Developments in Automatic Text Retrieval

GERARD SALTON

Recent developments in the storage, retrieval, and manipulation of large text files are described. The text analysis problem is examined, and modern approaches leading to the identification and retrieval of selected text items in response to search requests are discussed.

MUCH OF THE INFORMATION CIRCULATING IN THE MODERN world consists of written, natural language text. Often the texts are available in machine-readable form and can be stored and reproduced automatically and transmitted from place to place on electronic networks. Among the information items now routinely processed on computers are electronic messages, wire service stories and bulletins, newspaper articles, research papers and documents, textbook materials, instruction manuals, dictionary and encyclopedia articles, and published materials of many kinds.

It is easy to store large masses of information, but storage in itself is of no value unless systems are designed that make selected items available to interested users. In particular, the information content must be analyzed, and appropriate content identifiers must be generated and attached to the stored items; user needs must be identified and formulated in terms understandable by an automated system; and representations of document content and user needs must be compared, leading to the retrieval of items judged to be sufficiently close to the information requested.

Conventional Retrieval Methods

In conventional information retrieval, the stored records are normally identified by sets of keywords or phrases known as index terms, or simply terms. Requests for information are typically expressed by Boolean combinations of index terms, consisting of search terms interrelated by the Boolean operators *and*, *or*, and *not*. The retrieval system is then designed to select those stored items that are identified by the exact combination of search terms specified in the available queries. Thus, in a four-term query statement such as [(T_1 *and* T_2) *or* (T_3 *and* T_4)], each retrieved item contains either the term pair T_1 and T_2, or the pair T_3 and T_4, or both. The terms characterizing the stored texts may be assigned manually by trained personnel; alternatively, automatic indexing methods may be used to handle the term assignment automatically. In some systems one can avoid or circumvent the content analysis, or indexing, operation by using words contained in the texts of the documents for content identification. When all text words are used for content identification (except for common words specified on a list of excluded words), one speaks of a full text retrieval system (*1*).

The author is with the Department of Computer Science, Cornell University, Ithaca, NY 14853.

The conventional retrieval environment has become widely accepted, because the Boolean formulations can express synonymous term relationships specified by *or* operators ("minicomputers *or* microcomputers *or* hand-held calculators") or term phrases specified by *and* operators ("information *and* retrieval"). Furthermore, one obtains fast responses even for very large document collections by constructing auxiliary inverted index files and performing the search operations using list manipulations in the index. In general, the index consists of lists of document identifiers for each allowable index term: thus, all documents identified by a given term X are included in the corresponding X list in the index. To obtain responses to queries such as (T_1 *or* T_2) and (T_1 *and* T_2), the system extracts the T_1 and T_2 lists from the index and constructs a single common (T_1, T_2) list by list-merging operations. The duplicated items in the merged list then represent answers to (T_1 *and* T_2), and the unique items on the list are the answers to (T_1 *or* T_2).

The list-merge technology is well understood, and a high degree of effectiveness is sometimes obtained with conventional inverted file searches. However, the use of Boolean operators may prove disadvantageous, most importantly because users who are not trained in logic find it difficult to generate effective queries that produce the proper amount of output and the expected proportion of relevant materials. Also, the conventional Boolean logic treats all terms as equally important and all retrieved documents as equally useful. Thus, the retrieved items are presented to the user in an arbitrary order that does not normally correspond to the order of usefulness of the items. Furthermore, the Boolean logic is unusually rigid in a retrieval setting, because the presence of a single query term in a document suffices for retrieval in response to an *or* query such as (T_1 *or* T_2 *or*. . .*or* T_z), whereas the absence of a single query term from a document suffices for rejection in response to an *and* query such as (T_1 *and* T_2 *and*. . .*and* T_z).

Refinements have been introduced into the Boolean processing environment that are designed to control the query formulation process (*2*) and provide more discriminating output by allowing the terms assigned to documents (but not those assigned to queries) to carry term weights in decreasing order of presumed term importance (*3*). When term weights are introduced, as they are in the so-called fuzzy-set retrieval model, the retrieved documents can be ranked in decreasing order of the weights of certain matching query terms. However, the fuzzy-set retrieval system is still based on ordinary Boolean logic, and it carries much the same limitations as the conventional Boolean model.

Alternative Retrieval Models

In the vector space system, the documents are identified by sets of attributes, or terms, as in the Boolean system. Instead of assuming that all terms are equally valuable, the system uses term weights to assign importance indications to the terms. If t distinct terms are available for content identification, a document $\mathbf{D}_i$ is representable

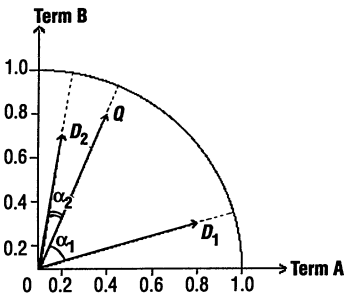

Fig. 1. Vector similarity computation. $\mathbf{D}_1 = (0.8, 0.3)$, $\mathbf{D}_2 = (0.2, 0.7)$, $\mathbf{Q} = (0.4, 0.8)$, $sim(\mathbf{Q}, \mathbf{D}_1) = \cos\alpha_1 = 0.74$, $sim(\mathbf{Q}, \mathbf{D}_2) = \cos\alpha_2 = 0.98$.

internally as a t-dimensional vector of pairs, $\mathbf{D}_i = (d_{i1}, w_{d_{i1}}; d_{i2}, w_{d_{i2}}; \ldots; d_{it}, w_{d_{it}})$, where d_{ij} represents the jth term assigned to documents D_i and $w_{d_{ij}}$ is the corresponding term weight. In principle, all t terms could appear in each vector: a weight of zero would be used for terms not present, and larger weights, between 0 and 1, would designate terms actually assigned to the items (*4*).

In the vector processing system, the Boolean queries are replaced by weighted term sets similar to those used for the document representations. Thus a query $\mathbf{Q}$ appears as $\mathbf{Q} = (q_1, w_{q1}; q_2, w_{q2}; \ldots; q_t, w_{qt})$, where once again a weight of zero is used for terms that are absent. When both the stored texts and the information requests are represented by weighted term vectors, a global, composite vector comparison can measure the degree of similarity between a query-document pair on the basis of the weights of the corresponding matching terms. The cosine measure of similarity, computed as the normal inner product between vector elements normalized for vector length, has been widely used for this purpose:

$$sim(\mathbf{Q}, \mathbf{D}_i) = \frac{\sum_{j=1}^{t} w_{qj} w_{d_{ij}}}{\sqrt{\sum_{j=1}^{t} (w_{qj})^2 \sum_{j=1}^{t} (w_{d_{ij}})^2}} \quad (1)$$

Equation 1 gives the cosine of the angle between vectors $\mathbf{Q}$ and $\mathbf{D}_i$, producing a value of 0 when no common terms exist between the vectors and a value of 1 when all terms match and the vectors are identical. A typical cosine similarity computation is illustrated in Fig. 1 for one sample query and two documents. For example, applying the formula of Eq. 1 to the query vector $\mathbf{Q} = (0.4, 0.8)$, where terms 1 and 2 receive weights of 0.4 and 0.8, respectively, and document $\mathbf{D}_2 = (0.2, 0.7)$ produces the following computation:

$$sim(\mathbf{Q}, \mathbf{D}_2) = \frac{(0.4 \cdot 0.2) + (0.8 \cdot 0.7)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.2)^2 + (0.7)^2]}} = \frac{0.64}{\sqrt{0.42}} = 0.98$$

as indicated in Fig. 1.

The vector processing model offers simple, parallel treatments for both queries and documents. The model accommodates weighted terms and provides ranked retrieval output in decreasing order of query-document similarity. Query and document vectors are also easily modified, as required for query reformulation and other purposes. On the negative side is the assumed lack of relationships between terms (formally, the vector space is assumed to be orthogonal, and hence the terms are linearly independent) and the lack of theoretical justification for some of the vector manipulations, such as the use of the cosine measure to obtain vector similarities.

The assumption that the terms are independent, which is made in other retrieval models as well, implies that the subject matter of each item is covered exhaustively by a set of mutually exclusive terms. In these circumstances, the greater the number of term matches between query and document vectors, the greater the similarity between the respective items. In practice, the terms used for indexing purposes may not be independent and may exhibit various relationships with each other. In such circumstances, the number of term matches may not be directly related to the real query-document similarity, and the computed vector similarity may not always be meaningful. In test situations, the vector space approach provides much better retrieval output than the conventional Boolean approach normally used in operational retrieval situations.

Extensions to the vector and Boolean models have been proposed, notably including a generalized vector space model based on an orthogonal vector space of dimension 2^t (with 2^t basic terms), replacing the original space of dimension t (*5*). In the generalized space, the basic terms are specified by the different maximal Boolean products of the original terms, and these products are automatically independent. Furthermore, Boolean queries are representable as easily as vector queries, so a common retrieval model is obtained that subsumes both Boolean and vector processing models (*6*).

Another common retrieval model is the extended Boolean system, which accommodates term weights assigned to both query and document terms as well as strictness indicators known as p-values that are attached to Boolean operators (*7*). A typical query formulation in the extended Boolean system would be $\{[(T_1, a)\ or^{p_1}\ (T_2, b)]\ and^{p_2}\ (T_3, c)\}$, where a, b, and c are the weights for terms T_1, T_2, and T_3, respectively, and p_1 and p_2 are p values that control the strictness of interpretation of the Boolean operators. Values of p range from 1 to ∞; the upper limit represents total strictness of interpretation, equivalent to a standard Boolean system, whereas the lower limit represents total relaxation, equivalent to a vector processing system where the distinctions between *and* and *or* are lost. The extended system thus covers vector processing, Boolean, and fuzzy-set retrieval in a common framework, and it produces vastly improved retrieval performance over simple Boolean operations at the cost of a substantially increased computational effort.

The probabilistic retrieval model differs from those previously discussed in that it represents an attempt to set the retrieval problem on firm theoretical foundations. Concepts of decision theory—a theory offering criteria for reaching decisions in situations of uncertainty—are used based on the notions of the relevance (and nonrelevance) of a document with respect to a query, to reach the conclusion that the expected usefulness of a retrieval system is optimized when the item with the highest probability of relevance is extracted from the file at each point (*8*). This leads to the well-known probability-ranking principle, which states that documents should be brought to the users' attention in decreasing order of their probability of usefulness to the users (*9*).

In the probabilistic approach it becomes necessary to estimate for each document $\mathbf{D}_i$ with respect to $\mathbf{Q}_j$ the quantity $P(Rel|\mathbf{Q}_j, \mathbf{D}_i)$, the probability of relevance of $\mathbf{D}_i$ with respect to $\mathbf{Q}_j$. One approach to this estimation process consists in regarding retrieval as an inference, or evidential reasoning process, where an answer to a user query is deduced from the evidence provided by each document (*10*). To estimate the overall measure $P(Rel|\mathbf{Q}_j, \mathbf{D}_i)$, one considers the individual term factors $P(Rel|T_k, \mathbf{D}_i)$, representing the probability that a document $\mathbf{D}_i$ will be judged relevant to a query, given that it contains query term T_k.

In the classical probabilistic models, the needed term probabilities are estimated by accumulating a number of user queries containing term T_k and determining the proportion of times document $\mathbf{D}_i$ is found relevant to the respective queries; alternatively, a fixed query $\mathbf{Q}_j$ is considered, and an attempt is made to determine the probability that an arbitrary document $\mathbf{D}_i$ containing query term T_k will be judged relevant (*11*). In either case, it is necessary to deal with a small number of query-document pairs with common terms T_k to obtain the needed term probabilities. The difficulties inherent in this estimation process and the impossibility of gathering enough relevance data before a search is actually conducted have prevented the practical implementation of most probabilistic retrieval strategies.

A suggestion recently made replaces the estimation of $P(Rel|T_k, \mathbf{D}_i)$ by an estimate of $P[Rel|x(T_k, \mathbf{D}_i)]$, where $x(T_k, \mathbf{D}_i)$ is a relevance description of term T_k that includes many factors other than simple term occurrences in documents—for example, the total number of documents containing term T_k, the total number of documents in the collection, the number of terms in document $\mathbf{D}_i$, and so on. The formulation using relevance descriptions makes available more evidence for estimation purposes. However, it is necessary to generate effective relevance descriptions for particular collection environments before the model can actually be used, and the learning process proposed for this purpose, which uses information derived from sample queries and documents, may be difficult to carry out in practical search situations (*12*).

The probabilistic retrieval approach accommodates a large number of different phenomena about terms and documents as part of the probabilistic estimation process, including term co-occurrence information, term relationships derived from dictionaries and thesauruses, and prior knowledge about the occurrence distributions of terms (*13*). The model also offers justifications for certain empirical procedures used in the vector space model—for example, the use of the inner product (the numerator of Eq. 1) to compute similarities between queries and documents (*14*) and the introduction of particular forms of term weighting in the vector system (*15*). However, the sample data and subjective relevance assessments of documents with respect to queries that are needed in probabilistic retrieval may not be available in most operational environments.

Automatic Indexing and Text Analysis

All retrieval operations depend crucially on the terms and keywords assigned to queries and text items for content representation. The assignment of terms, normally called indexing, can be performed manually by trained personnel or automatically by extraction of appropriate information from the document and query texts. The assigned terms can in principle be freely chosen, or the choice of terms can be controlled by a preexisting schedule of allowable indexing units. The following discussion is restricted to the use of automatic indexing procedures that use freely assigned vocabulary.

The simplest type of automatic indexing consists of the assignment of single-term indexing units to represent text content. A typical approach would be to identify the individual words occurring in the documents of a collection (or in the query statements). A stop list of common function words (and, of, or, but, the, and so on) would then be used to delete the high-frequency function words that are insufficiently specific to represent document content. A suffix-stripping routine would be applied to reduce the remaining words to word stem form. A weighting factor w_{ik} would be computed for each term T_k in document $\mathbf{D}_i$ to indicate term importance. Finally, each document $\mathbf{D}_i$ would be represented by a set, or vector, of weighted word stems of the kind introduced earlier (*16*). A typical stop list of English common words would include several hundred entries. Suffix deletion can similarly be based on a specially constructed short list of deletable suffixes (*17*). Suffix removal reduces entries such as analysis, analyzer, analyzing, and so forth, to a common form such as "analy" and helps reduce the size of the indexing vocabulary and the length of document vectors.

All steps in the indexing chain are straightforward except for the term-weighting operation. Term weights are used to distinguish terms that are likely to be important for content representation from other terms likely to be less important. Various term weighting theories have been proposed: the most valuable terms for retrieval purposes appear to be those able to distinguish particular documents from the remainder of the collection. This suggests that the best terms will occur frequently in particular documents, but rarely on the outside. Two main components of the term weight must therefore be distinguished: the frequency of occurrence of a term T_k in a document $\mathbf{D}_i$, also known as the term frequency, tf_{ik}, of T_k in $\mathbf{D}_i$, and the inverse document frequency, idf_k, of term T_k, which varies inversely with the number of documents to which T_k is assigned. (Typically, idf_k can be computed as $\log(N/n_k)$, where N is the total number of documents in a collection and n_k is the number of items with T_k.) These two factors can be combined by multiplication into a single factor, known as the ($tf \times idf$) weight (*4, 18*).

In addition to the term frequency and inverse document frequency, the length of each document, measured by the number of assigned terms, must also be taken into account. Otherwise, the longer documents have a better chance of being retrieved than the shorter ones because they contain more terms, and hence possibly more matching query terms. Each document is given equal chance of retrieval by normalizing the term weight and computing the weight of term T_k in document $\mathbf{D}_i$ as

$$w_{ik} = \frac{tf_{ik} \log (n_k/N)}{\sqrt{\sum_{k=1}^{t} (tf_{ik})^2 [\log(n_k/N)]^2}} \quad (2)$$

When normalized term weights such as those of Eq. 2 are used for both document and query terms, the similarity between documents, or between a query and a document, may be computed as the inner product between corresponding vector elements; that is, $sim(\mathbf{D}_i, \mathbf{D}_j)$ or $sim(\mathbf{D}_i, \mathbf{Q}_j) = \Sigma_{k=1}^{t} w_{ik} \cdot w_{jk}$. For normalized term weights, the inner product computation is then equivalent to the cosine similarity of Eq. 1.

Single-term indexing theories are easily implemented. However, substantial questions arise about the appropriateness of single-term indexing representations for text items. Indeed, critics believe that such an approach represents a dead end: "The keyword approach where absence or presence of keywords and their distributions are the only information being considered, has been typically assumed by many researchers to be sufficient: However. . .[this] approach with statistical techniques has reached its theoretical limit and further attempts for improvement are considered a waste of time" (*19*, p. 111).

Although such a claim reflects sentiment more than fact, attempts have been made to refine the text indexing process. One strategy consists in considering term specificity and replacing single terms that are too broad in scope (insufficiently specific) by term phrases composed of several term components while replacing narrow terms (terms that are too specific) by broader entities extracted from a thesaurus (*20*). Refined linguistic analysis methods are also usable for text indexing, but experience indicates that reliable improvements in retrieval effectiveness beyond the weighted single-term assignments mentioned above are hard to come by.

Linguistic- and Knowledge-Based Approaches

Documents and texts are natural language constructs. Hence, it is useful to consider various language analysis tools for text-indexing purposes. The first possibility consists in using syntactic analysis to assign one or more syntactic tags to each word of the input text. Such a tagging operation can then be followed by a phrase construction process that identifies nominal constructs consisting of appropriate noun and adjective combinations usable for the content identification of the respective texts.

Many different syntactic approaches have been used in automatic indexing and information retrieval (*21*). Unfortunately, syntax alone is unable to cope with many ambiguities in the natural language, and

the accuracy of the available syntactic procedures leaves much to be desired. This means that false syntactic constructs may be erroneously assigned for content identification, and useful phrases that correctly reflect document content may not be assignable because of constraints imposed by the syntactic process.

Consider as an example a typical sequence such as "Alphabetic (adjective) characters (plural noun) occurring (gerund) most (quantifier) frequently (adverb) in (preposition) running (gerund) text (noun) account (noun or verb) for (preposition) 85 to (preposition) 95 percent (noun) of (preposition) letter (noun) occurrences (noun)." In this sentence the phrase "letter occurrences" is easily generated as a sequence of two adjacent nouns. The generation of the other noun phrases depends on contradictory interpretations of the present participles "occurring" and "running." If these are interpreted as verb forms (gerunds), the generated noun phrases are "alphabetic characters" (correct) and "text account" (false); if, on the other hand, the two participles are interpreted as adjectives, then the generated phrases are "alphabetic characters occurring" (false) and "running text" (correct). In either case, one false phrase is generated and one important phrase is lost. Standard syntactic approaches generate only 60% of the wanted phrases correctly. Overall, noun constructions are thus more reliably obtained with statistical methods based on detectable co-occurrences between phrase components in the available texts (*22*).

Because of the uncertainties inherent in a purely syntactic approach, additional semantic criteria may be introduced in the form of dictionaries or thesauruses, providing semantic specifications for the text words. Thesauruses of many kinds have been constructed, often tailored to topic areas and designed to reveal a semantic relationship between thesaurus entries. Alternatively, useful information might be extracted from one of the available machine-readable dictionaries covering large slices of the language (*23*).

However, it is not easy to apply thesauruses and machine-readable dictionaries in practical information retrieval. The construction of thesauruses and other vocabulary specification tools is an art, and there is no guarantee that a thesaurus tailored to a particular text collection can be usefully adapted to another collection. As a result, it has not been possible to obtain reliable improvements in retrieval effectiveness by using thesauruses with a variety of different document collections. In addition, it is difficult to obtain reliable information from machine-readable dictionaries, because most dictionary entries carry multiple definitions and the relationships between multiple defining statements for a single dictionary entry are hard to assess.

A further extension in the sophistication of the text analysis is provided by the knowledge-based approaches that are popular in artificial intelligence. A knowledge base is a structure representation of the subject matter of interest in a particular area of discourse. Normally such a knowledge base includes a description of the main concepts of interest in an area as well as the properties and interrelationship between concepts. Many formalisms have been proposed for the representation of knowledge, including semantic nets, frames, scripts, and so on (*24*). Network structures are often used; in that case, concepts are represented by network nodes and concept relationships by branches between corresponding nodes. An excerpt of such a semantic network is shown in Fig. 2. A number of basic concepts describing the cardiovascular system are shown in Fig. 2, together with selected relationships between certain concepts. Thus the network specifies that the heart may be affected by blood pressure and that a heart attack is an example of a cardiovascular illness.

In information retrieval, knowledge bases are used with inference rules that provide the rules for traversing the concept network. A typical rule might be the following: if concept A is found, and concept A is related in the network to concept B by a certain type of relationship, then an additional concept C is also valid. Probability measures, or weights, may also be used in the network, and these weights may then be propagated through the network by appropriate use of the network traversal rules.

Many different approaches have been proposed for the implementation of intelligent information retrieval systems. Normally, the aim is to apply a particular semantic structure built for a particular field, together with appropriate inference rules, to derive answers to queries starting with the concepts contained in the available document descriptions (*25*). Among the techniques used to instantiate new concepts from old ones that are initially given are spreading activation [where initial activation weights attached to input nodes produce new activation weights attached to the outputs (*26*)], the calculus of generalized vector norms (*27*), advanced linguistic processing techniques (*28*), Bayesian inference techniques (*29*), and Dempster-Shafer belief theory (*30*). In all cases, the general aim is to derive a similarity value, or measure of closeness, between query and document, computed as the probability that the user's information need as expressed in the query statement can be inferred from the evidence supplied by each given document; alternatively, one measures the degree of belief in the query derivable from the available documents, that is, the degree to which the query can be satisfied by the available documents.

Substantial advantages have been claimed for these inference techniques in various areas of application, and it is likely that useful knowledge structures and reliable inference rules can in fact be generated that are valid in well-circumscribed situations for limited subject domains. However, substantial doubt remains about the viability of techniques based on complex network representations when large text collections must be processed in unrestricted subject areas. Eco and others have argued in this connection that any artificially constructed knowledge structure necessarily provides only partial and inadequate representations of meanings: "Semantic models such as that of Quillian are already. . .a portion of the universe in which a system has intervened in order to establish attractions and repulsions [between concepts, thereby favoring some concept relationships at the expense of others]" (*31*, p. 125).

In any case, complete theories of knowledge representation do not exist now. As a result, it is not clear what entities must be included in a knowledge base and what relationship between knowledge elements must be considered for particular applications. Attempts have been made to build very large knowledge bases covering unrestricted subject matter, but the applicability to large unrestricted document collections is unproven (*32*).

Another possibility for the formulation of viable text analysis systems that are valid for unrestricted text environments consists in performing detailed analyses of the available texts and incorporating in the analysis process the multiple contexts in which the words and

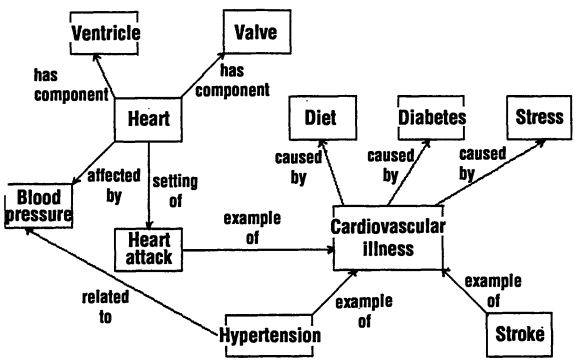

Fig. 2. Excerpt from a semantic network (topic: cardiovascular system).

expressions are used in the available texts. One remembers in this connection the pronouncements of Wittgenstein and his followers that word meaning cannot adequately be determined by consulting preconstructed dictionaries. Rather "for a large class of cases—though not for all—in which we employ the word 'meaning' it can be defined thus: the meaning of a word is its use in the language" (*33*, p. 21). This view suggests that similarities in word meaning might be ascertained by the determination of coincidences in the contexts in which the words are used in different text passages. When sufficiently large contextual similarities are detected, the conclusion follows that the word meanings in the corresponding texts are homogeneous. Documents or text passages may then be retrieved in answer to available statements of user needs by comparison of the query statements to the text passages at various levels of detail and retrieval of items that exhibit sufficiently similar global and local text similarities. Such an approach based on global comparisons between all stored texts is reminiscent of the memory-based reasoning strategies that have been advocated in other contexts. There are indications that such methods can operate with a high degree of accuracy in large text environments (*34*). A study of this approach is presented in a companion piece in this issue (*35*).

Retrieval Strategies

In conventional information retrieval environments, large collections of several hundred thousand indexed documents are routinely processed, and document references that match the available Boolean query statements are normally displayed in real time, that is, more or less instantaneously, while the users wait at the search terminal. Similar operating efficiencies are available in advanced vector processing environments, such as the Smart system, where tens of thousand of texts, corresponding to hundreds of thousands of text paragraphs or millions of sentences, are routinely manipulated in real time (*36*). The search efficiencies are in all cases attributable to the inverted index file technologies mentioned above, where all items that do not have at least one assigned term in common with the queries are immediately rejected.

Direct rather than inverted file searches can be implemented for large text files by parallel processing machines. A well-known direct file search system, operating without auxiliary index files, has been implemented on a Connection Machine with 64,000 individual processors. By storing the representation of a different document in each processor and broadcasting (sending) the same query to all processors, the system can compare a given query with 64,000 different documents in a single file comparison operation, each processor performing the query-document comparison with a different document at the same time (*37*). It remains to be seen whether the parallel processing approaches will prove cost-effective in practical retrieval environments.

When conventional inverted file technologies are used, the indexing information pertaining to a single document is dispersed in the file in many different document reference lists. This dispersal makes it impossible for users to browse through the documents or locate documents that are related to previously known texts. Related documents can be accessed together by the use of structured text files in which sufficiently similar documents are grouped in common classes. Groups, or clusters, of documents can be built by computing similarities between pairs of documents on the basis of similarity computations such as those specified in Eqs. 1 and 2 and using clustering criteria to group sufficiently similar documents (*38*).

A typical hierarchical cluster organization is shown in Fig. 3, where large document clusters are successively broken down into smaller and smaller classes of items. The distance between two x

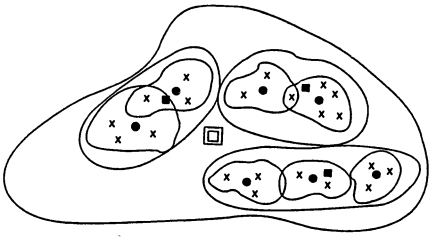

Fig. 3. Typical clustered file organization. The symbols ●, ■, and ⧈ represent centroids, and the symbol x denotes a document.

symbols is inversely related to document similarity; that is, the closer the two x symbols, the more similar are the corresponding items. By constructing special class representatives, known as centroids, and confining the file searches to clusters with centroids exhibiting large query-centroid similarities, one may generate efficient file search strategies that bypass most of the collection in any given search. Furthermore, when the cluster hypothesis is satisfied, that is, when documents that are jointly relevant to particular queries appear in common clusters, the retrieval effectiveness of clustered search techniques may also be relatively high (*39*). However, the construction, maintenance, and search of clustered files is expensive, especially for effective cluster structures consisting of many small, tightly clustered document groups. When efficiency is important, as it is in modern on-line search environments where fast responses are essential, the inverted file technology is generally preferred (*40*).

A different type of text structuring is based on the so-called hypertext model, in which large texts are broken down into linked portions of related text (*41*). Typically, complete text sections are then broken down into subsections that are further subdivided into individual paragraphs and sentences. All of these text components are then appropriately linked, and access to individual text components is obtained by appropriate use of the linked structure. Retrieval activities in hypertext may be especially useful for texts such as dictionaries and encyclopedias, textbooks, instruction manuals, and so on, that are not meant to be read sequentially (*42*). The queries are then compared in each case with identifiers corresponding to various portions of the linked structure, and links exhibiting high query similarities are followed in the search. Content-linked hypertext structures might be automatically generated on the basis of computed similarities between text portions with sufficiently high similarities (*43*).

In addition to structured file organizations, the retrieval process may be enhanced by introducing aids to the on-line search process. Thus, advanced information display options and graphic terminal equipment can help in controlling the search process (*44*), and sophisticated user-system dialogue schemes may be introduced (*45*). One especially simple and effective search strategy based on user-system interaction is the well-known relevance feedback method, in which user queries are automatically reformulated on the basis of relevance judgments obtained from the user for certain documents retrieved in earlier searches (*46*). By adding to the query terms from relevant documents retrieved previously or increasing the weight of such terms and similarly diminishing the importance of terms contained in nonrelevant documents retrieved previously, one obtains new query formulations that are more similar to the previously identified relevant documents and less similar than before to the identified nonrelevant items. These feedback queries can produce enhancement in retrieval effectiveness ranging from 50 to well over 100% (*47*). An illustration of the relevance feedback process appears as Fig. 4. Three retrieved documents are represented by x symbols and are assumed to have been designated as relevant by the user. The relevance feedback process builds a new query statement (the closed triangle) that is much more similar (much closer) to the previously retrieved documents than the original query. This new query is

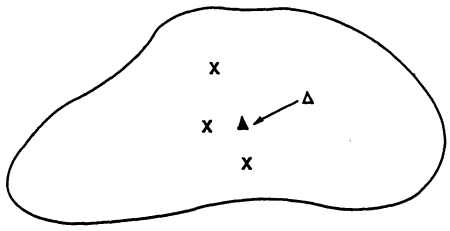

Fig. 4. Relevance feedback illustration. Original query, △; retrieved item, x; altered query, ▲.

expected to retrieve new relevant documents that are similar to relevant items retrieved earlier.

Methods analogous to those used for relevance feedback are also usable for the modification of document vectors (*48*). Operations with dynamic document spaces using expert system technologies for user-system interaction may form the basis of many information retrieval activities in the future.

In the early years of automatic text processing the feeling was widespread that it would never be possible to design useful retrieval protocols capable of performing satisfactorily in unrestricted text environments: "let it be immediately stressed that. . .neither the assignment of topic terms to a given request, nor the reformulation of a given request are processes which could conceivably be adequately mechanized, contrary to some speculation in this direction. . ." (*49*, p. 344). It is still not possible for computers to perform certain complex text processing tasks with the benefit of a complete understanding of text content. However, it is not difficult to identify useful relationships between different texts and in particular between text items and related search requests. The time is at hand when sophisticated searches can be conducted with large collections of natural language text in unrestricted subject areas that can provide high-quality, rapid file access for interested users.

REFERENCES AND NOTES

1. G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval* (McGraw-Hill, New York, 1983); F. W. Lancaster, *Information Retrieval Systems—Characteristics, Testing and Evaluation* (Wiley, New York, ed. 2, 1979); C. J. van Rijsbergen, *Information Retrieval* (Butterworths, London, ed. 2, 1979); G. Salton, *Automatic Text Processing—The Transformation, Analysis and Retrieval of Information by Computer* (Addison-Wesley, Reading, MA, 1989).
2. C. W. Cleverdon, *Inf. Serv. Use* **4**, 37 (1984).
3. T. Radecki, *Inf. Process. Manage.* **12**, 313 (1976); A. Bookstein, *J. Am. Soc. Inf. Sci.* **31**, 240 (1980); D. A. Buell and D. H. Kraft, *ibid.* **32**, 211 (1981).
4. G. Salton, C. S. Yang, A. Wong, *Commun. Assoc. Comput. Mach.* **18**, 613 (1975); G. Salton, C. S. Yang, C. T. Yu, *J. Am. Soc. Inf. Sci.* **26**, 33 (1975); G. Salton, *Regional Conference Series in Applied Mathematics* (Society of Industrial and Applied Mathematicians, Philadelphia, 1975), vol. 18.
5. V. V. Raghavan and S. K. M. Wong, *J. Am. Soc. Inf. Sci.* **37**, 279 (1986); S. K. M. Wong, W. Ziarko, P. C. N. Wong, in *Proceedings of the Eighth Annual International Conference for Research and Development in Information Retrieval*, Montreal, 5 to 7 June 1985 (Association for Computing Machinery, New York, 1985), pp. 18–25.
6. S. K. M. Wong, W. Ziarko, V. V. Raghavan, P. C. N. Wong, in *Proceedings of Ninth Annual International Conference for Research and Development in Information Retrieval*, F. Rabitti, Ed., Pisa, Italy, 8 to 10 September 1986 (Association for Computing Machinery, New York, 1986), pp. 175–185.
7. G. Salton, E. A. Fox, H. Wu, *Commun. Assoc. Comput. Mach.* **26**, 1022 (1983); G. Salton, E. A. Fox, E. Voorhees, *J. Am. Soc. Inf. Sci.* **36**, 200 (1985); G. Salton and E. Voorhees, in *Proceedings of the Eighth Annual International Conference on Research and Development in Information Retrieval*, Montreal, 5 to 7 June 1985 (Association for Computing Machinery, New York, 1985), pp. 54–69.
8. A. Bookstein and D. Swanson, *J. Am. Soc. Inf. Sci.* **26**, 45 (1975); W. S. Cooper and M. E. Maron, *J. Assoc. Comput. Mach.* **25**, 67 (1978).
9. S. E. Robertson, *J. Doc.* **33**, 294 (1977).
10. A. Bookstein, *Annu. Rev. Inf. Sci. Technol.* **20**, 117 (1985); W. S. Cooper, *Inf. Storage Retr.* **7**, 19 (1971).
11. M. E. Maron and J. L. Kuhns, *J. Assoc. Comput. Mach.* **7**, 216 (1960); S. E. Robertson and K. Sparck-Jones, *J. Am. Soc. Inf. Sci.* **27**, 129 (1976); S. E. Robertson, M. E. Maron, W. S. Cooper, *Inf. Technol. Res. Dev.* **1**, 1 (1982).
12. N. Fuhr and C. Buckley, in *Proceedings of the Thirteenth Annual International Conference on Research and Development in Information Retrieval*, J. L. Vidick, Ed., Brussels, Belgium, 5 to 7 September 1980 (Association for Computing Machinery, New York, 1990), pp. 45–61; P. Biebricher, N. Fuhr, G. Lustig, M. Schwantner, G. Knorz, in *Proceedings of the Eleventh Annual International Conference on Research and Development in Information Retrieval*, Y. Chiaramella, Ed., Grenoble, France, 13 to 15 June 1988 (Association for Computing Machinery, New York, 1988), pp. 333–342.
13. C. J. van Rijsbergen, *J. Doc.* **33**, 106 (1977).
14. A. Bookstein, *Research and Development in Information Retrieval*, vol. 146 in *Lecture Notes in Computer Science*, G. Salton and H. J. Schneider, Eds. (Springer-Verlag, Berlin, 1983), pp. 118–132.
15. W. B. Croft and D. J. Harper, *J. Doc.* **35**, 285 (1979); H. Wu and G. Salton, *ACM SIGIR Forum* **16**, 30 (1981).
16. G. Salton, *ACM SIGIR Forum* **16**, 22 (1981).
17. J. B. Lovins, *Mech. Transl. Comput. Linguist.* **11**, 11 (1968).
18. G. Salton and C. S. Yang, *J. Doc.* **29**, 351 (1973); G. Salton and C. Buckley, *Inf. Process. Manage.* **24**, 513 (1988).
19. T. M. T. Sembok and C. J. van Rijsbergen, *Inf. Process. Manage.* **26**, 111 (1990).
20. K. Sparck-Jones, *J. Doc.* **28**, 11 (1972); *Keyword Classification for Information Retrieval* (Butterworths, London, 1971).
21. M. Dillon and A. S. Gray, *J. Am. Soc. Inf. Sci.* **34**, 99 (1983); L. S. Gay and W. B. Croft, *Inf. Process. Manage.* **26**, 21 (1990); K. Sparck-Jones and J. I. Tait, *J. Doc.* **40**, 50 (1984); A. F. Smeaton and C. J. van Rijsbergen, in *Proceedings of the Eleventh Annual International Conference on Research and Development in Information Retrieval*, Y. Chiaramella, Ed., Grenoble, France, 13 to15 June 1988 (Association for Computing Machinery, New York, 1988), pp. 31–51.
22. J. Fagan, *J. Am. Soc. Inf. Sci.* **40**, 115 (1989); G. Salton, C. Buckley, M. Smith, *Inf. Process. Manage.* **26**, 73 (1990).
23. D. E. Walker and R. A. Amsler, in *Analysing Language in Restricted Domains*, R. Grishman and R. Kittredge, Eds. (Erlbaum, Hillsdale, NY, 1986); E. A. Fox, J. T. Nutter, T. Ahlswede, M. Evens, J. Markowitz, in *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, TX, 9 to 12 February 1988 (Association for Computational Linguistics, Morristown, NY, 1988), pp. 101–108; R. A. Amsler, *Annu. Rev. Inf. Sci. Technol.* **19**, 161 (1984).
24. M. R. Quillian, in *Semantic Information Processing*, M. Minsky, Ed. (MIT Press, Cambridge, MA, 1968), pp. 216–270; N. Findler, Ed., *Association Networks: The Representation and Use of Knowledge by Computers* (Academic Press, New York, 1979); R. Brachman and H. Levesque, *Readings in Knowledge Representation* (Morgan-Kaufmann, Los Altos, CA, 1985); R. Schank, *Conceptual Information Processing* (North-Holland, Amsterdam, 1975).
25. K. Sparck-Jones, in *Intelligent Information Retrieval: Informatics 7*, K. P. Jones, Ed., Cambridge, 22 to 23 March 1983 (ASLIB, London, 1983), p. 136; C. J. van Rijsbergen, *Comput. J.* **29**, 481 (1986); L. F. Rau, *Inf. Process. Manage.* **23**, 269 (1987); Y. Chiaramella and B. Defude, *ibid.*, p. 285; C. Brajnik, G. Guida, C. Tasso, *ibid.*, p. 305; W. B. Croft, *ibid.*, p. 249.
26. P. R. Cohen and R. Kjeldsen, *Inf. Process. Manage.* **23**, 255 (1987).
27. R. M. Tong, L. A. Appelbaum, V. N. Askman, J. F. Cunningham, in *Proceedings of the Tenth Annual International Conference on Research and Development in Information Retrieval*, C. T. Yu and C. J. van Rijsbergen, Eds., New Orleans, LA, 3 to 5 June 1987 (Association for Computing Machinery, New York, 1987), pp. 247–253.
28. P. S. Jacobs and L. F. Rau, in *Proceedings of the Eleventh Annual International Conference on Research and Development in Information Retrieval*, Y. Chiaramella, Ed., Grenoble, France, 13 to 15 June 1988 (Association for Computing Machinery, New York, 1988), pp. 85–99.
29. H. Turtle and W. B. Croft, in *Proceedings of the Thirteenth Annual International Conference for Research and Development in Information Retrieval*, J. L. Vidick, Ed., Brussels, Belgium, 5 to 7 September 1990 (Association for Computing Machinery, New York, 1990), pp. 1–24.
30. G. Biswas, J. C. Bezdek, V. Subramanian, M. Marques, *J. Am. Soc. Inf. Sci.* **38**, 83 (1987).
31. U. Eco, *A Theory of Semiotics* (Indiana Univ. Press, Bloomington, IN, 1976).
32. D. Lenat, M. Prakash, M. Shepherd, *AI Mag.* **6**, 65 (1986).
33. L. Wittgenstein, *Philosophical Investigations* (Basil Blackwell and Mott, Oxford, 1953).
34. C. Stanfill and D. L. Waltz, *Commun. Assoc. Comput. Mach.* **29**, 1213 (1986); G. Salton and C. Buckley, "Approaches to Global Text Analysis," *Technical Report TR 90-1113* (Computer Science Department, Cornell University, Ithaca, NY, 1990).
35. G. Salton and C. Buckley, *Science* **253**, 1012 (1991).
36. G. Salton, Ed., *The Smart Retrieval System—Experiments in Automatic Document Processing* (Prentice-Hall, Englewood Cliffs, NJ, 1971).
37. C. Stanfill and B. Kahle, *Commun. Assoc. Comput. Mach.* **29**, 1229 (1986); C. Stanfill, in *Proceedings of the Thirteenth Annual International Conference on Research and Development in Information Retrieval*, J. L. Vidick, Ed., Brussels, Belgium, 5 to 7 September (Association for Computing Machinery, New York, 1990), pp. 413–428; G. Salton and C. Buckley, *Assoc. for Comput. Mach. Commun.* **31**, 202 (1988).
38. A. Griffiths, L. A. Robinson, P. Willett, *J. Doc.* **40**, 175 (1984); F. Murtagh, *Comput. J.* **26**, 354 (1982); W. B. Croft, *J. Am. Soc. Inf. Sci.* **28**, 341 (1977).
39. E. M. Voorhees, in *Proceedings of the Eighth Annual International Conference on Research and Development in Information Retrieval*, Montreal, 5 to 7 June 1985 (Association for Computing Machinery, New York, 1985), pp. 188–196.
40. _____, in *Proceedings of the Ninth Annual International Conference on Research and Development in Information Retrieval*, F. Rabitt, Ed., Pisa, Italy, 8 to 10 September 1986 (Association for Computing Machinery, New York, 1986), pp. 164–174.
41. J. Conklin, *Computer* **20**, 17 (1987); R. Furuta, *Comput. J.* **32**, 493 (1989).
42. D. R. Raymond and F. W. Tompa, *Commun. Assoc. Comput. Mach.* **31**, 871 (1988); M. E. Frisse, *ibid.*, p. 880; W. B. Croft and H. Turtle, in *Proceedings of Hypertext '89*, Pittsburgh, PA, 5 to 8 November 1989 (Association for Computing Machinery, New York, 1989), pp. 213–224.
43. G. Salton and C. Buckley, "Approaches to Text Retrieval for Structured Documents," *Technical Report 90-1083* (Computer Science Department, Cornell Univer-

sity Ithaca, NY, 1990); ———, Z. Zhao, "Text Linking and Retrieval: Experiments for Textbook Components," *Technical Report 90-1125* (Computer Science Department, Cornell University, 1990).
44. D. C. Crouch, in *Proceedings of the Ninth Annual International Conference on Research and Development in Information Retrieval* (Association for Computing Machinery, New York, 1986), pp. 58–67; T. E. Doczkocs, *ibid.*, pp. 49–57; P. Ingwersen and I. Wormell, *ibid.*, pp. 68–76.
45. R. N. Oddy, *J. Doc.* **33**, 1 (1977); S. E. Pollitt, *ASLIB Proc.* **36**, 229 (1984); G. Guida and C. Tasso, *Automatica* **19**, 759 (1983).
46. J. J. Rocchio, in *The Smart System—Experiments in Automatic Document Processing*, G. Salton, Ed. (Prentice-Hall, Englewood Cliffs, NJ, 1971), pp. 313–323; G. Salton, *ibid.*, pp. 324–336; E. Ide, *ibid.*, pp. 337–354; S. K. M. Wong and Y. Y. Yao, *J. Am. Soc. Inf. Sci.* **41**, 334 (1990).
47. G. Salton and C. Buckley, *J. Am. Soc. Inf. Sci.* **41**, 288 (1990).
48. T. C. Brauen, in *The Smart Retrieval System—Experiments in Automatic Document Processing*, G. Salton, Ed. (Prentice-Hall, Englewood Cliffs, NJ, 1971), pp. 456–484.
49. Y. Bar-Hillel, Ed., *Language and Information—Selected Essays on Their Theory and Application* (Addison-Wesley, Reading, MA, 1964), pp. 330–364.
50. This study was supported in part by NSF grant IRI 89-15847.

Animal Choice Behavior and the Evolution of Cognitive Architecture

LESLIE A. REAL

Animals process sensory information according to specific computational rules and, subsequently, form representations of their environments that form the basis for decisions and choices. The specific computational rules used by organisms will often be evolutionarily adaptive by generating higher probabilities of survival, reproduction, and resource acquisition. Experiments with enclosed colonies of bumblebees constrained to foraging on artificial flowers suggest that the bumblebee's cognitive architecture is designed to efficiently exploit floral resources from spatially structured environments given limits on memory and the neuronal processing of information. A non-linear relationship between the biomechanics of nectar extraction and rates of net energetic gain by individual bees may account for sensitivities to both the arithmetic mean and variance in reward distributions in flowers. Heuristic rules that lead to efficient resource exploitation may also lead to subjective misperception of likelihoods. Subjective probability formation may then be viewed as a problem in pattern recognition subject to specific sampling schemes and memory constraints.

THE EMERGING FIELD OF COGNITIVE SCIENCE ATTEMPTS TO explain the nature of thought and the appearance of intelligence. Cognitive analyses have mostly been applied to language capabilities and the acquisition of skills in humans (*1*), but have been expanded to include problem-solving and communication in animals (*2–5*). The cognitivist view suggests that the processing of information (by either animals or humans) involves three stages. First, sensory data are translated and encoded into a form that can be manipulated through mental operations. Second, encoded information is acted upon by specific computational rules. And third, these rules produce alternative "representational" states that depend on the informational input. The concept of "representation" remains controversial, especially for animals (*5*). However, these three stages may be viewed, in a less controversial manner, as three components of a single dynamical system mechanistically tied to the organism's nervous system. The encoding of information would then correspond to initial inputs, computational rules correspond to transient dynamics, and representations would correspond to the equilibrium configurations resulting from the transient dynamics. The animal reaches a representation of the environment through the operation of specific computational rules applied to a particular pattern of incoming sensory information.

The computational rules used by organisms can be symbol-processing programs, as in most artificial intelligence models (*6*), or can be models of nervous systems, as in neural networks (*7*). My thesis is that these computational rules are evolutionarily adaptive. Different computational schemes may generate behaviors or representation of the environment that lead to different efficiencies in the use of resources, acquisition of mates, or acquisition of skills necessary for survival. Differential efficiencies may then confer different evolutionary advantages. The design features of information-processing ("cognitive architecture") may be subject to natural selection in a manner analogous to any other aspect of the organism's phenotype.

The link between mental process, cognition, and evolution originates in Darwin's writings (*8*) and has found continuous support from many investigators since the Darwinian revolution (*9*). Many more recent studies have explicitly examined the adaptive nature of specific mental processes in animals and have argued for varying degrees of adaptive specialization in mental function to accommodate specific ecological requirements (*10*). Few studies, however, have explicitly examined specific computational rules in the evolutionary ecology of organisms, though the adaptive nature of computational rules has been proposed (*3*). In this article, I summarize research on floral choice behavior in bumblebees (*Bombus* spp.) and argue for an evolutionary basis for the computational rules employed by bees as they acquire floral resources in their natural environment.

Bumblebee as a Model System

The choice of bumblebees as model experimental organisms was not arbitrary. Bumblebees have many features which make them ideal for examining the evolution of decision-making processes. Individual worker bumblebees are almost exclusively engaged in a

The author is in the Department of Biology, University of North Carolina, Chapel Hill, NC 27599-3280.