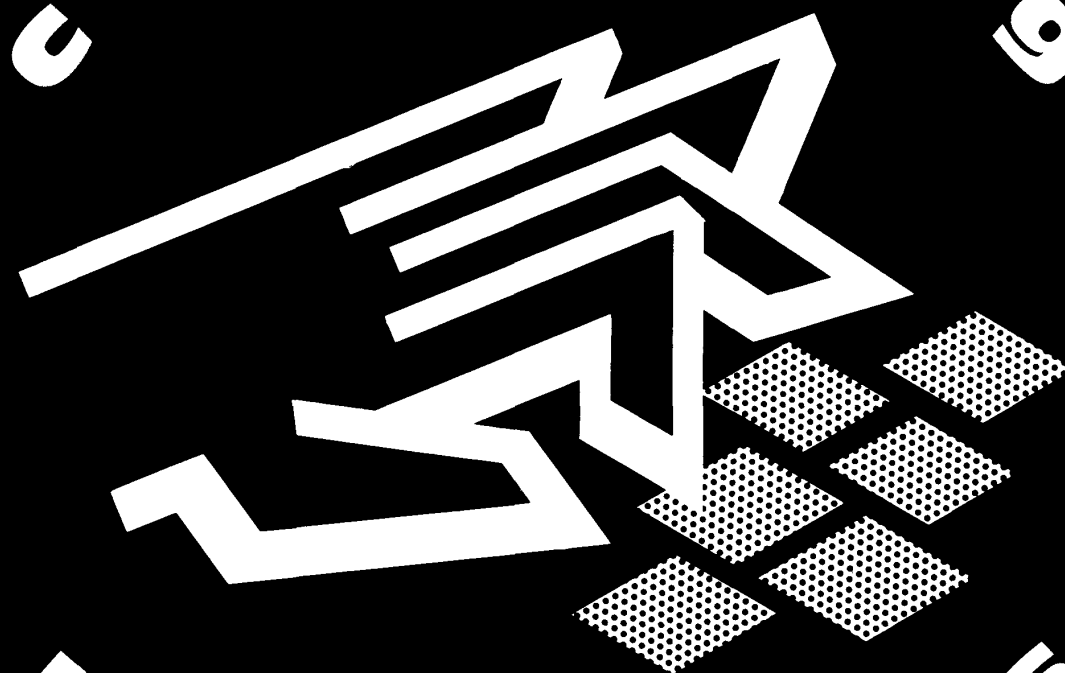




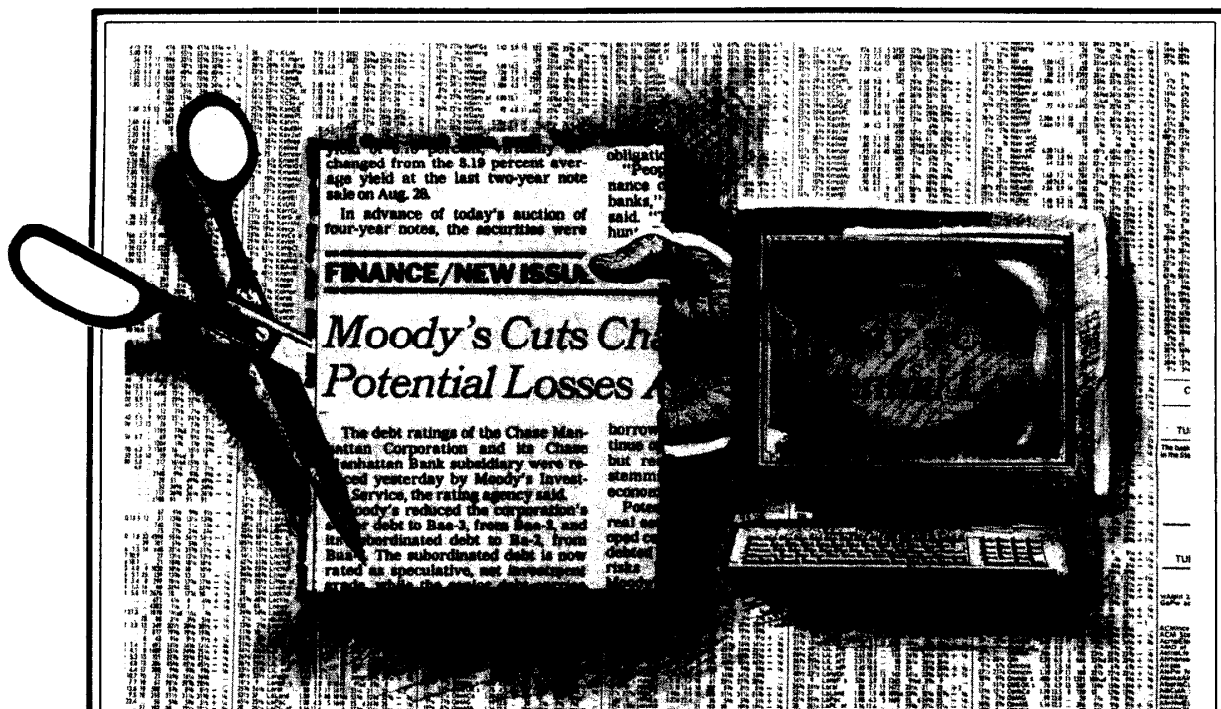
computing



practices

SCISOR: Extracting Information from On-line News

An Object-Oriented Relational Database



SCISOR: Extracting Information from On-line News

Paul S. Jacobs and Lisa F. Rau

Edgar H. Sibley, Panel Editor

The future of natural language text processing is examined in the SCISOR prototype. Drawing on artificial intelligence techniques, and applying them to financial news items, this powerful tool illustrates some of the future benefits of natural language analysis through a combination of bottom-up and top-down processing.

New technologies, such as high-speed networks, inexpensive massive storage, and optical-character readers, have combined to produce a sharp increase in the availability of on-line text. The overflow of information demands advanced techniques for organizing and accessing texts, including the automatic extraction of selected information from on-line sources [9].

Taking advantage of text information in many applications requires more than what text retrieval systems can offer, including higher accuracy in constrained domains as well as a broader range of information retrieval capabilities. Most text retrieval methods use potentially

ILLUSTRATION: DENNIS AHLGREN

complex boolean queries with systems that perform word-based statistical matching and produce inadequate, albeit state-of-the-art, results [1]. Although automatic full-text retrieval systems are more accurate than manually indexed retrieval systems [17], the typical precision rate one can expect is 75 percent with a recall rate of 20 percent. While the widespread commercial applications of these methods demonstrates their usefulness, they are inaccurate and unsuitable for many tasks. The systems often require users to read irrelevant texts as well as relevant ones and cannot perform functions that demand an analysis of the *content* of the documents, such as text summarization and the automatic generation of databases from texts.

The System for Conceptual Information Summarization, Organization, and Retrieval (SCISOR) is a prototype system that performs text analysis and question answering in constrained domains. Developed over the last four years, SCISOR operates on financial news, selecting and analyzing stories about corporate mergers and acquisitions from an on-line financial service (Dow Jones™). It runs in real time on a Sun™ workstation in Common Lisp (Lucid) under UNIX™. It has also been ported to extract information from other types of messages in various subject areas.

SCISOR processes news at the rate of about six stories per minute. It performs the following tasks:

- The lexical analysis of the input character stream, including names, dates, numbers, and contractions.
- The separation of the raw news feed into story structures, with separate headline, byline, and dateline designations.
- A topic determination for each story, indicating whether it is about a corporate merger, acquisition, or other topic;
- The natural language analysis of each selected story using an in-

tegration of two interpretation strategies—"bottom-up" linguistic analysis and "top-down" conceptual interpretation.

- The storage and retrieval of conceptual representations of the processed texts into and out of a knowledge base.

The design of SCISOR combines artificial intelligence (AI) methods, especially natural language processing, knowledge representation, and information retrieval techniques, with more robust but superficial methods, such as lexical analysis and word-based text search. This approach provides the broad functionality of AI systems without sacrificing robustness or processing speed. In fact, the system has a throughput for real text greater than any other text extraction system we have seen (e.g., [18, 19]), while providing knowledge-based capabilities such as producing answers to English questions and identifying key conceptual roles in the text (such as the suitor, target, and per-share price of a merger offer). SCISOR consists of roughly 50,000 lines of Common Lisp code. It was developed entirely on Sun workstations.

The rest of this article will present a brief overview of SCISOR and samples of SCISOR in operation, with the reasons behind many of the design choices in the system. The discussion will give details on each of the major components of the system, followed by the results achieved in experimenting with SCISOR and an analysis of the methods available for evaluating these results.

System Design

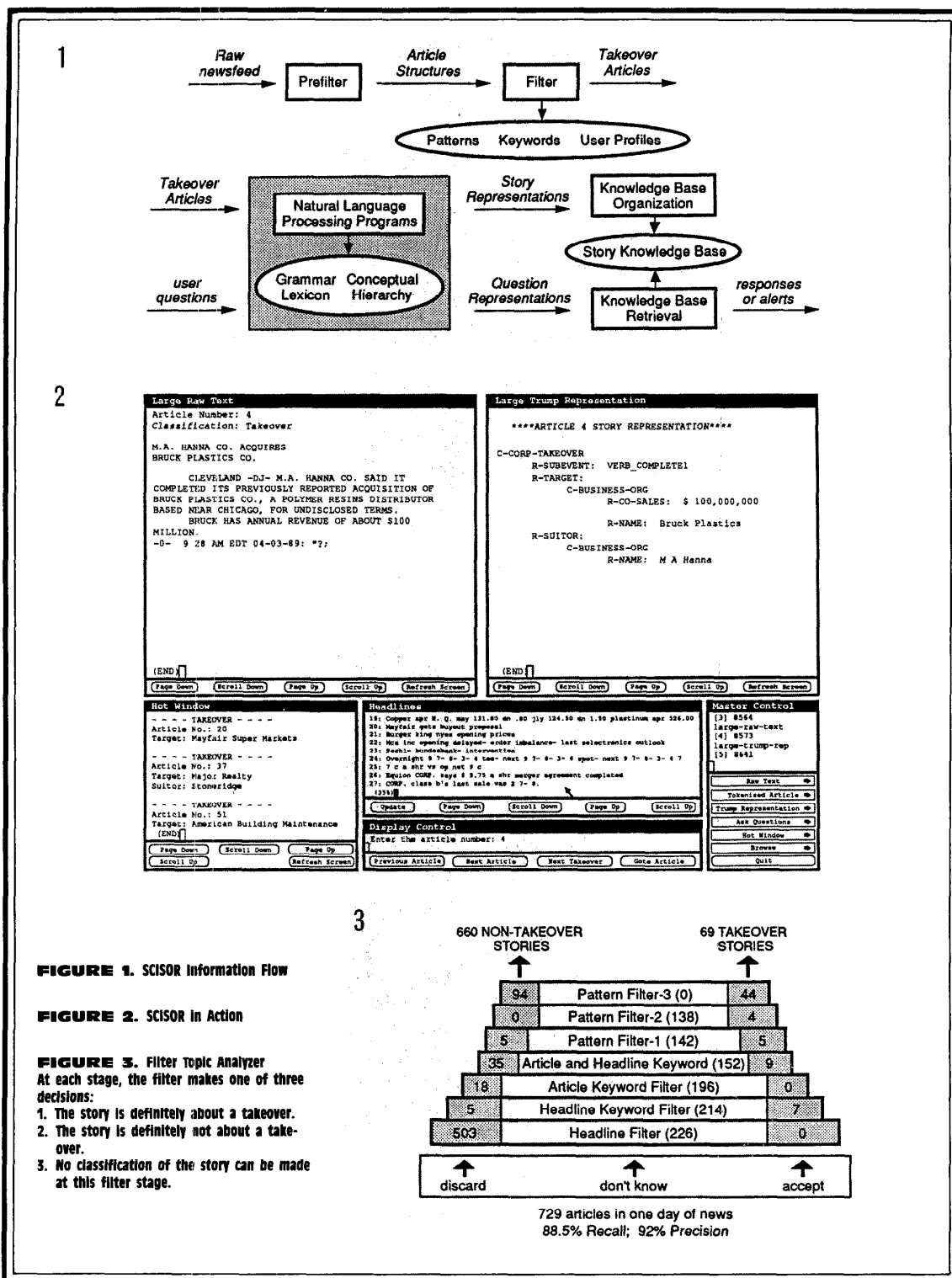
SCISOR's design provides each system component with access to a rich, hand-coded knowledge base, but each component applies the knowledge selectively, avoiding the computation that a complete analysis of each text would require. The architecture of the system allows for *levels* of language analysis, from rough skimming to in-depth con-

ceptual interpretation.

Figure 1 shows the information flow in SCISOR. Input texts feed in from the news service. The pre-filter program distinguishes headlines and other structured information from the stories. The filter component (or *topic analyzer*) selects stories about mergers and acquisitions and performs the lexical analysis of names, dates, numbers, and other special inputs. The natural language components, using a combination of language-driven (bottom-up) and conceptual (top-down) analysis, process the takeover texts, identifying key roles such as target, suitor, and price, as well as other features like financing, company products, or legal complications. The result of this analysis is a single representation of each story that the program adds to a central knowledge base. The conceptual retrieval component accesses information in this knowledge base by analyzing English questions in the same manner and matching the questions to the story representations stored in the knowledge base. Later we will give more details on each of the processes.

SCISOR provides the user with information in multiple forms. Users can browse the headlines and the original texts. A "hot window" continuously displays the target, suitor, and price of the latest takeover stores, and it flashes when a new takeover story comes across the newswire. For more general information needs, an "ask question" window allows the user to type in simple English questions (e.g., "What was offered for Polaroid?") as well as query fragments (e.g., "acquisitions by Shamrock").

Figure 2 shows a Sun workstation screen during the operation of SCISOR. The "Master Control" window in the lower right allows the user to open or access the various features of the system. The "Headlines" and "Display Control" in the lower center show the headlines of all stories (with headlines of takeover stories in bold) and guide





the selection of texts for browsing. The "Hot Window," or alert feature, is at the lower left, alerting users at the moment a new, potentially relevant article comes across the newswire. The "Raw Text" and "Trump Representation" windows at the top display each selected story, showing key portions of text in boldface with a summary of the language analysis in the upper right.

The windows all currently use Sun and Unix utilities such as tool-tool [14], less, and shell scripts. The Lisp process that performs the text and query analysis is hidden, often running on a different machine. This design makes the natural language part of the system a "server," processing the news and queries while users browse the results. As a result, the system could be extended for many users to select, highlight, and retrieve different information for each user according to that user's prescribed interests.

The main bottleneck with this approach presently is that the Lisp "server" process must be interrupted when any user asks an English question (since there is only one natural language program at work). Each question takes a few seconds to answer, so frequent interruptions might cause the text processing to lag slightly behind the news feed. These delays, however, are quickly recovered, as the natural language program is capable of performing its analysis of texts faster than news comes across the newswire.

This section has covered the overall design and *look and feel* of SCISOR. The discussion that follows will give the technical highlights of each of the major components of the system.

The Topic Analyzer

The topic analyzer, or filter, breaks up the steady stream of stories and determines whether a given story is about a merger or an acquisition. The filter applies a series of screening processes, increasing in the

degree of computational complexity and linguistic analysis, to weed out stores not related to the topic. The function of this component (to decide to include or exclude a document from the set of selected documents) is similar to traditional information retrieval; however, the design is unusual in three important ways.

First, it was built with a modular architecture to support its primary function as an experimental research vehicle. This design supports the plugging in of new algorithms to filter the stories at any stage in the filtering process. Programs automatically compare the results against previous system configurations. Second, some general knowledge about linguistic structure (such as the relationship between verbs and objects) augments word-based statistics. Finally, the analyzer relies on negative as well as positive information, eliminating articles that are *not* about takeovers in order to increase precision while not missing any takeover stories.

By the final stages of the filter, about 10 percent of the total incoming stories relate to mergers and acquisitions, with slightly over 90 percent (combined recall and precision) accuracy. When a story appears in the "hot window," it not only must relate to an acquisition but must identify either a target or suitor (thus excluding divestitures, strategic holdings, or corporate buybacks of stock).

Figure 3 illustrates one successful combination of filtering processes applied by the topic analysis program in its analysis of the newswire. The *correct* classifications in all of our experiments are those made by one person. We estimate about a 10 percent margin of error caused by differences in classification of the stories' subjects by various people. The numbers that appear after the labels in the layers indicate the number of stories in a randomly selected day of news stories that passed through each stage of the filter.

The filter is too complex a pro-

gram to describe fully here; more details can be found in [15]. The discussion that follows provides some basic information on the primary methods of filtering stories.

The first processing stage the filter uses breaks the news feed into structures consisting of a headline, optional byline, story, and dateline. Some headlines indicate recurrent update stories that can be immediately discarded, such as "DOW JONES STOCK AVERAGES" or "NYSE MOST ACTIVES." Stories that begin with these preset headlines are known with 100 percent certainty not to be about the topic of takeovers.

After this headline filtering, the program applies keywords first to just the headlines of the articles, then to just the articles, and then to both the headlines and the articles together. Keyword filtering looks for the presence of certain prespecified keywords, such as *buy*, *merger*, and *acquisition*, to signal that a story could be about a takeover. There are currently about 150 keywords used during this stage. Each keyword has a weight associated with it, and each article has a threshold determined by the length of the article and the total number of keywords in the article. These numbers are determined automatically after a user selects sample stories about the subject.

After keyword analysis, a pattern-matching process scans only those stories that pass through these first stages, applying both positive patterns (indicating the story *is* about the subject), and negative patterns (indicating that the story is *not* about the topic). For example, a story included on the basis of a positive keyword such as *buy* may be discarded because of a negative pattern such as *buy side* or *buy &&& debt obligations*, if there is no other positive evidence to outweigh this. In this example, the &&& indicates where the pattern will match some small number of intervening words. For example, a sentence that contained the words *BUY FOREIGN DEBT OBLIGA-*

TION would match this pattern. The total score for a story is the sum of the scores for positive (confirming) patterns minus the sum of the scores for negative (refuting) patterns.

The most computation-intensive stage of the filter takes as input those stories that the previous stages could not definitely categorize and subjects those stories to linguistic and conceptual analysis. This process, for example, can use knowledge that the conceptual *object* of a takeover is a *company* in order to determine that a story about the sale of the Sears Tower is not a takeover story.

Bottom-up and Top-down Analysis: TRUMP and TRUMPET

SCISOR's text-processing algorithm is one of two features that sets it apart from other natural lan-

guage programs (the knowledge base design, described later in this article, is the other). Text processing in SCISOR combines two analysis strategies—*bottom-up*, language-driven interpretation, and *top-down*, expectation-driven processing. Bottom-up analysis starts with a parse of each sentence, identifying linguistic structures and mapping these linguistic structures into a conceptual framework. Top-down analysis starts with conceptual expectations, such as the knowledge that takeovers involve two companies, and tries to fill these expectations given partial information from the text.

Each of the two processing strategies has strengths and weaknesses. Bottom-up analysis [4–6] uses knowledge about language and can produce accurate, if only partial results, in arbitrary texts and texts that contain unexpected informa-

tion. Top-down analysis [3, 13] is much more tolerant of unknown words and grammatical lapses, but is often *fooled* or misses information in unusual situations. For example, top-down analysis may have trouble understanding a story involving the so-called "Pac Man" defense where a smaller company, in response to an unwanted takeover offer, turns the tables and attempts to take over the hostile suitor. The top-down strategy might cause a system to assume that a company must be either a target or suitor in a given takeover story (but not both) and rely on that assumption to fill in these roles.

Processing in the SCISOR domain seems to require the combination of the two strategies. The language of financial news is rich enough to require a depth of linguistic knowledge in order to accurately identify targets and suitors,

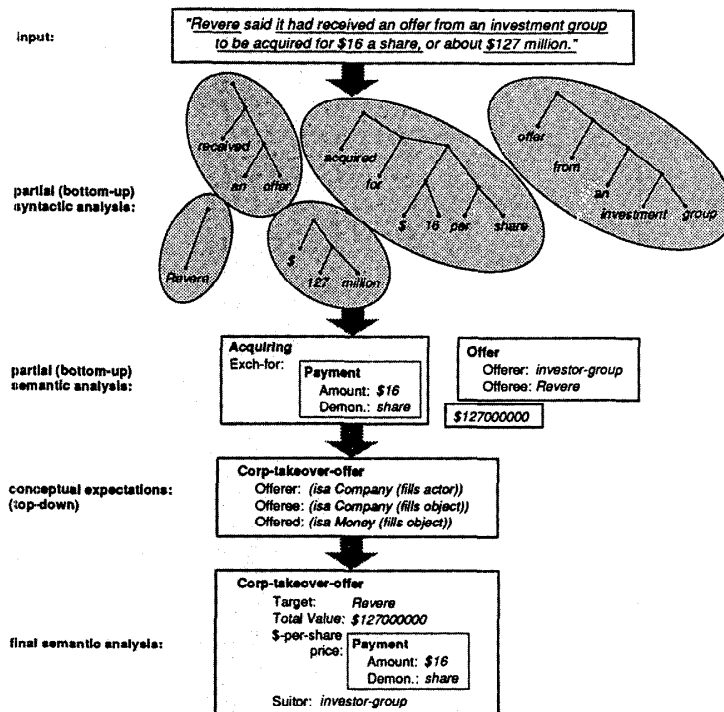


FIGURE 4. Integrating Bottom-up and Top-down Analysis

but it is also constrained enough that top-down analysis can aid the interpretation task. For example, the sentence "Acme initiated an offer" places Acme in a different role from "Acme rejected an offer." Only knowledge about language, including grammatical and lexical information, can accurately determine whether Acme is the target or suitor in this sort of example. Because of the specifics of corporate takeovers, however, (e.g., that there are typically two companies involved, one a target and one a suitor), it is reasonable to assume that some company mentioned in a merger story is either a target or a suitor. This top-down information often helps to fill in a missing role where the bottom-up component is only able to fill one. For example, in the headline "Kimberly Clark Up, Rumor Perelman Making Bid Tonight" (Dow Jones, Oct. 5, 1988), the correct interpretation (with Kimberly Clark as the target) stems from the top-down assumption that it must be since it is not the suitor. Also lending credence to this assumption is the expectation that the target of the takeover is typically the same company whose stock rises.

The manner in which top-down and bottom-up analysis interact in SCISOR distinguishes the system from some other programs that are otherwise quite similar. For example, FRUMP [3], IPP [12], and POLITICS [2] were all systems that processed news stories and claimed to be integrated and extensible. SCISOR gives considerably more weight to linguistic knowledge than any of these systems and can even produce a fair amount of syntactic and semantic information from a text in a completely new domain. Another difference is in the degree to which the systems have actually been extended; handling a volume of hundreds of texts per hour in multiple domains requires more robustness than these earlier systems exhibited, although they were all significant in their time.

The bottom-up analyzer of

SCISOR, TRAnsportable Understanding Mechanism Package (TRUMP) [7] is a general-purpose parser and semantic interpreter that applies knowledge about language to the task of producing a conceptual interpretation. TRUMP derives a frame-like semantic representation of each phrase or sentence, taking into account word meanings and surface structures but not domain-specific knowledge. TRUMP Expectation Tool (TRUMPET) performs the top-down component of analysis, matching TRUMP's conceptual structures with expectations. For example, TRUMP will analyze the input "Ace received an offer from Acme" to identify the *offerer* and *offeree* while TRUMPET relates those roles to the *target* and *suitor* of an acquisition. More detail on the integration of these two language-processing mechanisms is found in [16].

Figure 4 shows an example of the combination of language analysis strategies. The input sentence is "Revere said it had received an offer from an investment group to be acquired for \$16 a share, or about \$127 million." TRUMP understands all the words in the story, but through bottom-up analysis alone it cannot draw some of the required conclusions. For example, the phrase starting with "to be acquired" might attach to "an investment group" or "an offer," but in this case "Revere" is the subject of the phrase. TRUMP learns from TRUMPET that the *offerer* must be the same as the *acquirer* and that the *acquirer* must be different from the *acquiree*, so "Revere" is the *acquiree* and therefore the *target* of the takeover. TRUMPET also distinguishes the per share value (\$-per-share-price) from the total value of the acquisition.

As TRUMP scans the input text, it looks ahead for words that link to concepts associated with expectations in TRUMPET's knowledge base and then uses syntactic knowledge to draw boundaries in the text where information from the text

modifies key concepts. Determiners (e.g., "the" and "a"), coordinating conjunctions (e.g., "and" and "or"), and certain punctuation (e.g., colon, semicolon, or parenthesis) help to determine these boundaries by definitively marking the beginning or end of a phrase. This preliminary segmentation of the text serves three purposes.

- (1) It makes the parser faster by allowing it to skip sections of the text that are irrelevant or too complex.
- (2) It determines roughly where the bottom-up component should prevail over expectations, that is, where there are strong linguistic preferences.
- (3) It highlights in boldface type the crucial segments of the raw text input for display to the user.

Language analysis performs several functions in SCISOR: it is the final stage of the filtering process. If the combination of analysis strategies cannot accurately determine a target or suitor, this is usually evidence that the text is not about a takeover. The analyzer also helps in the text-browsing facility by highlighting relevant portions of text. Then it directs key features of the text into the "hot window." Finally, the synthesized results of the language analysis, including the role that each company plays in a takeover, become part of SCISOR's conceptual knowledge base.

Story Storage and Retrieval

After TRUMP and TRUMPET have completed an analysis of a story, SCISOR stores the conceptual representation of the story as a network of unique instances in long-term memory. The instances are individual members of conceptual categories, such as companies, offers, and mergers, and these serve as indices for information retrieval. Figure 5 shows a representation of a portion of a takeover story involving the companies

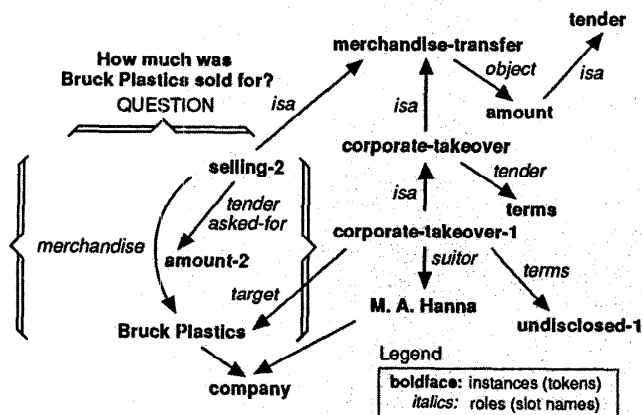


FIGURE 5. Answer Retrieval

Bruck Plastics and M. A. Hanna, with the constituent instances in boldface. The analysis of a user's questions by TRUMP and TRUMPET produces a representation in a similar form, also shown in the figure.

Retrieval of an answer to an input question uses a conceptual graph-matching algorithm that ranks and compares the representation of the question to representations of stories. This process avoids excessive comparisons by using a two-step method of conceptual retrieval. The first step is a rough, efficient comparison of features of the question with features of stored representations of texts; the second pass is a more careful match of relationships that are asked for or implicit in the question. Figure 5, for example, shows the retrieval process in response to the question: "How much was Bruck Plastics sold for?" This question relates to the answer "Terms were not disclosed" through the MERCHANDISE-TRANSFER category and the inheritance and refinement of the TENDER role of MERCHANDISE-TRANSFER to the TERMS role of a CORPORATE-TAKEOVER. The names of the companies are the primary contributors to the retrieval of this network; however, any instance can contribute to the retrieval of a can-

didate story.

After the system retrieves a set of candidate stories, the graph matcher compares the question and story representations and computes a score for each comparison. The semantics of the nodes and links in these representations guide the determination of how the question matches parts of the story and helps to determine the score for each match. For example, in Figure 5, the match between the *merchandise* role in the question and the *target* role of the story gets a high score. This matching process selects both the best story (for reference to the user) and the answer to the question, where applicable.

Other Features of SCISOR

Many of the appealing features of SCISOR stem from its central knowledge base, which the system components use to perform different levels of analysis. For example, the central system lexicon includes knowledge about words and word meanings that applies to topic analysis, bottom-up and top-down text processing, and response generation. This knowledge-based design has helped the flexibility and transportability of the system.

Core Lexicon and Linguistic Knowledge

SCISOR uses a general lexicon of

about 10,000 word roots, with about 75 affixes (prefixes and suffixes) that can derive variations from those roots. Each lexical entry includes the linguistic properties of a word root and a set of senses, which link to a core concept hierarchy of 1,000 general conceptual categories from *state-change* (including many events and verb senses) to *animate-internal-part* (including microbes, organs, and other anatomical entities). This organization helps to apply the word knowledge in the lexicon to new domains, either by adding new senses or refining existing lexical knowledge. In the domain of mergers and acquisitions, the core lexicon covers over 95 percent of word occurrences (exclusive of proper names). The M&A lexicon includes only 12 words outside of the core lexicon (such as *debenture* and *CEO*), although many words have preferred (domain-specific) senses. The domain knowledge also includes 23 lexical combinations (such as *seek control* and *make a bid*) and 32 specific concepts.

Text processing requires substantial lexical analysis beyond individual words, such as handling proper names, special text structures and symbols, tables, abbreviations, numbers, and codes. SCISOR uses a lex-style finite-state based lexical analyzer for translating these inputs into a stream of tokens, which often contribute to TRUMP's interpretation of a text. In addition, the program uses a special multi-token analyzer for finding previously unknown company names in an input, that correctly identifies most new company names using special words such as "Partners," "Inc.," and "Ltd." Once the full name of a company has been identified, another special program generates potential variations of the company name such as its acronym or other abbreviated references. The new company name and its potential variations are then added to the company name database.

The TRUMP analyzer shares the entire linguistic knowledge base,



including the lexicon and grammar, with a language generator called Knowledge INTensive Generator (KING) [8], which can produce English responses to questions. In addition to being more efficient than having separate knowledge bases, this sharing of linguistic knowledge assures that the different components of the system "speak" the same language; e.g., a specific phrase like "tender offer" in the vocabulary of the financial domain is available to the analysis and generation components. In the current system, responses to questions are quite simple; however, as the program expands, KING will become more important in constructing summaries of complex events.

Portability to New Domains

SCISOR has been designed for

transportability, from the lowest level of the design of its knowledge base [10] to acquisition algorithms [11, 21] that can help to customize the system's lexicon and determine some of the meanings of new words in context. The system developers and several other locations in General Electric have applied most of the system components to other domains. Substantial prototypes (e.g., extracting thousands of conceptual structures or database fields with high accuracy from hundreds of texts) in these other application areas have required no more than a few person-months of effort.

Shortly after the completion of SCISOR, the program was ported from the mergers and acquisitions domain to the domain of military OPREP-3 messages as part of a government-sponsored effort to evalu-

ate natural language systems called MUCK-II [19]. The task in this domain involved the extraction of ten features from over 200 messages, and we have collected results from porting SCISOR to this new task.

We estimate that it took approximately 40 person-days (1.3 person-months), to port the entire system to OPREP-3 messages. The time includes knowledge engineering, system development, scoring of results, documentation of the effort, and modification of the demonstration environment. System enhancements included the addition of a spelling checker and telegraphic parsing mode, broader recovery strategies for ill-formed input, a context-switching mechanism, a concretion (or simple inference) mechanism, and a few domain-

```

shelltool - /bin/csh
marfak> deuce

****Welcome to the DEUCE directory system****

DEUCE knows about CRD organizations and personnel.
Wait six seconds for the prompt, then ask your questions in English.
End with <Control-D>.
[A transcript of this session is being recorded for debugging purposes.]

DEUCE> Who's the manager of the artificial intelligence program?
the manager of the ARTIFICIAL INTELLIGENCE PROGRAM is NORMAN K SONDHEIMER

DEUCE> Where's it located?
the ARTIFICIAL INTELLIGENCE PROGRAM is located in building K-1 room 5C17

DEUCE> The manager of superconductivity?
the manager of the APPLIED SUPERCONDUCTIVITY PROGRAM is EVANGELOS T LASKARIS
the manager of the SUPERCONDUCTIVITY APPLICATIONS PROGRAM is RICHARD L RHODENIZER

DEUCE> Wood's mgr's Tel#
With Respect To: DONALD L WOOD
LADDIE L STAHL's telephone number is 6454

With Respect To: NANCY C WOOD
NORMAN K SONDHEIMER's telephone number is 5362

With Respect To: RALPH T WOOD
enter your name or <CR>: paul jacobs
PAUL S JACOBS's telephone number is 5314

DEUCE> Who's my manager?
the manager of PAUL S JACOBS is NORMAN K SONDHEIMER

DEUCE>

---Emacs: *shell*          18:38am 8.52[8]      (Shell: run)-----All-----

```

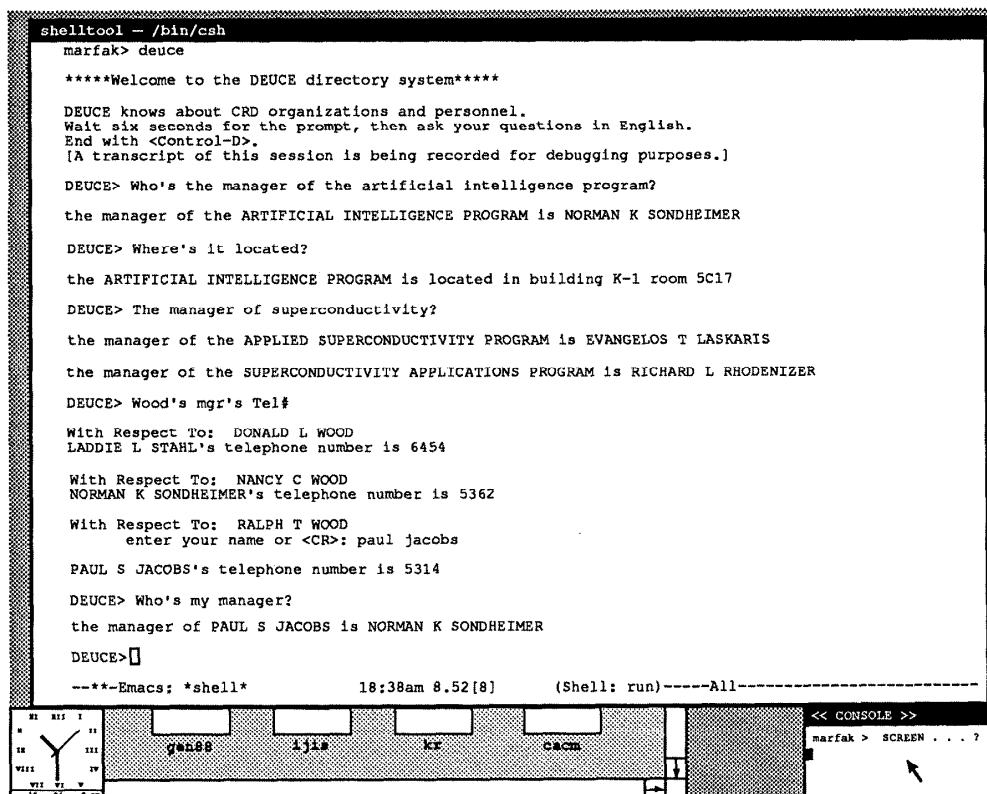


FIGURE 6. A Different Application of TRUMP

specific heuristics. Our success in porting the entire system to a very different application area with a minimum of effort makes us confident of the system's general portability.

In addition to text-processing applications, TRUMP also serves as the natural language front end to an organizational directory, called DEUCE (Directory Extraction Using Common English). Figure 6 shows some of the capabilities of this system, including handling pronoun references, abbreviations, contractions, and database joins. This application required a few months of customization, mainly devoted to integrating the database package with the natural language program.

In general, our preliminary estimates show that each new application requires about a 5 percent margin of labor from the original system. In other words, if an application uses tools that result from three person-years of effort, about two person-months suffice for a prototype in that application.

Performance Evaluation

It would be difficult and probably futile to perform a controlled study of SCISOR against a traditional IR system, for two reasons.

- (1) Traditional IR systems are tested on arbitrary, unconstrained texts, while natural language systems still work only in constrained domains.
- (2) SCISOR performs many tasks other than document retrieval, such as extracting information from stories and directly answering users' questions.

Only the filter portion of the program has a set of documents as its output. Comparing the filter with a traditional IR system would be possible if the input to the IR system were a broad topic classification, such as "takeover stories" or "stories about stock buybacks." We have performed experiments to determine the recall and precision of the

filter under one topic area (see the Topic Analyzer section) but have not compared these numbers with a search of the same document set using other IR systems.

Evaluation problems of the entire system stem from the unique functionality of the SCISOR system. Document retrieval systems, even sophisticated ones like RUBRIC [20], do not extract features from the documents they retrieve; thus it is impossible to compare them to SCISOR. We have performed some tests, however, that do measure SCISOR's accuracy.

The government-sponsored MUCK-II [19] evaluation is, to our knowledge, the most meaningful test of natural language text processing, but the participants in the MUCK-II evaluation were not at liberty to release the specific results of the experiment. We will try, however, to summarize the status of performance evaluation in general terms. Evaluation of content-based, text-processing systems such as SCISOR is not nearly as established as evaluation methods in information retrieval. There are many tasks to be tested in this emerging type of system, including accuracy of question answering, helpfulness of alerts, and coverage of structured information (such as target and suitor). No mature methods exist for testing any of these tasks.

Aside from text categorization (as done by the filter component of SCISOR, which does somewhat resemble traditional IR), the easiest result to evaluate is the accuracy with which a system produces "fixed field" or "fixed format" information from free text. These results, however, depend on the difficulty of the texts, the nature of the fixed field information, and the method for scoring errors. Producing the results over a large set of texts is time consuming and may test the accuracy of the human solution keys as much as the system output.

¹The FRUMP [3] program, for comparison purposes, achieved 38 percent accuracy in one test of unseen newswire stories.

In spite of the problems with evaluating this sort of system, we would like to be informative about how our program performs. As a rule, it can extract key features from large sets of constrained texts with 80–90 percent (combined recall and precision) accuracy. It can achieve better results (and has) with more constrained texts, but would also produce almost nothing useful, for instance, in reading the entire *Wall Street Journal*. It is realistic to expect 90 percent accuracy for certain useful, carefully constructed tasks, and unrealistic to expect much higher than this.¹ Many difficulties in reading texts appear when trying to achieve better results, but the most common limitation seems to be the degree of real inference required for understanding. In spite of its fairly sophisticated methods for combining linguistic and world knowledge, SCISOR really has very little of the latter.

In a recent test of SCISOR, the program analyzed one day's worth of stories directly from the newswire source. Of the 729 stories, the filter achieved slightly over 90 percent of averaged recall and precision in its determination of which stories were about mergers and acquisitions (69 in all). SCISOR correctly identified the target and suitor in 90 percent of all the stories. When dollar-per-share amounts of offers were present in the stories, SCISOR extracted this quantity correctly 79 percent of the time and the total value of the offer 82 percent of the time.

Summary and Conclusion

SCISOR represents a prototype implementation of one vision of the future of natural language text processing, setting artificial intelligence techniques in the context of an information retrieval problem. The program accurately processes financial news, selects items of interest, provides a user interface that allows easy access to results, and extracts important information in a structured form. This is all done



with sufficient flexibility to port to new domains and applications. As the proliferation of on-line text continues, we see this sort of text-based conceptual information system at the heart of intelligent systems technology.

Acknowledgment

Codevelopers of and contributors to SCISOR include and have included Richard Povinelli, Nancy Irwin, Barbara Kipfer, and George Krupka. The authors would like to give special thanks to our past and present managers, Drs. Larry Sweet and Norm Sondheimer, for their encouragement and continuing support of this project. Three anonymous reviewers provided valuable comments and feedback on this article. **G**

References

- Blair, D.C., and Maron, M.E. An evaluation of retrieval effectiveness for a full-text document retrieval system. *Commun. ACM* 28, 3(Mar. 1985), 289-299.
- Carbonell, J. Towards a self-extending parser. In *Proceedings of the 17th Annual Meeting of the Association for Computational Linguistics* (La Jolla, Ca., Aug. 11-12). ACL, Menlo Park, Ca., 1979, pp. 3-7.
- DeJong, G. Prediction and substantiation: A new approach to natural language processing. *Cog. Sci.* 3, 3(Mar. 1979), 251-273.
- Grishman, R., and Hirschman, L. PROTEUS and PUNDIT: Research in text understanding. PROTEUS Proj. Memo. 1, NYU, New York, 1986.
- Grishman, R., and Kittredge, R., Eds. *Analyzing Language in Restricted Domains: Sublanguage Description and Processing*. Lawrence Erlbaum, Hillsdale, N.J., 1986.
- Hobbs, J.R. Site report: Overview of the TACITUS project. *Comput. Ling.* 12, 3(Mar. 1986), 220-222.
- Jacobs, P.S. Language analysis in not-so-limited domains. In *Proceedings of the ACM-IEEE Computer Society Fall Joint Computer Conference* (Dallas, Tx., Nov. 2-6). ACM and IEEE-CS, New York and Washington, D.C., pp. 247-259.
- Jacobs, P.S. Knowledge-intensive natural language generation. *Artif. Intel.* 33, 3(Nov. 1987), 325-378.
- Jacobs, P.S., Ed. *Proceedings of the AAAI Spring Symposium Series: Text-Based Intelligent Systems*. The Amer. Assoc. for Artificial Intelligence, (Palo Alto, Ca., 1990).
- Jacobs, P.S., and Rau, L.F. Ace: Associating language with meaning. In *Proceedings of the Sixth European Conference on Artificial Intelligence* (Pisa, Italy, Sept. 5-7). AICA/AISB, pp. 137-146.
- Jacobs, P.S., and Zernik, U. Acquiring lexical knowledge from text: A case study. In *Proceedings of the 7th National Conference on Artificial Intelligence* (St. Paul, Mn., 1988), pp. 739-744.
- Lebowitz, M. Memory-based parsing. *Artif. Intel.* 21, 4(Dec. 1983).
- Lytinen, S., and Gershman, A. ATRANS: Automatic processing of money transfer messages. In *Proceedings of the 5th National Conference on Artificial Intelligence* (Philadelphia, Pa., 1989-1093).
- Musciano, C. *Tooltool User's Guide*. Advanced Technology Dept., Harris Corp., 1989.
- Povinelli, R. Topic analysis: Filter. Tech. Rep., GE Corporate R&D, Schenectady, N.Y., 1989.
- Rau, L.F., and Jacobs, P.S. Integrating top-down and bottom-up strategies in a text-processing system. In *Proceedings of the 2nd Conference on Applied Natural Language Processing* (Morristown, N.J. ACL, Menlo Park, CA., 1988), pp. 129-135.
- Salton, G. Another look at automatic text-retrieval systems. *Commun. ACM* 29, 7(July 1986), 648-656.
- Sondheimer, N. Proceedings of DARPA's 1986 strategic computing natural language processing workshop. Tech. Rep. ISI/SR-86-172, Univ. Southern California, 1986.
- Sundheim, B. Second message understanding conference (MUCK-II) test report. Tech. Rep. 1328, Naval Oceans System Center, San Diego, Ca., 1990.
- Tong, R.M., et al. RUBRIC III: An object-oriented expert system for information retrieval. In *Proceedings of the 2nd Annual IEEE Symposium on Expert Systems in Government* (McLean, Va., Oct. 20-24). IEEE-CS, Washington, D.C., 1986, pp. 106-115.
- Zernik, U. Lexical acquisition: Learning from corpus by capitalizing on lexical categories. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence* (Detroit, Mi., 1989), pp. 1556-1562.

CR Categories and Subject Descriptors: H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing—*linguistic processing*; H.3.4 [Information Storage and Retrieval]: Systems and Software—*current awareness systems (selective dissemination of information-SDI); question-answering (fact retrieval) systems*; I.2.7 [Artificial Intelligence]: Natural Language Processing—*language parsing and understanding; text analysis*

Additional Keywords and Phrases: SCISOR, TRUMP, TRUMPET

About the Authors:

PAUL S. JACOBS is currently a computer scientist with the GE Research and Development Center in Schenectady, NY. His current research interests include investigation of natural language text skimming, lexicon development, and methods for broad-coverage semantic interpretation.

LISA F. RAU is currently a computer scientist at GE Research and Development Center. Her current research interests include advanced methods for memory organization and retrieval, integrated text processing, and conceptual information processing.

Authors' Present Address: Artificial Intelligence Program, GE Research and Development Center, Schenectady, N.Y. 12301. Email: psjacobs@crd.ge.com; rau@crd.ge.com

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

© 1990 ACM 0001-0782/90/1000-0088 \$1.50