1.

- *Partial Parsing.* In most applications, it is not possible, using current natural language technology, to recognize each word in a text and account for its role in the input. Most text processing, therefore, relies on the "fail soft" approach of partially processing the input, tolerating unknown elements as much as possible but relying heavily on domain knowledge to make up for gaps in linguistic knowledge. This approach has been used, for example, in reading news stories [DEJONG79b].

- *Text Skimming.* Partial parsing can be used to generate as complete as possible a representation of the input text, while in text skimming the goal is to extract a particular piece of information or look for only new information. This strategy is practical when reading a recount of a news story that has already been read, or in skimming a text to find the answer to a particular question.

Full text search is text skimming in the extreme, as the only goal of the search is to find a particular word. Of the three classes above, text skimming has received the least attention from natural language research. The preponderance of research in natural language has concentrated on full parsing, while more text processing work has emphasized partial parsing. The reason for this emphasis is that, in practice, it is difficult for a system to recognize what is relevant or what is already known without processing the texts as completely as possible. In order to accomplish text skimming, a system must automatically put itself into "skimming mode" when it recognizes this situation. In theory, this obstacle should be surmountable, and it should then not be necessary to preprocess all texts in order to allow for natural language retrieval.

This discussion has described the state of the art in natural language interfaces and text processing technology, along with some applications of this technology to information retrieval. In the next section, the design and implementation of a particular text processing system will be presented, with a focus on its choice of application strategy, technical approach and acquisition strategy.

4 The SCISOR System

Currently, the improvements that natural language technology contributes to information retrieval are limited to the same sort of successes that NL has in database interfaces. That is, natural language makes the systems easier to use, but does not drastically change the type of information retrieved or the manner in which it is accessed. In the future, however, natural language should impact accuracy, information content, and ease of use.

SCISOR (System for Conceptual Information Summarization, Organization and Retrieval) is a prototype for the information retrieval system of the future [RAU87b,RAU87a]. SCISOR reads news stories about mergers and acquisitions from a newswire source, extracts information from the stories, and answers questions in English pertaining to what it has read.

The following are the choices made in SCISOR with respect to the three categories mentioned in the introduction:

- *Application Strategy.* The news stories handled by SCISOR generally cover events that may be easily stereotyped, such as takeovers, offers, anti-takeover plans, and stock changes. The stories are seldom entirely self contained, but depend on events covered in other related articles. Often a new article will repeat much of what has been covered in previous stories and then provide a new important detail. This makes traditional document retrieval impractical, because the result of a successful document search would be a set of stories tiresomely retelling and revising the same takeover event. By contrast, the result of retrieval using SCISOR is an updated summary of the sequence of events.

- *Technical Strategy.* SCISOR combines a bottom-up full parser TRUMP (TRansportable Understanding Mechanism Package) [JACOBS86], and a top-down skimming partial parser TRUMPET (TRUMP Expectation Tool) [RAU88], with a conceptual organization and retrieval system. TRUMP performs the analysis of both the input texts and the questions asked of SCISOR; however, the architecture of the system allows for conceptual expectations to guide the parsing of the texts with TRUMPET, while simple queries are processed using entirely bottom-up (TRUMP) methods. In other words, questions are processed robustly without reference to previous events, while texts require the integration of linguistic clues with knowledge about how the new information fits into an existing framework. This approach facilitates the processing of extended texts without requiring the complete linguistic coverage that is so difficult to achieve in practice. Once the information in texts has been put into a conceptual framework, SCISOR exploits this framework to perform flexible retrieval and summarization of the information.

- *Acquisition Strategy.* SCISOR now uses a basic lexicon of several thousand words, enough to cover much of the common English in the texts but not sufficient for many collocations, specialized terms, and names. This specialized language, in addition to a great deal of conceptual knowledge particular to the domain of corporate takeovers, must still be encoded by "brute force" knowledge engineering. While we are investigating a number of techniques for automated acquisition, the conceptual knowledge will for the forseeable future depend upon the encoding of knowledge by hand, and it is this limitation that makes domain-independent information retrieval difficult to obtain. Nevertheless, the combination of an extended lexicon with some automated acquisition allows for conceptual retrieval in limited but real domains.

The assumption that underlies SCISOR's system architecture is that, in constrained domains, it is within the capabilities of current-day technology to use a natural language interface to access a conceptual knowledge base derived through text analysis. Implementation of this type of system will demonstrate whether natural language technology can ultimately improve the usefulness and effectiveness of an information retrieval system. The sections that follow give an overview of the internals of SCISOR and discuss specific capabilities of the program that are difficult or impossible to duplicate in a traditional document retrieval system.

4.1 System Architecture

Figure 1 illustrates the architecture of the SCISOR system.

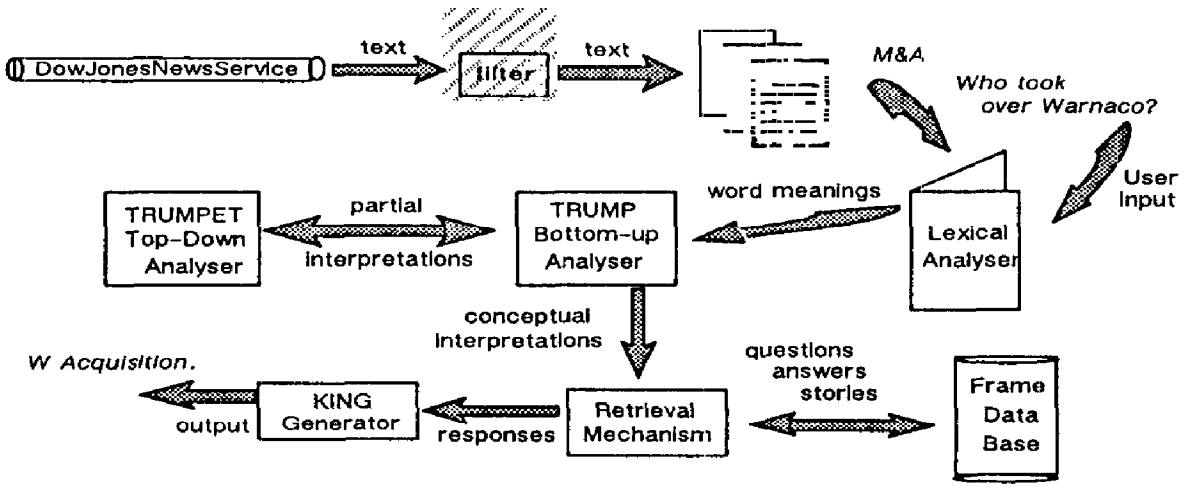

Figure 1: SCISOR System Architecture

Each of the boxes in Figure 1 represents a major component of SCISOR. First, newspaper stories, or questions about the stories that deal with corporate takeovers, are interpreted using TRUMP and TRUMPET. Input questions are parsed with the same understanding mechanism as is used for the input stories. The questions are stored along with the stories, for future user modeling, and to enable the system to answer a question when the answer later becomes known. This "automatic alert" capability is more fully described in Section 4.3.3. After answers to input questions have been retrieved, they are passed to the KING (Knowledge INtensive Generator) [JACOBS87] natural language generator for expression.

These stories are represented in the KODIAK [WILENSKY86] knowledge representation language. KODIAK has been augmented with some scriptal knowledge [SCHANK77] of typical sequences of events in the domain.

The following is a typical input to the SCISOR system:

Group Offers to Sweeten Warnaco Bid

```
April 8 - An investor group said yesterday that it is
prepared to raise its cash bid for Warnaco, Inc. from
$40 a share to at least $42.50, or $433.5 million, if
it can reach a merger agreement with the apparel maker.

The California-based group, called W Acquisition Corp.,
already had sweetened its hostile tender offer to $40 a
share from the $36 a share offered when the group launched
its bid in mid-March.
```

The discussion that follows covers some of the important elements of the SCISOR system in further detail.

4.2 TRUMP and TRUMPET

TRUMP (TRansportable Understanding Mechanism Package) is a suite of natural language processing tools. The system currently includes a parser, a semantic interpreter, and a set of lexical acquisition tools, although its theoretical foundation lies in work done on knowledge representation for natural language generation [JACOBS87]. Some of the background of TRUMP is described in [JACOBS86].

The TRUMP parser combines words into phrases and sentences, checks syntactic constraints, and instantiates linguistic relations. The system's acquisition strategy is to facilitate the addition of domain-specific vocabulary while preserving a core of linguistic knowledge across domains. Consider a phrase such as *the investor group seeking control of Warnaco*. TRUMP may be capable of parsing the phrase without much knowledge about corporate takeovers, but with such knowledge it will produce both a more precise interpretation and a better choice in the case of ambiguity. *The investor group* describes a group of investors, not a group that also invests; *seeking control* refers to the takeover objective, not really a "search"; *of Warnaco* means that Warnaco is to be controlled, and not that Warnaco is currently in control of something else. These decisions are obvious, yet they all actually require a great deal of knowledge on the part of the system. This knowledge is sometimes used to help parsing, but mostly it is used to produce more specific interpretations from general concepts.

TRUMP is designed to allow a variety of knowledge sources to contribute to the interpretation of the input. TRUMP does not really care where additional information is coming from, but tries to fit any more specific interpretations with what it has already produced. Thus the specific interpretation of *investor group* may be derived from a lexical entry for that compound nominal, while the interpretation of *seeking control* depends on domain knowledge about corporate takeovers.

TRUMPET incorporates a more "top-down" or expectation-driven mode of operation. In this mode, various sources of information are combined to aid in the extraction of expected pieces of information from text. The concretion mechanism integrates information from these multiple knowledge sources, deriving as complete an interpretation as possible. A more detailed description of this process may be found in [RAU88].

This top-down, or expectation-driven, approach, offers the benefit of being able to "skim" texts for particular pieces of information, passing gracefully over unknown words or constructs and ignoring some of the complexities of the language. A typical, although early, effort at skimming news stories was implemented in FRUMP [DEJONG79b], which accurately extracted certain conceptual information from texts in preselected topic areas. FRUMP proved that the expectation-driven strategy was useful for scanning texts in constrained domains. This strategy includes the banking telex readers TESS [YOUNG85] and ATRANS [LYTINEN86].

The combination of top-down (full parsing) and bottom-up (partial parsing) processing allows this text processing system to make use of all sources of information it may have in the understanding of texts in a domain. It can perform in-depth analysis when more complete grammatical and lexical knowledge is present, or perform more superficial analysis when faced with unknown words and constructs. This flexibility in processing is necessary to achieve robust text processing.

4.3 Information Retrieval in SCISOR

SCISOR provides capabilities for improving the usefulness of an information retrieval system by exploiting the conceptual nature of the information. In particular, SCISOR's

method of retrieval allows for close matching between the items of interest to a user and items actually present in the source documents. Also, conceptual information may be summarized. Finally, the system can alert users to items of interest as they appear in source documents. The implementation of each of these three capabilities will be described in more detail.

4.3.1 Conceptual Information Retrieval

SCISOR can answer questions when the questions provide information that may be *partial*, *similar*, or *wrong*. The following examples illustrate these three "anomalous states of knowledge".

ANSWER: **GE took over RCA.**

Partial: *What did GE do?*

Similar: *Did GE make an offer for RCA?*

Wrong: *Did RCA take over GE?*

This flexible method of retrieval is implemented with a form of constrained marker-passing [CHARNIAK83] and intersection search. In this technique, features in user requests are relaxed by passing markers heuristically up and down the abstraction (or "isa") hierarchy. Answers are located when enough markers intersect in a conceptual representation of a portion of a document. This method of information retrieval and some of its advantages are more fully described in [RAU87b].

4.3.2 Summarization

In application areas dealing with events that take place over time, the ability to summarize information is especially important. For a user interested in a concise summary, a large set of documents, each describing one event in an ongoing story plus a rehash of all events up to that point, might constitute acute information overload.

In the SCISOR system, summaries are produced by generating (in natural language) the contents of a particular category, such as CORPORATE-TAKEOVER-SCENARIO1. The "contents" of this category are the events that have taken place in this particular corporate takeover "script". To limit the depth of description, a limit is placed on the number of nodes that are passed to the language generator for expression. If the number of nodes at any given level of description is too large, other heuristics are used to capture generalities that can be expressed to the user. The simplest heuristic is to check if there are two or more concepts that are members of the same category. If this is the case, this generalization can be expressed more succinctly to the user.

In the following example, three components of the representation of the corporate takeover scenario are members of the LEGAL COMPLICATION category. This causes the sentence "..involving some potential conflict of interest matters..." to be generated in place of the details of those legal complications.

Figure 2 illustrates a portion of a corporate takeover scenario. The encircled concepts are the concepts that are to be expressed in the summary. This figure corresponds to the summary given below.

Summary: *Rumors that ACME was to be taken over started May 13, 1985. The stock rose $5 a share to $65. On May 16, the ACE company announced that it had made*

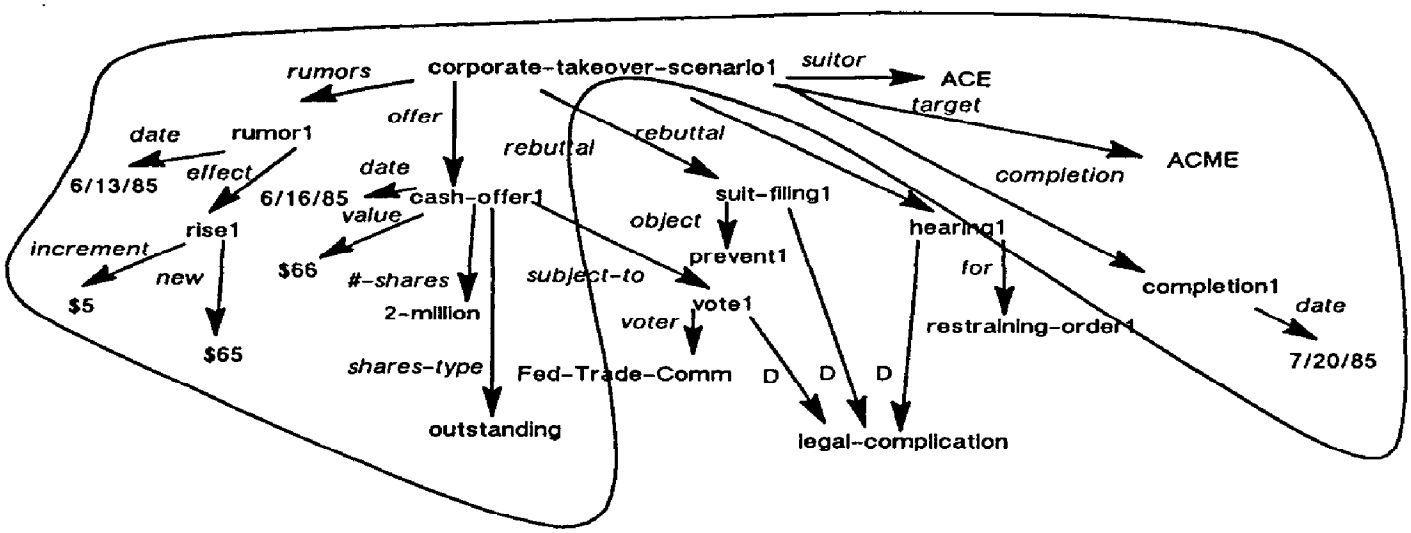

Figure 2: Story to be Summarized

an offer to ACME valued at $66 a share. The offer was a cash offer to purchase all 2 million outstanding shares of ACME. After a month of deliberation involving some potential conflict of interest matters, the ACE-ACME deal was completed last Friday.

With expository texts, as opposed to narratives, the conciseness of a summary is roughly a linear function of the distance of the concepts to be expressed from the containing abstract category. This is because the purpose of expository texts is to give facts. In narrative texts, a program must have a more complicated method of determining the relative importance of the events described in the story. Although the "linear function of the distance" heuristic does not always produce an optimal summary, it is a starting point for more complex heuristics.

One additional aspect that allows SCISOR to produce appropriate summaries is the presence of higher-level semantic representations to guide the descriptions expressed. For example, a very long article about a takeover may have many extraneous, historical details and surrounding context that might not be appropriate for a summary of that takeover. Although these details are included in the representation of the article, they are not used in summarization because of their conceptual distance from the corporate takeover script.

In order to guide the summary of an event, the user must refer to the event at an appropriate conceptual level. For example, to elicit the summary above, the user might request "Please give me a summary of the events in the ACE - ACME takeover". The program outputs the contents of any conceptual category that is inquired about by the user. If that category is a high level one, a summary will be produced. If it is very specific, something very specific such as an answer to a specific question will be produced. If the user asks for a "short" summary or an "extended" summary, SCISOR uses heuristics to compute a relative measure of length, based on the longest paths existing in the representations.

4.3.3 Automatic Alert

In addition to its conceptual matching and summarization capabilities, SCISOR has the additional feature of automatically alerting the user to new information of interest. If a

user has previously asked a question that remains active, the answer to the question may become available in later texts. In this case, the best conceptual match for the input information from the texts will be the active question (inactive questions are filtered out). This allows SCISOR to retrieve the active question and alert the user to the new answer:

```
PREVIOUSLY:

USER:   How much did ACE offer to take over ACME?
SYSTEM: No figures have been released yet.

LATER ON:

SYSTEM: ACE has just offered $5 per share for all outstanding
        shares of ACME.
```

In this example, the user is alerted to the new answer. The answer is stored along with the question. If the user does not wish the question to remain active, it is marked inactive. The user can keep questions active when there is no real answer, as in this case, or in other cases where continual updates to the answer are required. For example, an increased offer in the above example will result in a new alert and a new answer.

4.4 Other Related Research

Much of the research closely related to SCISOR has been mentioned during the previous sections. Two retrieval systems that have not been covered are also similar to SCISOR in some ways. The ADRENAL system of Croft and Lewis [CROFT87] uses natural language query processing, like SCISOR, to enhance document retrieval. Unlike SCISOR, however, this system uses natural language processing only for the queries, and is thus currently constrained to statistical methods for retrieval. Also, the result of retrieval in ADRENAL is a document, while SCISOR produces updates, direct answers, and summaries.

The GRANT system [COHEN87] uses a spreading activation mechanism similar to SCISOR's to perform conceptual information retrieval from a knowledge base. GRANT, however, has no natural language processing component, and thus does not provide many of the same functions. The most important functions of SCISOR, such as cooperative responses to questions, summaries, and alert capabilities, are dependent on the use of natural language for both text analysis and interfaces.

4.5 System Status

SCISOR is currently capable of reading news stories about corporate takeovers, extracting certain prespecified features from these stories, and answering questions about the information derived. The system can also take English inputs not in the form of specific questions and use these inputs as indices for retrieval of relevant information. This has the obvious advantages of a flexible natural language interface, allowing the user to use English to access the desired information. It also has the important feature of keeping constant track of stories as they unfold, rather than treating each text as a distinct document.

Two major enhancements remain to be made. First, the ability to understand questions is way ahead of the ability to provide answers. In many circumstances, a question calls for an inference, summary, or synthesis from the texts, rather than information specifically contained in the texts. This requires both the ability to generalize from what has been read, and the capacity to generate intelligible summaries. While we have built programs that perform some of these functions, the programs are still not fully integrated into the SCISOR system.

The second major area for improvements to SCISOR is in the definition of more domain knowledge. The ability of the system to extract information is limited to those scenarios that it understands. This knowledge acquisition problem can be addressed only by giving SCISOR more and more structured knowledge. Ultimately, we hope this will be achieved automatically from reading the texts, but for now, the knowledge is hand coded.

Until we have made substantial progress in these two areas, the efficacy of using natural language in a general information retrieval environment will be difficult to prove. We have, however, made a significant start on the general problem, and have established one context in which natural language question answering and conceptual information retrieval can be effectively combined. The capabilities currently provided by SCISOR would be a useful augmentation to an information retrieval system, even if more traditional methods were used as a general strategy. For example, the question answering, summarization, and automatic alert capabilities could supplement keyword retrieval. This will be a logical combination before NL techniques mature.

5 Summary and Conclusion

Using natural language processing techniques for information retrieval presents a number of major technical challenges that are not easily met by the current state of the art of artificial intelligence technology. General purpose text processing is the most difficult problem; and, without text processing, the use of natural language as an interface tool does not achieve its full potential.

The choice of application domain, knowledge acquisition strategy, and technical approach are the keys to the design of an information retrieval system that overcomes some of these obstacles. SCISOR is a natural language question answering system that retrieves information from news stories about corporate takeovers. These stories provide a well-motivated application area due to the fact that they deal with ongoing events and are well constrained, making document retrieval of limited usefulness and providing hope for natural language approaches.

The main technical strategy in SCISOR is to use the same language analyzer for both input questions and input text, but to rely heavily on knowledge of the domain to handle the text. Once information has been translated to a conceptual form, this form is used to provide intelligent information retrieval and summarization. This approach, along with the constraints of the domain and a sizable lexicon, allows for the practical use of natural language in an intelligent information retrieval context.

The current benefits of the natural language techniques applied in SCISOR are cooperative system responses, including summaries and alerts. The future challenges for the system are the development of substantial lexicons and knowledge bases, and the combination of natural language and IR techniques in progressively broader domains.

References

[BALLARD84] B. Ballard and J. Lusth. The design of DOMINO: a knowledge-based retrieval module for transportable natural language access to personal databases. In *Proceedings of the Workshop on Expert Database Systems*, Kiawah Isaland, South Carolina, 1984.

[CHARNIAK83] E. Charniak. Passing markers: a theory of contextual influence in language comprehension. *Cognitive Science*, 7(3):171–190, 1983.

[COHEN87] Paul R. Cohen and Rick Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Information Processing and Management, Special Issue on Artificial Intelligence for Information Retrieval*, 23(4):255–268, 1987.

[CROFT87] W. B. Croft and D. D. Lewis. An approach to natural language processing for document retrieval. In *Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 26–32, New Orleans, 1987.

[DEJONG79a] Gerald DeJong. Prediction and substantiation: a new approach to natural language processing. *Cognitive Science*, 3(3):251–273, 1979.

[DEJONG79b] Gerald DeJong. *Skimming Stories in Real Time: An Experiment in Integrated Understanding*. Research Report 158, Department of Computer Science, Yale University, 1979.

[DILLON83] Marin Dillon and Ann Gray. Fasit: a fully automatic syntactically based indexing system. *Journal of the American Society for Information Science*, 34(2):99–108, 1983.

[FAGAN87] Joel Fagan. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods*. PhD thesis, Cornell University, September 1987. Computer Science Department Technical Report 87-868.

[GINSPARG83] J. Ginsparg. A robust portable natural language data base interface. In *Proceedings of the Conference on Applied Natural Language Processing*, Santa Monica, CA, 1983.

[GROSZ85] B. Grosz, D. Appelt, P. Martin, and F. Pereira. *TEAM: An Experiment in the Design of Transportable Natural Language Interfaces*. Technical Report 356, SRI International, 1985.

[HARRIS84] L. Harris. Experience with INTELLECT: artificial intelligence technology transfer. *AI Magazine*, 5(2), 1984.

[JACOBS86] Paul S. Jacobs. Language analysis in not-so-limited domains. In *Proceedings of the Fall Joint Computer Conference*, Dallas, Texas, 1986.

[JACOBS87] Paul S. Jacobs. Knowledge-intensive natural language generation. *Artificial Intelligence*, 33(3):325–378, November 1987.

[JONAK84] Zdenek Jonak. Automatic indexing of full texts. *Information Processing and Management*, 20(5-6):619–627, 1984.

[JONES84] Karen Sparck Jones and J. I. Tait. Automatic search term variant generation. *Journal of Documentation*, 40(1):50–66, March 1984.

[KITTREDGE82] R. Kittredge and J. Lehrberger. *Sublanguages: Studies of Language in Restricted Domains.* Walter DeGruyter, New York, 1982.

[LEWIS88] David Lewis. *Comments on Joel Fagan's 1987 Dissertation.* Technical Report IIR Memo 1, University of Massachusetts at Amherst, 1988.

[LYTINEN86] Steven Lytinen and Anatole Gershman. ATRANS: automatic processing of money transfer messages. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, 1986.

[MCKEOWN82] K. McKeown. *Generating natural language text in response to questions about database structure.* PhD thesis, University of Pennsylvania, 1982.

[RAU87a] Lisa F. Rau. Information retrieval in never-ending stories. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 317–321, Los Altos, CA, Morgan Kaufmann Inc., Seattle, Washington, July 1987.

[RAU87b] Lisa F. Rau. Knowledge organization and access in a conceptual information system. *Information Processing and Management, Special Issue on Artificial Intelligence for Information Retrieval*, 23(4):269–283, 1987.

[RAU88] Lisa F. Rau and Paul S. Jacobs. Integrating top-down and bottom-up strategies in a text processing system. In *Proceedings of Second Conference on Applied Natural Language Processing*, pages 129–135, Austin, TX, Association for Computational Linguistics, Morristown, NJ, February 1988.

[SAGER78] Naomi Sager. Natural language information formatting: the automatic conversion of texts to a structured data base. In M. C. Yovits, editor, *Advances in Computers: 17*, Academic Press, New York, 1978.

[SALTON86] G. Salton. Another look at automatic text-retrieval systems. *Communications of the Association for Computing Machinery*, 29(7):648–656, 1986.

[SCHANK77] Roger C. Schank and Robert P. Abelson. *Scripts, Plans, Goals, and Understanding.* Lawrence Erlbaum, Halsted, NJ, 1977.

[SONDHEIMER86] N Sondheimer. *Proceedings of DARPA's 1986 Strategic Computing Natural Language Processing Workshop.* Technical Report ISI/SR-86-172, University of Southern California, ISI, 1986.

[WILENSKY86] R. Wilensky. Knowledge representation - a critique and a proposal. In Janet Kolodner and Chris Riesbeck, editors, *Experience, Memory, and Reasoning*, pages 15–28, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1986.

[YOUNG85] S. Young and P. Hayes. Automatic classification and summarization of banking telexes. In *The Second Conference on Artificial Intelligence Applications*, pages 402–208, IEEE Press, 1985.