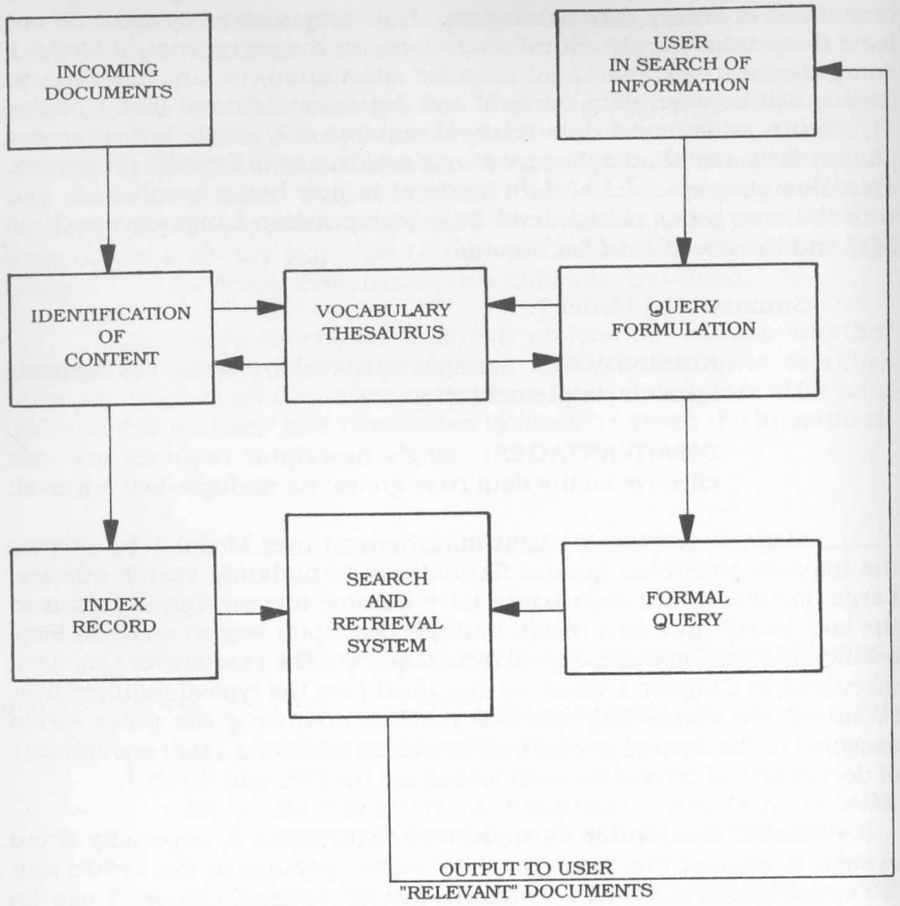


Figure 2.1

MODEL 2

I. REQUEST: A SET OF DESCRIPTORS.

II. INDEXING ASSIGNMENTS: A SET OF DESCRIPTORS.

III. DOCUMENTS ARE EITHER RETRIEVED OR NOT.

IV. RETRIEVAL RULE: DOCUMENT IS RETRIEVED IF *ALL* THE DESCRIPTORS IN THE REQUEST ARE IN THE INDEX RECORD OF THE DOCUMENT.

e.g.: REQUEST = D_k and D_j

DOCUMENT A = $< D_a, D_k, D_m >$ not retrieved

DOCUMENT B = $< D_a, D_k, D_j >$ retrieved

is equivalent to:

$D_k \cdot -(-D_j \cdot -D_l)$ [using DeMorgan's Theorem with "-" equalling "not"]

Thus, Model 2 is a "complete" Boolean retrieval system. Ordinarily, though, it is presumptuous to expect the average inquirer to be able to make such transformations readily, so Model 2 can be made more realistic by permitting the descriptors to be non-atomic disjunctions of related terms. For example:

$D_k = (D_k \vee D_j \vee \ldots \vee D_n)$

Summarizing Model 2:

ADVANTAGES: Enables more flexible searching. (permits multiple descriptor requests)

DISADVANTAGES: Cumbersome to implement on a manual system. Logical transformations may be difficult for the average inquirer to make.

MODEL 3

I. REQUEST: SET OF DESCRIPTORS PLUS A "CUT OFF VALUE".

II. INDEXING ASSIGNMENTS: A SET OF ONE OR MORE DESCRIPTORS.

III. DOCUMENTS ARE EITHER RETRIEVED OR NOT.

IV. RETRIEVAL RULE: DOCUMENT IS RETRIEVED IF IT SHARES A NUMBER OF DESCRIPTORS WITH THE REQUEST THAT *EXCEEDS* THE CUT-OFF VALUE.

e.g.: REQUEST = D_k, D_j AND D_m

CUTOFF = 1

DOCUMENT A = < D_a, D_j, D_o >	not retrieved
DOCUMENT B = < D_a, D_k, D_j, D_m >	retrieved
DOCUMENT C = < D_b, D_c, D_j, D_m >	retrieved

Models 1 and 2 retrieved documents only if all of the terms in the search request were assigned to the retrieved documents. Model 3 relaxes this retrieval constraint by permitting documents to be retrieved if their assigned descriptors match a subset of the set of specified search terms. This means, of course, that the match between the formal search quest and the sets of descriptors assigned to retrieved documents does not have to be exact. Since there may be a significant amount of indeterminacy in the assignment of descriptors to documents, it follows that more flexible retrieval rules, such as in Model 3 also permit the inquirer to expand or limit his search by altering the cutoff value. For example, the inquirer could begin by submitting the request in the example of Model 3 (*Supra*) but with a cut-off value of 2 instead of 1. This request would retrieve only those documents to which descriptors D_k, D_j, and D_m are assigned (thereby providing retrieval like a Model 2 system). If the inquirer then wanted to search further, he could resubmit

his request but now with a cut-off value of 1. This would cause the system to retrieve all documents with any one of the following descriptor combinations:

$D_k \cdot D_j \cdot D_m$

$D_k \cdot D_j$

$D_k \cdot D_m$

$D_j \cdot D_m$

(The documents retrieved with $D_k \cdot D_j \cdot D_m$ will be the same as those retrieved in the previous search.) By working out the request term combinations automatically, Model 3 removes much of the burden of query reformulation from the inquirer, and this, consequently, makes the inquirer's interface with the system simpler. But this flexibility is both the virtue and the difficulty with Model 3. A naive inquirer might include a large number of descriptors in his request along with a low cut-off value. If he were to include 7 descriptors in his request and a cut-off value of 4, 29 different subsets of the request terms would be matched against the documents in the data bases. In other words, as the inquirer lowers the cut-off value in his request, the number of subsets of the request terms which the system must generate increases combinatorially. That is, for a request of n descriptors and a cut-off value of r, the number of request subsets which the system must generate is equal to:

$$\sum_{i=r+1}^{n} C_i^n$$

This combinatorial expansion of requests can impose severe processing constraints on the system, so it is imperative that inquirers who use Model 3 systems understand this. Another problem results from this procedure of generating requests from different combinations of request terms: Many of the retrieved documents will be retrieved more than once (that is, they will be matched with more than one set of request term combinations). For example, the document which matches the request terms < D_a, D_b, D_c, D_d > (cut-off value = 1), will be retrieved by *all* the following combinations of request terms generated as search queries:

$D_a \cdot D_b \cdot D_c \cdot D_d$

$D_a \cdot D_b \cdot D_c$

$D_a \cdot D_b \cdot D_d$

$D_a \cdot D_c \cdot D_d$

$D_b \cdot D_c \cdot D_d$

$D_a \cdot D_b$

$D_a \cdot D_c$

$D_a \cdot D_d$

$D_b \cdot D_c$

$D_b \cdot D_d$

$D_c \cdot D_d$

This document, then, could appear 11 times in the retrieved set, which only serves to increase the amount of irrelevant material the inquirer must look through (naturally the document is only potentially relevant the *first* time the inquirer sees it). Every document which matches 2 or more terms above the cut-off value will be retrieved by multiple request term combinations, like the above document. There are two ways to eliminate this redundancy:

1. A running record can be maintained of the unique document identifiers of the retrieved documents, and as documents are retrieved based on new combinations of search terms, the new documents' keys are checked against the keys of previously retrieved documents so that they may be excluded if they have been retrieved already.

2. Another method is to change each of the request term combinations to its equivalent *complete conjunctive normal form*. For example:

$$D_a \cdot D_b \cdot D_c \text{ -----> } D_a \cdot D_b \cdot D_c \cdot \text{-}D_d$$

$$D_a \cdot D_b \cdot D_d \text{ -----> } D_a \cdot D_b \cdot \text{-}D_c \cdot D_d$$

. .

. .

. .

$$D_c \cdot D_d \text{ -----> } \text{-}D_a \cdot \text{-}D_b \cdot \text{-}D_c \cdot D_d$$

The sets of documents matching the Complete Conjunctive Normal Form Combinations will be, by definition, disjoint, so no running record of retrieved documents need be kept. From a processing point of view, alternative number 1 would probably be faster, but this might not always be the case, so some benchmarking might be necessary to determine which alternative would be better for a particular system. There is one final drawback to Model 3. It treats all the descriptors as if they were of equal value, or weight. This means that the documents in the retrieved sets will not be ranked or ordered. In other words, for a request of 5 descriptors with a cut-off value of 1, documents which match all 5 request descriptors are not automatically ranked higher than documents which match only 2 request descriptors. This would force the inquirer to examine all the documents in a given retrieved set to be assured that he has seen the best of those retrieved documents.

Summarizing Model 3:

> ADVANTAGES: Match between request and assigned document descriptors does not have to be exact; simple inquirer interface.
>
> DISADVANTAGES: Combinatorial increase in request subset generation may cause severe processing problems; retrieved documents are not ranked; documents may be retrieved redundantly.

As we mentioned in our discussion of Model 3, the system may have to generate quite a few search requests (combinations of search terms) if the inquirer includes many descriptors in his request along with a low cut-off value. These combinations of request terms are, in effect, separate formal search queries, and by submitting so many queries in combination there is a good chance that the retrieved set of documents will be exceptionally large--certainly larger than typical retrieved sets of Models 1 and 2. By not ranking the retrieved documents, Model 3 makes it very hard for the inquirer to satisfy this Futility Point Criterion (Chapter 1, *Supra*). Model 4 attempts to allevi-

ate this problem by ranking the retrieved documents according to their degree of overlap between the document descriptors and the set of request terms, so that documents with all the request descriptors will be ranked first, documents with any combination of one fewer request terms will be ranked second, etc. This means that even though the retrieved set of documents may have a greater number of documents than the inquirer's "futility point" (*vid.* Chapter 1), the inquirer is not discouraged from looking through it since the documents are ranked according to decreasing overlap with the request (which, hopefully, also means that they are ranked according to decreasing usefulness, though this is by no means certain).

In spite of the improvement over Model 3, Model 4 does have some drawbacks. In the first place, the amount of processing necessary to perform the retrieval has increased from Model 3 to Model 4, due to the retrieved set ranking procedures. Another problem in that like Model 3 many of the retrieved documents may be retrieved more than once, and this will require significant processing to alleviate.

A final problem is that while Model 4 ranks retrieved documents, it treats all of the terms as being equally important. That is, documents which match the same *number* of request terms will be ranked equally, regardless of which terms they match. Since most inquirers would admit that they perceive some request terms as "better" than others, it follows that documents which match, say, two of the better terms should be ranked more highly than documents which match two of the less preferred terms. Model 4 does not do this.

Summarizing Model 4:

ADVANTAGES: Documents are ranked within retrieved sets.

DISADVANTAGES: Significant increase in processing time and complexity (due to ranking procedures); Model 4 does not discriminate between more important and less important search terms.

MODEL 4

I. REQUEST: SET OF DESCRIPTORS PLUS A "CUT OFF VALUE".

II. INDEXING ASSIGNMENTS: A SET OF ONE OR MORE DESCRIPTORS.

III. RETRIEVED DOCUMENTS ARE RANKED.

IV. RETRIEVAL RULE: DOCUMENTS SHOWING WITH THE REQUEST *MORE* THAN THE SPECIFIED NUMBER OF DESCRIPTORS ARE RANKED IN ORDER OF DECREASING OVERLAP.

e.g.: REQUEST = D_k, D_j AND D_m

CUT-OFF = 1

DOCUMENT A = < D_a, D_j >	not retrieved
DOCUMENT B = < D_b, D_j, D_m >	retrieved, ranked 2nd
DOCUMENT C = < D_a, D_j , D_k, D_m >	retrieved, ranked 1st

While Model 4 did not permit the inquirer to indicate his preference for some of his selected terms over others, Model 5 permits the inquirer to assign to each request term a weight that represents the relative importance of the term in his search. The retrieved documents are then ranked according to these weights. Some care must be taken in choosing the range of numbers from which the request term weights can be selected. If any positive number may be selected as a weight then the inquirer must be careful not to select weights which will overpower the weights of the other request terms. For example, if the request terms and their respective weights are:

MODEL 5 (WEIGHTED REQUESTS)

I. REQUEST: SET OF DESCRIPTORS EACH OF WHICH HAS A POSITIVE NUMBER ASSOCIATED WITH IT.

II. INDEXING ASSIGNMENTS: A SET OF ONE OR MORE DESCRIPTORS.

III. RETRIEVED DOCUMENTS ARE RANKED.

IV. RETRIEVAL RULE: DOCUMENTS ARE RANKED IN DECREASING ORDER OF THE SUM OF THE WEIGHTS OF DESCRIPTORS COMMON TO THE REQUEST AND THE INDEX RECORD.

e.g.: REQUEST = D_k (7), D_j (4), D_m (2)

RANKING			STATUS VALUE
LAST	$DOC_a = <D_a, D_b>$	=	0
2nd	$DOC_b = <D_a, D_k, D_o>$	=	7
1st	$DOC_c = <D_k, D_j, D_o>$	=	11
3rd	$DOC_d = <D_b, D_j, D_m, D_o>$	=	6

D_a (500)

D_b (10)

D_c (3)

D_d (2)

then there will be, in effect, two separate ranked groups: those documents which are indexed with D_a, and those which are not indexed with D_a. The presence of D_a with a document overshadows any other combination of request terms. By assigning such a high weight to a single term, the inquirer is, in effect, making the retrieval system perform more like a Model 1 system than a Model 5. For this reason, the range of the weighting values should be limited to, for example, the seven point ranking scale popular in psychology, or a set of normalized values between 0 and 1. The normalized scale is useful if a probabilistic interpretation of the weights is taken.[3]

One of the major drawbacks of Model 5 is that while there are compelling reasons why request term weighting is a good idea, the typical inquirer may find the process of assigning these weights confusing. Since there are no benchmarks by which to estimate the improvement of retrieval effectiveness which Model 5 offers over simpler models, it may be difficult to convince the reluctant inquirer to expend the effort to learn how to estimate the weights. One way to handle this is to modify Model 5 so that it operates in both a naive and a normal mode. In the normal mode, Model 5 would operate in the way we have described it. In the naive mode, the inquirer would supply the request terms in the order in which he prefers them and the system would then assign a reasonable set of weights to the terms to reflect this preference ordering (of course, determining what these "reasonable" weights are is not a trivial problem). If the weights are given a probabilistic interpretation, then the system could be given an interface which would negotiate with the inquirer, and, using techniques popular in formal decision theory[4], elicit subjective probabilities from the inquirer by offering him hypothetical alternatives of request term use.

Still another possibility is to treat the weighting of request terms like natural language hedges (e.g., "very...", "somewhat...", "exceptionally...", etc). The statement "I would like a document that is mostly concerned with Data Bases and somewhat concerned with Distributed Systems" could be translated into request term weights of, for example, .8 for Data Bases and .3 for Distributed Systems. There is an intuitive appeal to such an interpretation of term weights. For the inquirer to specify a set of request terms without any accompanying weights is like asking him to describe something in detail without the use of any natural language hedges--everything must be either "good" or "bad", "useful" or "useless", "about computers", or "not about computers", etc.--and this is a very awkward way to try to describe something. The obvious problem with the analog of term weights as hedges is that it may be quite difficult to determine accurate quantitative interpretations for commonly used hedges. Still, we are not totally in the dark. Zadeh[5] has already done extensive work on the quantification of hedges in natural language using Fuzzy Sets theory. It would be interesting to test this

work by developing an inquirer interface which frees the searcher from having to assign numerical weights to request terms.

Summarizing Model 5:

ADVANTAGES: Ranking of retrieved documents reflects the relative importance of the request terms to the inquirer.

DISADVANTAGES: Weighting scale must be carefully selected; inquirers may find the estimation of term weights difficult.

One of the problems faced by indexers (or automatic indexing procedures) is the Procrustean nature of traditional binary indexing assignments, in which the indexer is often faced with the decision of whether or not to assign an index term that only marginally describes the content of the document being indexed. If the indexer does not assign the term (e.g., I_j) to the document (e.g., D_i) then there is the possibility that an inquirer who would find D_i useful would use I_j as a term in his "anchor set" of request terms (*vid.* Chapter 1) and thereby not be able to retrieve D_i. (In other words, the inquirercannot satisfy the Prediction Criterion). On the other hand, if the indexer assigns index terms to documents even when they only marginally describe the content of these documents, then the number of documents being retrieved when these terms are used as search requests will go up signifi-cantly. This increase in term breadth on a large system makes it harder for the inquirer to satisfy the Futility Point Criterion of his search (*vid.* Chapter 1).

Model 6 provides a solution to the problems engendered by binary index term assignments by permitting the indexer to weight his term assignments to documents being indexed. There is a problem, of course, in selecting the right scale for the weights, just as there was for Model 5 and estimating weights for assigned indexing terms is not a trivial procedure. Nevertheless, the weighting of index terms, in itself, offers several advantages. First of all, Model 6 permits finer retrieval rankings to be made than unweighted retrieval models, and it does so without complicating the inquirer's interface with the system, as Model 5 does. Another advantage for Model 6 is that by weighting the index

MODEL 6 (WEIGHTED INDEXING)

I. REQUEST: SET OF DESCRIPTORS.

II. INDEXING ASSIGNMENT: SET OF DESCRIPTORS EACH OF WHICH HAS A POSITIVE NUMBER ASSIGNED TO IT.

III. RETRIEVED DOCUMENTS ARE RANKED.

IV. RETRIEVAL RULE: DOCUMENTS ARE RANKED IN DECREASING ORDER OF THE SUM OF THE WEIGHTS OF DESCRIPTORS COMMON TO THE REQUEST AND THE INDEX RECORD.

e.g.: REQUEST = D_k, D_j, D_m

RANKING			STATUS VALUE
LAST	$DOC_a = < D_a (7), D_b (3) >$	=	0
1st	$DOC_b = < D_a (4), D_k (6), D_o (1) >$	=	6
2nd	$DOC_c = < D_a (1), D_k (4), D_o (6) >$	=	4
3rd	$DOC_d = < D_a (3), D_k (1), D_o (2) >$	=	3

terms, it enables the inquirer to submit single term search requests to the system even if the individual breadths of those terms exceed the Futility Point of the inquirer. That is, even though a request may retrieve a thousand documents, since they are ranked by assigned term weight, the more "highly weighted" documents would be at the beginning of the retrieved set. The inquirer, then, would merely look through this retrieved set until the documents began to look increasingly less useful. He would then be reasonably certain that the remainder of the retrieved set need not be examined. If the inquirer can perform most of his searching with single term requests, then he will avoid the "anchoring bias" which we discussed in Chapter 1.

Model 6, unfortunately, also has some drawbacks. The process of weighting index terms may significantly increase the time required to index new documents. Since document indexing is a major bottleneck for information retrieval systems, any procedure which slows it down even more must be carefully weighed against the advantages it hopes to create. Automatic index term weighting procedures, such as suggested by Sparck Jones[6], may get around this problem.

Another problem which may occur is that the indexer and inquirer may disagree significantly about how much an assigned index term should be weighted in reference to a particular document. If this disagreement is consistent across a large portion of the indexer and inquirer populations, then the inquirers can no longer be assured that the first part of a large retrieved set of documents will contain the documents that are most likely to be useful.

Summarizing Model 6:

ADVANTAGES: Discriminates important terms without complicating the user interface; ranking of documents permits retrieved sets larger than the inquirer's Futility Point.

DISADVANTAGES: Indexing process may take longer; indexer and inquirer may consistently disagree about the weights of assigned index terms.

Models 5 and 6 were designed to free the inquirers and indexers, respectively, from the rigidity of unweighted descriptors. Model 7 combines these individual solutions into one system by permitting both the inquirer and indexer to weight request or indexing descriptors, and then uses both weights to determine retrieved document rankings. (Care must be taken to provide both the indexers and inquirers with the same weighting scale.) The same advantages and disadvantages for Models 5 and 6 also apply, in aggregate, to Model 7. An additional advantage for Model 7 is that subtler gradations of *both* the inquirer's and indexer's points of view are accommodated by the retrieved set ranking algorithm. The principal disadvantage of Model 7 is that because both request term weights and index term weights must be used to rank output, the total processing time needed for ordering a large retrieved set may be substantial.

Summarizing Model 7:

ADVANTAGE: Ranking of retrieved documents accommodates both inquirer's and indexer's points of view.

DISADVANTAGE: Increase in processing to carry out ranking of large retrieved sets.

One of the questions which comes up about models that have both weighted index terms and request terms is how to combine these two kinds of weights to produce a single ranking of retrieved documents. Model 7 proposed the simplest procedure for combining the term and request weights--simply multiply them together. But much work has been done to find procedures for combining the weights to produce a ranking of retrieved documents which is more like the ranking of importance which the inquirer would give these same documents. One suggestion which, though proposed some time ago, has remained

---

MODEL 7 (WEIGHTED REQUESTS AND INDEXING)

I. REQUEST: SET OF DESCRIPTORS EACH OF WHICH HAS A POSITIVE NUMBER ASSOCIATED WITH IT.

II. INDEXING ASSIGNMENT: SET OF DESCRIPTORS EACH OF WHICH HAS A POSITIVE NUMBER ASSIGNED TO IT.

III. RETRIEVED DOCUMENTS ARE RANKED.

IV. RETRIEVAL RULE: DOCUMENTS ARE RANKED BY THE SUM OF PRODUCTS EACH OF WHICH RESULTS FROM THE MULTIPLICATION OF THE WEIGHT OF THE DESCRIPTOR IN THE REQUEST BY THE WEIGHT OF THE SAME DESCRIPTOR IN THE INDEX RECORD.

e.g.: REQUEST = D_k (7), D_j (2), D_m (4)

RANKING			STATUS VALUE
LAST	$DOC_a = < D_a (7), D_b (3) >$	=	0
2nd	$DOC_b = < D_a (4), D_k (6), D_o (1) >$	=	42
1st	$DOC_c = < D_a (1), D_k (4), D_o (6) >$	=	28
3rd	$DOC_d = < D_a (3), D_k (1), D_j (2) >$	=	11

---

popular, is to treat the request and the document descriptors as vectors in an n-dimensional space. The ranking of the retrieved documents is determined by calculating the Cosine of the angle between the search request vector and the vectors of documents in the data base. Intuitively, the smaller the angle between the search request vector and a given document vector the more similar the direction of their vectors are, and, hence, document decriptors match the search request. The Cosine function is also nice from a scaling point of view since the ranking values range from 0 to 1.

One of the most beneficial results of the vector space model of information retrieval is that it provides one of the essential constituents of any scientific theory--an analogy, or conceptual model. (The nature of scientific theory and its relation to information retrieval will be discussed in greater detail in Chapter 6.) Such a conceptual model can serve as a basis for the extension and development of the basic theory. This is precisely what has happened with the vector space interpretation of information retrieval. An entire industry of research and development has grown out of this model beginning with the SMART project [7] and continuing to the present.[8] This is all healthy and good for stimulating research, but the disadvantage of such a conceptual model is that it locks researchers into a particular point of view which may prevent them from addressing important questions of research--rather like the old story of a man who lost his keys on a dark street at night but spent his time looking for them on another street because, he said, "the light is better." I am not, of course, saying that the vector space model provides a myopic view of information retrieval problems (on the contrary, it has stimulated a great deal of interesting research), only that like other conceptual models it may preclude certain lines of research [9].

Summarizing Model 8:

ADVANTAGE: Provides a conceptual model to interpret and extend information retrieval models.

DISADVANTAGE: As a conceptual model it may inadvertently preclude certain lines of research, or suggest correlations which do not exist in reality.

Model 9, the Boolean search model, is probably the most popular retrieval design for computerized document retrieval systems, being used for, inter alia, the DIALOG, ORBIT and STAIRS retrieval systems.

MODEL 8 (COSINE RULE)

I. REQUEST: SET OF DESCRIPTORS EACH OF WHICH HAS A POSITIVE NUMBER ASSOCIATED WITH IT.

II. INDEXING ASSIGNMENT: SET OF DESCRIPTORS EACH OF WHICH HAS A POSITIVE NUMBER ASSIGNED TO IT.

III. RETRIEVED DOCUMENTS ARE RANKED.

IV. RETRIEVAL RULE: THE WEIGHTS OF THE DESCRIPTORS COMMON TO THE REQUEST AND AN INDEXING RECORD ARE TREATED AS VECTORS. THE VALUE OF A RETRIEVED DOCUMENT IS THE COSINE OF THE ANGLE BETWEEN THE VECTORS:

$$\text{VALUE} = \frac{R_1 \cdot I_1 + \ldots + R_n \cdot I_n}{[(R_1^2 + \ldots + R_n^2)(I_1^2 + \ldots + I_n^2)]^{1/2}}$$

WHERE R_1 = WEIGHT OF REQUEST TERM 1. I_1 = WEIGHT OF INDEX TERM 1

e.g.: REQUEST = D_k (8), D_j (2), D_m (4)

RANKING			STATUS VALUE
LAST	$DOC_a = < D_a (8), D_b (3) >$	=	0
1st	$DOC_b = < D_a (4), D_k (8), D_o (1) >$	=	.78
2nd	$DOC_c = < D_a (1), D_k (4), D_o (6) >$	=	.52
3rd	$DOC_d = < D_a (3), D_k (1), D_j (2) >$	=	.35

MODEL 9 (BOOLEAN REQUESTS)

I. REQUESTS ARE ANY BOOLEAN COMBINATION OF DESCRIPTORS OF THE FOLLOWING FORMS:

$D_a \cdot D_b$ (INTERSECTION)

D_a v D_b (UNION)

$-D_a$ (NEGATION)

II. INDEXING ASSIGNMENTS:
A SET OF ONE OR MORE DESCRIPTORS.

III. DOCUMENTS ARE EITHER RETRIEVED OR NOT.

IV. RETRIEVAL RULES:

--IF REQUEST = $D_a \cdot D_b$ THEN RETRIEVE ONLY DOCUMENTS WITH *BOTH* D_a AND D_b.

--IF REQUEST = D_a v D_b THEN RETRIEVE ONLY DOCUMENTS WITH *EITHER* D_a or D_b. (non-exclusive "or")

--IF REQUEST = $-D_a$ THEN RETRIEVE ALL DOCUMENTS *WITHOUT* D_a.

Because of the popularity of the Boolean system, designers of new, or more advanced retrieval systems often take Model 9 as the basic interface which they can enhance by adding term weighting schemes, associative searching procedures, feedback techniques, etc. Nevertheless, in spite of the popularity of Model 9, both as a system by itself and as the basis for more advanced systems, there are objections to its use, namely, that the typical inquirer lacks the understanding of Boolean (or propositional) logic necessary to compose anything but the simplest search request. For example, they may be able to compose a formal request like:

(1) $(D_p \cdot D_q)$ v D_r

but be at a loss to understand or compose a formal request like:

(2) $(D_p \cdot D_q)$ v $(D_q \cdot D_r)$ v $(D_p \cdot D_r)$ v D_r

The irony is that requests (1) and (2) are, from a logical point of view, equivalent. For the inquirer who remembers or understands the laws of propositional logic, Model 9 can be a very useful request structure. The rules of replacement such as (inter alia):

$(D_p \cdot D_q) = -(-D_p \text{ v } -D_q)$ (DeMorgan's Theorems)

$(D_p \text{ v } D_q) = -(-D_p \cdot -D_q)$

$D_p = (D_p \text{ v } D_p) = (D_p \cdot D_p)$ (Tautology)

$D_p = (D_p \text{ v } (D_p \cdot D_q))$

$D_p = (D_p \cdot (D_p \text{ v } D_q))$

can be quite useful in composing and simplifying search requests.[10]

Summarizing Model 9:

ADVANTAGES: Popular and widely used request interface; enables the inquirer to construct formal requests comprising complex combinations of search terms; the laws of propositional logic can be used to simplify complex query formulations.

DISADVANTAGES: Being the most popular retrieval model implies, erroneously, that by itself it is the best model; the Boolean combinations of search terms can easily become too complex for the typical inquirer, and there is no easy technique for simplifying Boolean expansions.

Model 10 offers an increasingly popular enhancement of the basic Boolean retrieval system. Instead of having the retrievable documents represented by a set of subject and context (e.g., author, title, date, etc.) descriptors, the entire text of stored documents is searchable, if desired. In effect, the retrievable documents are represented by the aggregate of "content-bearing" words they contain, and retrieval is effected by having the inquirer attempt to predict the words and phrases

which might occur in documents he would hope would be useful to him. Systems like IBM's popular STAIRS--*ST*orage *A*nd *I*nformation *R*etrieval *S*ystem--permit the inquirer to construct search requests which are variations on the traditional Boolean intersection of search terms:

D_a and D_b:	Both terms must appear in the same document
D_a adjacent D_b:	Both terms must appear next to each other in the text (e.g., "San" adjacent "Francisco")
D_a with D_b:	Both terms must appear in the same paragraph in a document
D_a same D_b:	Both terms must appears in the same sentence in a document

Full-text information retrieval systems, such as STAIRS, GESCAN, etc., are more dependent on cheap, fast, large-scale computer power than the previous models. And although the cost of computer power and storage continues to go down dramatically, full-text retrieval systems, in general, still cost more per retrieval than most other retrieval models. Many systems designers are willing to endure the higher processing and storage costs of full-text retrieval because it frees them from the biggest design headache of information retrieval--indexing. By storing the entire text of a document on line and allowing it to be searchable, indexing (whether manual or automatic), it is argued, is not necessary. The underlying assumption here is that inquirers are able to retrieve useful documents in small retrieved sets by trying to anticipate the words and phrases that appear in those documents.

The designers of information retrieval systems have three major objections to proposing systems which require manual (or intellectual) indexing. In the first, and most obvious, place there are enormous conceptual difficulties with indexing (e.g., how do you accurately or reliability describe the subject or content of a document?). This is a major area of concern in information retrieval, but from a practical point of view it's always possible to hire individuals to index documents, though some will be better at this skill than others. The second major concern which system designers have about manual indexing is an organizational concern. Namely, how do you manage and control all the indexers you might need to handle input to a realistically large system? Most organizations which have a need for large-scale document retrieval have no place in their organizational hierarchy for a large group of indexers. The data processing side of the organization feels uncomfortable managing indexers because their skills are not at all like

---

MODEL 10 (FULL TEXT RETRIEVAL)

I. REQUESTS ARE ANY BOOLEAN COMBINATION OF DESCRIPTORS

$D_a \cdot D_b$	(INTERSECTION)
$D_a \text{ v } D_b$	(UNION)
$-D_a$	(NEGATION)

II. INDEXING ASSIGNMENTS:

ENTIRE TEXT OF THE DOCUMENTS IS SEARCHABLE (excluding non-content "stop" words such as "and", "of", "the", etc.).

III. DOCUMENTS ARE RETRIEVED OR NOT.

IV. RETRIEVAL RULES:

--IF REQUEST	=	$D_a \cdot D_b$ THEN RETRIEVE ONLY DOCUMENT WITH *BOTH* D_a AND D_b.
--IF REQUEST	=	D_a v D_b THEN RETRIEVE ONLY DOCUMENT WITH *EITHER* D_a or D_b
--IF REQUEST	=	$-D_a$ THEN RETRIEVE ALL DOCUMENTS *WITHOUT* D_a.
--IF REQUEST	=	D_a ADJACENT D_b THEN RETRIEVE ALL DOCUMENTS WITH D_a OCCURRING NEXT TO D_b IN THE TEXT.
--IF REQUEST	=	D_a WITH D_b THEN RETRIEVE ALL DOCUMENTS WITH D_a OCCURRING IN THE SAME PARAGRAPH AS D_b.
--IF REQUEST	=	D_a SAME D_b THEN RETRIEVE ALL DOCUMENTS WITH D_a OCCURRING IN THE SAME SENTENCE AS D_b.

---

typical data processing skills, and the management side of the organization feels equally uncomfortable supervising the indexers since their work is clearly a support activity, not a traditional managerial function. As a result, there is no clear organizational position for indexers to exist in most commercial environments. This, more than the conceptual problems of indexing, has made simple full-text retrieval systems (like STAIRS and GESCAN) appealing in spite of the greater retrieval costs. (Automatic indexing procedures which are based on full-text data bases or full text systems with more advanced retrieval capability (such as Verity's TOPIC) have not, as yet, seriously challenged the market share of the simple Model 10 systems[11].)

A final objection to manual indexing is its cost. Hiring indexers has been likened to hiring expensive, highly educated, slow keypunchers. By opting for manual indexing a system designer may feel like he or she is creating an input bottleneck and also paying a premium for it. Full-text retrieval does save you the cost of manual indexing, but its overall cost will be cheaper than a more selectively indexed system only if the documents to be inputed do not have to be converted to machine-readable form. If the documents are not in machine-readable form, the added cost of inputing and verifying the full-text of the documents (as opposed to just index terms in manual indexing) may exceed the cost (and time) of manual indexing.

While cost and organizational issues are important considerations for the design of information retrieval systems, the most important factor in selecting a retrieval design is whether or not it offers inquirers the levels of retrieval effectiveness they need. Simple full-text retrieval systems suffer from certain disadvantages that the previous models do not have to the same degree. When an inquirer constructs a search query to use to retrieve documents on a full-text retrieval system, he must select terms which he anticipates will not only appear in documents which he wants the system to retrieve, but will *not appear* in documents which he *does not* want the system to retrieve. Most inquirers tend to think that their retrieval strategy is to predict only the words and phrases which appear in the documents they want, and are not often sufficiently sensitive to the fact that the selected terms, if possible, should not appear in documents which the inquirer does not want (otherwise, these undesirable documents will be retrieved, too). Simple Full-text retrieval systems appeal to this naivete by offering the inquirer the capability for retrieving documents by specifying any of the "content-bearing" words or phrases which might occur in a document, or documents, he wants. Since the entire content of documents can be stored on a full-text system, it is a relatively easy task to guess at least some of the words which are likely to occur in those documents which he would like to retrieve. Unfortunately, because all the "content-bearing" words of the documents on the data base are available for the inquirer to

use, almost any term which occurs in a document which the inquirer wants will most likely occur also in the text of documents which he doesn't want. Consider a simple example: On an information retrieval system which contains mostly documents on the subject of Computer Science, one would expect that a desired document would contain the word "computers" in it. Unfortunately, *most* of the documents on the data base would also contain the word "computers" in it, and it is unlikely that the inquirer would want all of those documents to be retrieved. In the limiting case, even a document with the phrase "...this article is *not* about computers..." would be retrieved by a search query specifying the retrieval of documents with the word "computers" in them. (Of course, there are many proposals for mitigating this problem by, for example, retrieving documents where the term "computers" occurs more than its average occurrence rate in all other documents. But we are only discussing the simple model here without any of these refinements built in.)

The extraordinary richness and flexibility of natural language make it extremely difficult for an inquirer to predict the precise words and phrases which would occur in the text of documents which he might want, but would not occur in the text of documents he would find irrelevant.

Another problem with simple full-text retrieval systems results from the fact that the entire text of retrievable documents is available for searching. A conventional document retrieval system might represent the intellectual content of a document with about 20 terms, while a simple full-text system will, on average, use half the words in the text to represent the document.[12] This means that a full-text retrieval system might use hundreds or even thousands of times more information to represent a given set of documents than a more conventional system. This increase in information manifests itself in two major ways: a single search term used on a full-text system will likely retrieve significantly more documents than the same term used to retrieve documents from a more conventional system with more selective indexing (assuming the same data base for both systems); and, the number of different terms in the search (or index) vocabulary of a full-text system will be significantly larger than the number of different terms in the vocabulary of a more conventional system. The major consequence of full-text search terms retrieving more documents than terms on a conventional system is that they will also retrieve more non-relevant or useless documents per search than a conventional system. To put it simply, since full-text systems permit searching on every occurrence of a given word or phrase in a document, it stands to reason that some, and perhaps many, of those occurrences may be trivial or misleading uses of those words or phrases. (e.g., an inquirer searching on a full-text system for documents about "distributed computer

systems" might find his retrieved sets of documents cluttered with articles on "computer systems distributed by Hewlett Packard...", or "...production control computer systems which allocate how work is to be distributed...", etc. Searching with the phrase "distributed *adjacent* computer *adjacent* systems" is too specific since it might exclude such relevant phrases as "...computer systems in which processing is distributed...")

The other consequence of full-text retrieval is that many more different terms will be available in the search vocabulary of a full-text system than the vocabulary of a more conventional system (this is known as an "uncontrolled vocabulary"). Such an increase in the number of available search terms dramatically increases the number of different search queries which could be formulated on a full-text system, as opposed to a conventional system. This increase in the number of possible search queries is a *combinatoric* increase which can easily provide the inquirer with more search query alternatives than he knows what to do with (the inquirer needs only a few search term alternatives, not several hundred). These combinatorial problems were discussed briefly in Model 3 and will also be discussed in Model 11.

Another problem with retrieving documents by anticipating which words and phrases were used to discuss a topic is that often the information necessaryfor retrieval is not contained in the text of the document. Documents often have implicit links between them in that they discuss the same issue, respond to a document making a request, make a commitment, provide information about a topic or activity, make a judgment or evaluation of another individual's proposal or statement, etc. For example, in the study of the full-text retrieval system for litigation support described in Chapter 3, it was frequently necessary to establish evidence for "who knew *what* about the litigated issue, and *when* did he know it," or, "did anyone object to X's proposal?" To search for documentary evidence of this type, one would have to identify every possible author of this type of document *and* describe every possible way in which such topics could be discussed, an impossible task using full-text retrieval on a document collection of even modest size. In addition, documents were found that were germane to the lawsuit but authored by individuals who were not employees of the company engaged in the lawsuit. It was very difficult for the lawyers (who were searching the document collection) to identify and find the documents that these individuals authored. There is growing evidence that some designers of information systems recognize the importance of these implicit document links and have designed systems which force the authors of documents to establish links between documents which are not explicitly described in their text. Examples are the COORDINATOR, the LENS system (Malone, et.al.) and Filenet's WORKFLO program[13].

Model 10, it must be emphasized, is a *simple* full-text retrieval system. There are full-text retrieval systems which include various more or less sophisticated enhancements designed to improve the quality of retrieval, but it is not clear how much improvement they would offer or whether they would compensate at all for the serious problems of the simple full-text systems. One recent development in this area is the construction of a full-text retrieval system based on the "Connectionist" principles of system design. [14] The major problem of simple full-text systems is that they overload the inquirer with searching alternatives by representing the documents on the data base with an excessive number of words and phrases. The new connectionist full-text retrieval system offers an interesting way in which to handle this combinatorial increase in complexity - a process not practical for computers with the traditional Von Neumann architecture (as all the commercial full-text retrieval systems are).

Summarizing Model 10:

> ADVANTAGES: No indexing (either manual or automatic) required to represent documents on the retrieval system.
>
> DISADVANTAGES: Using full-text to represent documents leads to: larger retrieved sets than conventional systems (ie, higher average "hit" rates for terms in full-text systems when compared to the same terms in a more selectively indexed system); excessively large numbers of search term alternatives and more spurious occurrences of terms; significantly larger data base than conventional systems; larger directory; higher data entry costs (if text of documents is not already machine readable); poor retrieval if important retrieval information is not contained in the text, or if the inquirer cannot anticipate how a desired document discusses a topic.

One of the major obstacles which the inquirer faces is the fact that he may find it difficult to think of all the possible search terms which are likely to have been assigned to documents which he would find useful. It is a well known fact that man's powers of recall are much less robust than his powers of recognition. He can easily recognize descriptors which are semantically related, when they are presented to him, but he may have trouble recalling these same terms without prompting. The addition of a thesaurus to an information retrieval system can be useful in providing an inquirer with additional search terms which are semantically related to the terms with which he starts his search (i.e., his "anchor set", vid. Chapter 1). The method of

operation of the thesaurus could be either "active" or "passive". The active thesaurus operates by automatically adding semantically related terms. For example, the search term (phrase) "data base management system(s)" might become:

([data base management system(s)] or [data base(s)] or [DBMS] or

[data base systems(s)])

Normally, such an addition of semantically related search terms would not be done automatically, but would be an option which could be selected by the inquirer if his initial anchor set (q.v) of search terms proves insufficient for retrieving documents from the data base. The automatic addition of all descriptors semantically related to the descriptors in the inquirer's original anchor set can sometimes cause more problems than it solves. Specifically, unless the thesaurus is extremely sparse, there will often be an unmanageable number of terms semantically related to those in the anchor set. Each of these terms when added to the formal query will be likely to cause additional documents to be retrieved from the data base. If quite a few semantically related search terms are added to the inquirer's anchor set it could have the effect of raising the number of retrieved documents beyond the inquirer's "futility point" (vid. Chapter 1). In short, merely adding semantically related search terms to the inquirer's original search query does not necessarily improve the inquirer's searching prospects.

An observer might counter that the number of terms semantically related to a given term must necessarily be limited, otherwise our language would have long ago given over to semantic anarchy. Our language clearly is not anarchic, but, short of this, it is extremely creative and varied, and we have a virtually unlimited number of ways of talking about or describing the same subject. As a result, it is a practical impossibility to describe all the words and phrases in natural language which might be used to discuss a particular topic. This non-deterministic nature of language (discussed more extensively in Chapter 4) has proved to be the undoing of many natural language projects in artificial intelligence and Computational Linguistics.

Clearly, then, a thesaurus would only be a useful aid to the searching process if the addition of terms semantically related to the inquirer's anchor set is done selectively. The inquirer needs not just semantically related search terms, but semantically related terms which in his estimation, represent his retrieval needs *appropriately* (that is, satisfy the "prediction criterion" (q.v. Chapter 1) of his search).

Model 11 (SIMPLE THESAURUS)

I. REQUESTS ARE SINGLE DESCRIPTORS.

II. INDEX ASSIGNMENTS: A SET OF ONE OR MORE DESCRIPTORS.

III. DOCUMENTS ARE EITHER RETRIEVED OR NOT.

IV. RETRIEVAL RULE: THE REQUEST TERM IS LOOKED UP IN A THESAURUS (ON-LINE) AND SEMANTICALLY RELATED TERMS ARE ADDED TO THE REQUEST TERM. RELATED TERMS ARE ADDED DISJUNCTIVELY.

TYPICAL THESAURUS (BINARY)

	T_A	T_B	T_C	T_D	T_E	.	.	.	T_N
T_A	*	0	0	1	0	.	.	.	0
T_B		*	1	0	1	.	.	.	0
T_C			*	1	0	.	.	.	0
T_D				*	0	.	.	.	0
T_E					*	.	.	.	0
.									.
.									.
.									.
T_N									*

One of the easiest ways in which to make the addition of related terms selective is to make the thesaurus function *passively*. That is, terms which are semantically related to the inquirer's anchor set are not automatically added to the tentative search query. Instead, the thesaurus will only display terms which are semantically related to individual terms which the inquirer specifically suggests. The inquirer is then free to select from all the semantically related terms presented by the thesaurus, those terms which he believes will be likely to prove useful as additions to his original formal query. As such, the thesaurus serves mostly as a reminder for the inquirer. Implicit in the notion of a passive thesaurus is that the inquirer is experienced enough to recognize additional good search terms when they are presented clearly.

Of course, the major issue behind the development of a thesaurus as an ancillary to the information retrieval process is how to derive the semantic relationships on which the thesaurus is built. Basically, these semantic relationships can be either *normative* or *descriptive*. Normative relationships are regulative or ideal and attain their status by fiat or agreement. Implicit in a normative nomenclature is the fact that the relationships could be stipulated in more than one way, thus making it necessary to choose one way of arranging things over another. The first example to come to mind of a normative nomenclature is, of course, the classifications of the Life Sciences: Botany, Biology, etc. In a hypothetical thesaurus we would place the description "whale" in a subordinate ("narrower than") relation to the description "mammal", if we wanted to remain faithful to the established Biological classification. But "whale" could just as easily be placed in a subordinate relation to "sea creatures", "swimming animals", "large animals", "herding animals", or "creatures which make plaintive sounds"; and this, by no means, exhausts the possible relations in which the term "whale" can participate. The reason why Biology stipulates that "whale" is subordinately related to "mammal" is that the primary *use* which Biologists have for the word "whale" is in its subordinate relation to the family of "mammals" and not the families of "swimming animals", etc. (We will have a great deal more to say about the primacy of the *use* of language in Chapter 4.) Normative classifications, in effect, impose a particular point of view on a system of relationships between terms or the classes of objects to which they refer. Such a regulative system in information retrieval runs the danger of being Procrustean whenever the established nomenclature relates terms differently from the way in which they are used by inquirers. This problem is more in evidence when we consider the principal normative classification systems for information retrieval: the Dewey Decimal System and the Library of Congress Classification System. Both systems embody ideals, and, as such, do not represent relationships between subject descriptions in the same manner that a typical information retrieval system might. (Researchers such as Tague[15] have shown that the term relations

elicited by association procedures tend to be different from those relations described in a traditional thesaurus.)

This dissonance between the term relationships stipulated in a thesaurus based on a normative nomenclature, and the manner in which these term relationships are perceived and used by inquirers or indexers leads one to the conclusion that a thesaurus useful in information retrieval must be *descriptively* based rather than normative. To be descriptively based, a thesaurus must derive the term relationships from the data base of document representations (either assigned index terms or full text). The most obvious method by which to derive semantic relationships between document descriptions is to tabulate the inter-term frequencies of co-occurrence. That is, to keep track of the number of times any two terms are used together to represent the content of a single document. The underlying assumption is, naturally, that if two descriptors are used frequently together to represent documents for retrieval, then these two descriptors are semantically related. 16 The co-occurence relation can also be interpreted probabilistically, where the co-occurence of term A with term B can be interpreted as the conditional probability of term A, given term B or vice versa:

$$P(Term_A/Term_B)$$

In model 11 we have described only a single binary thesaurus, so the values of the conditional probability of co-occurence can be "0" or "1". An assignment of "1" could be given if the conditional probability of one term given another rises above a specified cut-off value.

One interesting advantage of a descriptive thesaurus based on co-occurrence data is that the co-occurrence data need not be limited just to subject terms. For example, data could be collected on how frequently certain subject descriptors co-occur with particular author's names, journal names, or titles of institutions. In this way, a thesaurus could be used to gather a subject profile for certain authors, institutions, etc.

While a thesaurus, in its simplest form, expresses the semantic relationships between terms as primitive, uninterpreted and symmetric ($Term_a$ being related to $Term_b$ implies that $Term_b$ is related to $Term_a$), the tradition of normative thesuari includes processes for describing classes of relations:

$Term_a$ narrower than $Term_b$

$Term_a$ broader than $Term_b$

One of the easiest ways in which to make the addition of related terms selective is to make the thesaurus function *passively*. That is, terms which are semantically related to the inquirer's anchor set are not automatically added to the tentative search query. Instead, the thesaurus will only display terms which are semantically related to individual terms which the inquirer specifically suggests. The inquirer is then free to select from all the semantically related terms presented by the thesaurus, those terms which he believes will be likely to prove useful as additions to his original formal query. As such, the thesaurus serves mostly as a reminder for the inquirer. Implicit in the notion of a passive thesaurus is that the inquirer is experienced enough to recognize additional good search terms when they are presented clearly.

Of course, the major issue behind the development of a thesaurus as an ancillary to the information retrieval process is how to derive the semantic relationships on which the thesaurus is built. Basically, these semantic relationships can be either *normative* or *descriptive*. Normative relationships are regulative or ideal and attain their status by fiat or agreement. Implicit in a normative nomenclature is the fact that the relationships could be stipulated in more than one way, thus making it necessary to choose one way of arranging things over another. The first example to come to mind of a normative nomenclature is, of course, the classifications of the Life Sciences: Botany, Biology, etc. In a hypothetical thesaurus we would place the description "whale" in a subordinate ("narrower than") relation to the description "mammal", if we wanted to remain faithful to the established Biological classification. But "whale" could just as easily be placed in a subordinate relation to "sea creatures", "swimming animals", "large animals", "herding animals", or "creatures which make plaintive sounds"; and this, by no means, exhausts the possible relations in which the term "whale" can participate. The reason why Biology stipulates that "whale" is subordinately related to "mammal" is that the primary *use* which Biologists have for the word "whale" is in its subordinate relation to the family of "mammals" and not the families of "swimming animals", etc. (We will have a great deal more to say about the primacy of the *use* of language in Chapter 4.) Normative classifications, in effect, impose a particular point of view on a system of relationships between terms or the classes of objects to which they refer. Such a regulative system in information retrieval runs the danger of being Procrustean whenever the established nomenclature relates terms differently from the way in which they are used by inquirers. This problem is more in evidence when we consider the principal normative classification systems for information retrieval: the Dewey Decimal System and the Library of Congress Classification System. Both systems embody ideals, and, as such, do not represent relationships between subject descriptions in the same manner that a typical information retrieval system might. (Researchers such as Tague[15] have shown that the term relations

elicited by association procedures tend to be different from those relations described in a traditional thesaurus.)

This dissonance between the term relationships stipulated in a thesaurus based on a normative nomenclature, and the manner in which these term relationships are perceived and used by inquirers or indexers leads one to the conclusion that a thesaurus useful in information retrieval must be *descriptively* based rather than normative. To be descriptively based, a thesaurus must derive the term relationships from the data base of document representations (either assigned index terms or full text). The most obvious method by which to derive semantic relationships between document descriptions is to tabulate the inter-term frequencies of co-occurrence. That is, to keep track of the number of times any two terms are used together to represent the content of a single document. The underlying assumption is, naturally, that if two descriptors are used frequently together to represent documents for retrieval, then these two descriptors are semantically related. 16 The co-occurence relation can also be interpreted probabilistically, where the co-occurence of term A with term B can be interpreted as the conditional probability of term A, given term B or vice versa:

$$P(Term_A/Term_B)$$

In model 11 we have described only a single binary thesaurus, so the values of the conditional probability of co-occurence can be "0" or "1". An assignment of "1" could be given if the conditional probability of one term given another rises above a specified cut-off value.

One interesting advantage of a descriptive thesaurus based on co-occurrence data is that the co-occurrence data need not be limited just to subject terms. For example, data could be collected on how frequently certain subject descriptors co-occur with particular author's names, journal names, or titles of institutions. In this way, a thesaurus could be used to gather a subject profile for certain authors, institutions, etc.

While a thesaurus, in its simplest form, expresses the semantic relationships between terms as primitive, uninterpreted and symmetric ($Term_a$ being related to $Term_b$ implies that $Term_b$ is related to $Term_a$), the tradition of normative thesuari includes processes for describing classes of relations:

$Term_a$ narrower than $Term_b$

$Term_a$ broader than $Term_b$

$Term_a$ related to $Term_b$

$Term_a$ used for ("instead of") $Term_b$ [17]

Natural hierarchies can, of course, be inferred by looking at strings of narrower than and broader than relations. For example:

$Term_a$ narrower than $Term_b$

$Term_b$ narrower than $Term_c$

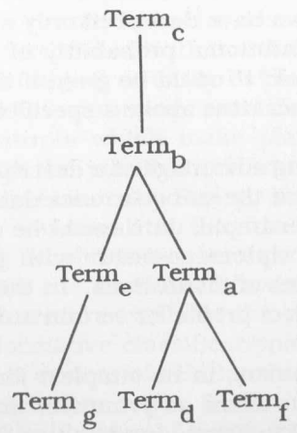
$Term_d$ narrower than $Term_a$

$Term_b$ broader than $Term_e$

$Term_a$ broader than $Term_f$

$Term_g$ narrower than $Term_e$

implies the following hierarchy of terms:



Although such classes of thesaurus relations are typically associated with normative thesauri, these distinctions can also be distilled from a descriptive thesaurus based on co-occurrence data. Using a probabilistic interpretation of co-occurence, we can define the traditional classes of relations as follows:

$Term_a$ is narrower than $Term_b$ iff:

$P(Term_b/Term_a) = 1$, and
$P(Term_b) > P(Term_a)$

$Term_a$ is broader than $Term_b$ iff:

$P(Term_a/Term_b) = 1$, and
$P(Term_a) > P(Term_b)$

$Term_a$ may be used for $Term_b$ (or vice versa) iff:

$P(Term_a/Term_b) \times P(Term_b/Term_a) = 1$

Normally, though, because of the indeterminacy inherent in indexing,[18] we would establish cutoff values for these probabilistic interpretations which would be close to 1.0 rather than precisely 1.0.

One of the interesting properties of many semantic relations is that they are transitive. This means that we not only have the relations which occur explicitly in the co-occurence data, but we can infer the existence of other relations (which are not explicit) by assuming transitivity. For example, if we find that $Term_a$ co-occurs with $Term_b$ and $Term_b$ co-occurs with $Term_c$, we can infer, using transitivity, that there exists a semantic relationship between $Term_a$ and $Term_c$, in spite of the fact that they do not co-occur. This distinction between relations which are explicitly represented in the data base, and relations which can be inferred from this explicit information, closely parallels the well-known logical distinction between the intension and extension of a term, symbol or proposition. The extension of a term is the class of entities to which it applies, while the intension of a term is the collection of properties which defines it. As Allwood, et. al.,[19], put it:

> An intention is something that relates a linguistic expression [term] to its extension. It determines the extension of a linguistic expression...an intension is a function: something that for every possible situation or world picks out exactly those objects which make up the extension of a given expression...

Maron[20] was the first to see this parallel between the intension and extension of logical expressions and the intension and extension of relations. He offered a somewhat restricted version of this classical distinction:

> The *intensional* meaning of a relation R is the interpretation of R in terms of other relations as, for example, when we say that the relation father-in-law which holds between a man and another individual means that the man's child is the spouse of the other individual in question. The *extensional* meaning of a binary relation R is the interpretation of R in terms of the sets of pairs for which R holds. In the case of the relation father-in-law, the extensional meaning would be the ordered sets of pairs of men and spouses for whom the relation father-in-law holds. Thus, we say that the extension of a relation is the set of ordered pairs for which the relation holds, and the intension of a relation is its definitional equivalent in terms of other relations.

Thus, according to Maron, the extensions of relations are those relations represented explicitly in the data base (here, the pairs of terms which co-occur), while the intension of a relation is a relation which must be inferred or derived from other relations (extensional or intensional).

In automated systems where the thesaurus is represented as a matrix, the relations which are not explicit in the data base can be determined fairly easily. This is known as the *transitive closure* of the original term matrix, and it is accomplished by applying Warshall's Algorithm: [21]

Given a term thesaurus represented as a matrix B:

1. Set a new matrix A = B
2. Set i := 1.
3. For all j, if A(j,i) = 1, then
   For k = 1,...,n set
   A(j,k) := A(j,k) + A(i,k).
4. Set i := i + 1.
5. If i = n then go to step 3; else stop.

To compute the transitive closure of any matrix will require that the assignment statement in step 3 be executed n^3 times, which makes the algorithm within the computational power of most systems. But these are not the only calculations which might be of interest here. Using Dijkstra's Algorithm [22] one could calculate the shortest path between any two terms which are transitively related. This could be

important for determining whether two terms are "significantly" related, under the assumption that $Term_a$ and $Term_b$ are more closely related semantically if you need only one other term to form a transitive string between them than if you need 4 or 5 other relations to get from $Term_a$ to $Term_b$. Also, the number of different paths between two terms can be calculated, with the possible interpretation that terms which have a large number of different paths between them may form, along with the terms in the mediating relations, a semantic "cluster" of related terms in which the two original terms are dominant.[23]

Summarizing Model 11:

> ADVANTAGES: Provides the inquirer with a list of terms which are semantically related to those in his Anchor Set (especially useful in systems with uncontrolled vocabularies); Provides a foundation for inferring term relationships which are not explicitly represented in the data base.
>
> DISADVANTAGES: (Normative thesaurus) Relationships described in the thesaurus may not accurately reflect the manner in which the terms are perceived to be related by the inquirers or the indexers; (Descriptive thesaurus) Associative techniques (such as co-occurence data) may produce a combinatorial growth in the number of term::term relations so that the inquirer may be overwhelmed with terms related to his anchor set, and thereby have difficulty satisfying his Prediction Criterion for searching.

Model 12 is the same model as 11, except that the relationships between terms in the thesaurus can be expressed in terms of degrees rather than the binary relations of Model 11. The relationships can, of course, be normative or descriptive just like the Model 11 (with the same advantages and drawbacks), but there are several advantages to being able to represent the degrees of semantic or statistical relatedness between terms. In the first place, weak relations can be represented in such a thesaurus. Such relations cannot be represented in a binary thesaurus since it might lead them to be interpreted as stronger than they really are (that is, as strong as other binary relations). The second advantage is that where a single term is related to several other terms the inquirer can readily see which of the several related terms are more strongly related to the first term than the others.

While there are many procedures for constructing a weighted thesaurus, just as there were many ways of constructing a binary thesaurus, a descriptively based thesaurus appears to have the same

MODEL 12 (WEIGHTED THESAURUS)

I. REQUESTS ARE SINGLE TERMS.

II. INDEX ASSIGNMENTS: A SET OF ONE OR MORE DESCRIPTORS.

III. DOCUMENTS ARE EITHER RETRIEVED OR NOT.

IV. RETRIEVAL RULE: THE REQUEST DESCRIPTOR IS LOOKED UP IN A THESAURUS (ON-LINE) AND SEMANTICALLY RELATED DESCRIPTORS ABOVE A GIVEN CUT-OFF VALUE (WEIGHT) ARE ADDED (DISJUNCTIVELY) TO THE REQUEST DESCRIPTOR. THE CUT-OFF VALUE COULD BE GIVEN BY THE INQUIRER.

advantages here as it did with Model 11. An obvious method for constructing such a descriptively based thesaurus is to use statistical co-occurrence data, where the degree to which one term is related to another is represented by the probability that one term co-occurs with another in representing a single document. *Viz.*,

$$P(I_j/I_k) = \frac{\text{No. of times } I_j \text{ co-occurs with } I_k}{\text{No. of times } I_k \text{ occurs}}$$

Note, of course, that the probability of I_j co-occurring with I_k is not the same as the probability of I_k co-occurring with I_j.

$$P(I_j/I_k) \neq (I_k/I_j)$$

It has been suggested that such co-occurrence data could be used to provide an "extensional" definition of a topic on an information retrieval system, [24] where a list of terms which co-occur with a given term and ranked according to their decreasing probability of co-occurrence provides a rough statistical profile of how the given term is used on the retrieval system. Such extensionally defined topics can be useful in two ways: for the inquirer, it provides a ranked list of terms statistically related to his potential search terms from which he can select additional search terms; and, for the indexer, it provides a list of suggested index terms that might be included with a particular selected term to index a given document (the "suggested" terms are those which

have already co-occurred with the selected terms, so they represent an explicit indexing policy that already exists on the retrieval system. Certainly an ordered list of co-occurring terms is easier to read than an unordered row in a term::term matrix. A system which provides such lists of terms related to a given term, ranked by the decreasing probability of co-occurrence has been described by Jacquesson.[25]

The chief drawback of such a system would be the processing time needed to maintain and continually update the topic lists for every unique index term on the retrieval system. The topic lists for every term assigned to a newly acquired document would have to be updated when that document (or document citation) is entered into the data base. That is, if a new document is assigned 8 index terms to describe its intellectual content, the 8 topic lists for these 8 index terms must all be updated. The record keeping for a system such as Model 11, with a thesaurus of binary relations, is less extensive. It needs to keep track only of those co-occurrence frequencies which fall below the cutoff for inclusion in the thesaurus. Once two terms co-occur enough times to be included in the thesaurus, they are entered into the thesaurus and need not be updated (this assumes, of course, that new documents which are indexed for the data base would not decrease the probability of co-occurrence of any two terms below the cutoff probability for inclusion in the thesuarus).

Finally, the weights could be left on the added search terms enabling Model 12 to operate like a Model 5 (weighted requests) system. The only difference is that in Model 12 the search terms selected by the inquirer will not have weights, while the terms added by the thesaurus will have them.

Summarizing Model 12:

ADVANTAGES: Same as for Model 11. In addition, by permitting degrees of semantic relatedness to be represented, the inquirer or indexer can easily see which terms are more related to a given term than others; terms with relatively low degrees of relatedness can still be represented in the thesaurus.

DISADVANTAGES: Same as for Model 11. In addition, the processing necessary to maintain and update the changing relations between all co-occurring terms in the retrieval system is significantly more than the processing necessary to maintain a binary thesaurus.

HYBRID MODELS

Hybrid models are those which combine techniques of any number of the previous twelve models into one system. For example, an information retrieval system could be constructed which contained the following features:

Weighted indexing
Weighted search terms
Boolean requests
Cut-off value
Weighted thesaurus
Full text searching on title and author

The advantages and disadvantages of each procedure would have to be weighed in aggregate for any hybrid model, though, naturally, such advantages and disadvantages are not necessarily additive. In fact, as more features are included in an information retrieval system, the more difficult it may be to determine how much improvement it would offer over a simpler system. One clear disadvantage of such hybrid models is that the complexity of the retrieval system may increase significantly and, as a result, provide an inquirer with more information than he or she knows how to use effectively.

CONCLUSION

The obvious question which arises after this discussion of many of the formal models of information retrieval is, which model is best? Naturally, any recommendation must take into consideration the size of the document data base (a small data base might be better off using a simple design), the required level of retrieval effectiveness (high or low recall), or the sophistication of the inquirers who would use the system (a complex system might be counter productive if the inquirers are naive). But, with those considerations aside, it is conceptually possible to imagine each model being tested on the same data base of documents and processing the same set of search queries. In such a hypothetical situation we might imagine being able to get some hard evidence as to which models are better than others at retrieving "all and only the relevant documents". Why this hasn't come about will be the focus of Chapter 3.

Chapter 2: Notes

1. William Cooper was the first, to my knowledge, to represent information retrieval models this way. For a formal description of this framework, see Bookstein and Cooper's "A General Mathematical Model for Information Retrieval Systems." Library Quarterly, v. 46:2, April 1976, pp. 153-167.

2. D.C. Blair, and M.E. Maron. "An evaluation of retrieval effectiveness for a full-text document retrieval system." Communications of the ACM, v. 28:3 (March 1985), pp 289-297. Output overload will be discussed in more detail in Chapter 3.

3. The probabilistic interpretation of index term weights was originated by M.E. Maron and J.L. Kuhn in their, "On Relevance, Probabilistic Indexing and Information Retrieval", Journal of the Association for Computing Machinery , v. 7:3, July 1960, pp. 216-244. A more recent paper in which the conceptual foundations of this probabilistic model are developed more fully is Maron's, "On indexing, retrieval and the meaning of about." Journal of the American Society for Information Science , v. 28:1 (January 1977), pp 38-43.

4. Much has been published in the area of decision analysis. A good introduction is Howard Raiffa's Decision Analysis: Introductory Lectures on Choices Under Uncertainty (Addison-Wesley Publishing Co., Reading, Mass., 1970).

5. Lotfi Zadeh, "The Concept of a Linguistic Variable and it's Application to Approximate Reasoning", Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, Memo. no. ERL-M411, 15 October 1973.

6. Karen Sparck Jones. Automatic Keyword Classification for Information Retrieval, Butterworths, London, 1971.

7. G. Salton, The SMART Retrieval System,---Experiment in Automatic Document Processing, Prentice-Hall, Englewood Cliffs, N.J., 1971.

8. For example, see Raghavan and Deogun's's "Optimal Determination of User-Oriented Clusters", ACM/SIGIR Conference Proceedings, June 1987, pp. 140-146.

9. Problems with the interpretation of the vector model of information retrieval are summarized by Raghavan and Wong in their "A Critical Analysis of Vector Space Model for Information Retrieval", Journal of the American Society to Information Science, v. 37:5, 1986, pp. 279-287.

10. See Quine's Methods of Logic (4th ed.). Harvard University Press, Cambridge, Mass. (1982) for a discussion of the issues of simplification in propositional logic (especially Chapter 11).

11. Verity Inc. "Corporate Backgrounder". June 1988, Mountain View, CA.

12. C.J. Van Rijsbergen, Information Retrieval , 2nd ed., Butterworths, London, 1979.

13. The COORDINATOR system is described in Fernando Flores' (with Michael Graves, Brad Hartfield and Terry Winograd) "Computer Systems and the Design of Organized Interaction" (ACM Transactions on Office Information Systems , v. 6:2, April 1988, pp. 153-172), and Terry Winograd's (with Fernando Flores) Understanding Computers and Cognition: A New Foundation for Design (Addison Wesley Reading, Mass., 1987). The LENS system is described in Thomas Malone's (with Kenneth Grant, Franklin Turbak, Stephen Brobst and Michael Cohen) "Intelligent Information Sharing Systems" (Communications of the ACM, v. 30:5, May 1987, pp. 390-402). WORKFLO is a product of FILENET. These systems, as well as the linguistic reasons for the poor retrieval of simple full-text systems, are discussed in more detail at the end of Chapter 4.

14. Craig Stanfill and Brewster Kahle, "Parallel Free-Text Search on the Connection Machine System", Communications of the ACM, v. 29:12, December 1986, pp. 1229-1239.

15. Jean Tague, "An evaluation of statistical association measures." Proceedings of the American Documentation Institute, vol. 3, 1966, pp 391-397.

16. C.J. Van Rijsbergen, "A theoretical basis for the use of co-occurrence data in information retrieval." Journal of Documentation, vol. 33:2, June 1977, pp 106-119.

17. F.W. Lancaster, Vocabulary Control for Information Retrieval . Information Resources Press, Washington, D.C., 1972.

18. There have been many discussions and empirical studies of inter-indexer inconsistency, but Zunde and Dexter's "Indexing Consistency and Quality" (American Documentation, pp. 259-267, July, 1969) is one of the high water marks in these investigations.

19. Jens Allwood, Lars-Gunnar Anderson and Osten Dahl. Logic in Linguistics. Cambridge University Press, London, 1977, pp 128-129.

20. M.E. Maron, "Relational Data File 1: Design Philosophy", in Information Retrieval: A Critical View , G. Schecter, ed., Academic Press, London, 1967, pp. 211-233 (quotation, p. 218).

21. S. Warshall, "A Theorem on Boolean Matrices". Journal of the ACM, vol. 9, January 1962, pp 11-12.

22. Romualdas Skvarcius, and William B. Robinson. Discrete Mathematics with Computer Science Applications. Benjamin/Cummings Publishing Company, Inc., Menlo Park, Ca., 1986.

23. *Op. Cit.*

24. David C. Blair, "An Extensional Semantic analysis of document indexing", working paper no. 395 (Division of Research, Graduate School of Business, University of Michigan). A revised version of a 1976 working paper from the University of California, Berkeley.

25. Alain Jacquesson and William Schieber, "Term Association Analysis on a Large File of Bibliographic Vocabulary", Information Storage and Retrieval, v. 9, 1973, pp. 85-94.