X. PROCEDURES FOR STATISTICAL PROCESSING AND REQUEST ALTERATION

Michael Lesk

1. Introduction

In order to find answers to requests, and to detect relationships between concepts, it is necessary to evaluate the similarity between members of a set of vectors. These vectors may represent requests in a space of concepts; concepts specified by their occurrences in sentences; or documents. The SMART system contains programs for the evaluation of relationships between such vectors. These programs operate on rows of a matrix (each row representing one vector). The values of the matrix elements are the frequencies of occurrence of the row-column pair. That is, in the document-concept matrix, the matrix elements represent the occurrences of concepts in documents; in the concept-sentence matrix, the elements represent the occurrences of concepts in sentences.

The programs to be described can produce a numerical measure of the similarity of two rows (their correlation), or a list of the rows "significantly" related to a selected row, or groupings of rows significantly related to each other. Request answering is then performed by asking the program to produce a list of the documents related to the request documents, using the correlations of the document-concept matrix. The matrices are stored as chained arrays, and may thus be altered easily. Alteration of request vectors, for example, is readily performed by one of two possible

methods. One may use for this purpose preset relationships between concepts, in the form of a hierarchy; or one may use statistical methods to detect concept relationships. Vectors of requests may be either expanded or contracted.

Since most of the internal data accumulated for each document appear in the form of such vectors, the programs that process them are of great importance. The present description is divided into two groups: programs applying statistical procedures to obtain term and document correlations, and programs handling the request processing.

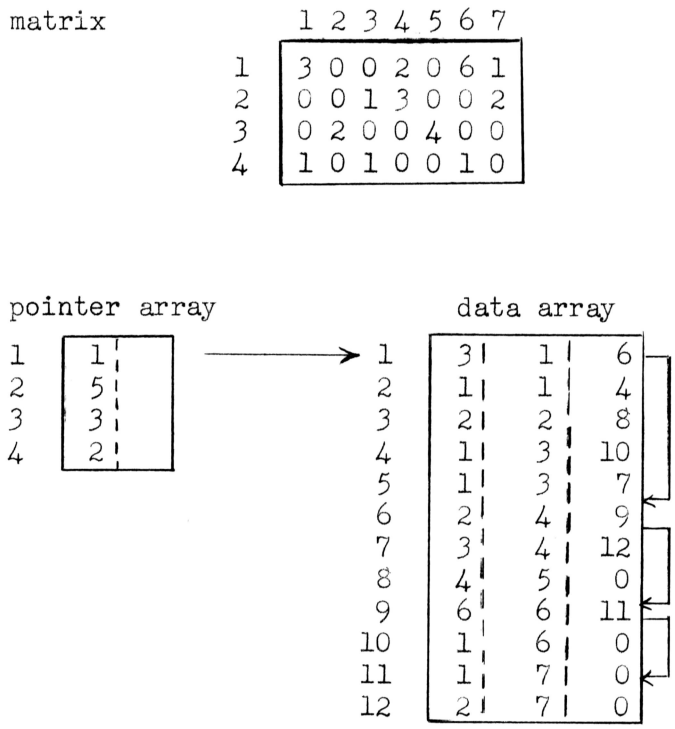
2. Statistical Programs

Statistical processing is done in SMART at three different times. These are: term-term processing within a single document, document-document processing when all documents have been read in, and concept-concept processing using the distribution of concepts over the document collection.

The same basic programs are used in each case. The programs use a set of chained lists, one for each "row" in the matrix being correlated. Two arrays are used. The first is a pointer array which includes one word for each row, a FØRTRAN integer giving the subscript in the second array of the first matrix element of that row. Within the second array, called the data array, each word contains three pieces of information: the first nine bits contain the numeric size of the element, the next twelve bits the column number, and the last fifteen bits the subscript in this list of the

next nonzero element in the same row (see Fig. 1). These two arrays are IROWST and FREQ in the term-term correlation, MSTART and DOCWRD in the document-document correlation, and IROWST and IWDLST in the concept-concept correlation.

IROWST-FREQ is set up by subroutine MATRIX from IWDLST, the input text (see Flowchart 3).* The rows are concept numbers and the columns



NOTE: This is only one of many acceptable representations of this matrix.

Encoding of an Occurrence Matrix

Figure 1

*The flowcharts appear in the Appendix to this section.

sentence numbers. MSTART-DOCWRD is accumulated as the documents are read in. The row identifiers represent document numbers and the column markers identify concept numbers. Transposing this, using subroutine TRANS yields the IROWST-IWDLST concept-concept matrix.

Three methods are available for correlating these matrices, referred to as cosine, matching or overlap, and asymmetric. When used to specify the logical options, these are punched as COS, OVLAP, or ASYM, respectively. Given two vectors x_i and y_i, the coefficients used are:

$$COS_{xy} = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2 \sum y_i^2}}$$

$$OVLAP_{xy} = \frac{\sum \min(x_i, y_i)}{\min(\sum x_i, \sum y_i)}$$

$$ASYM_{xy} = \frac{\sum \min(x_i, y_i)}{\sum x_i} \neq ASYM_{yx},$$

as shown in Fig. 2. To perform these correlations, subroutine ROWSUM is used to form the sums needed for the denominators (Flowchart 4). The correlation mode is transmitted through location PARAM1, and the matrix locations function as arguments for ROWSUM. ROWSUM leaves the sum of the elements (or their squares, as the case may be) of each row in array SUMSQ. ROCOR, shown in

$\vec{a} = 3\ 0\ 0\ 2\ 0\ 6\ 1$

$\vec{b} = 0\ 0\ 1\ 3\ 0\ 0\ 2$

$$\underline{\text{Cos}}\quad r_{ab} = \frac{\sum a_i b_i}{\sqrt{\sum a_i^2 \sum b_i^2}} = \frac{2 \cdot 3 + 1 \cdot 2}{\sqrt{(9+4+36+1)(1+9+4)}} = \frac{8}{\sqrt{700}} = \frac{4}{35}\sqrt{7} = .3$$

$$\underline{\text{Ovlap}}\quad r_{ab} = \frac{\sum \min(a_i, b_i)}{\min(\sum a_i, \sum b_i)} = \frac{2 + 1}{12} = .25$$

$$\underline{\text{Asym}}\quad r_{ab} = \frac{\sum \min(a_i, b_i)}{\sum a_i} = \frac{3}{12} = .25$$

$$r_{ba} = \frac{\sum \min(a_i, b_i)}{\sum b_i} = \frac{3}{6} = .5$$

Correlation Methods

Figure 2

Flowchart 5, now performs the actual correlations using the same parameters to determine the correlation mode. Each correlation is generated individually and sent to subroutines MCORR (if sense light 2 is on) and WCORR (if sense light 1 is on). MCORR stacks the correlations in a vector called IVCORR for further processing; WCORR stacks them in an array called PAGE for printing.

After each row is correlated, and its vector is stretched out in IVCORR, subroutine FRIEND is called to decide which of the correlating rows are to be considered "related" to the given row. Currently, FRIEND merely compares the correlations with a cutoff in KTEST; more sophisticated subroutines could, however, be substituted easily (see Flowchart 6). FRIEND copies the list of related rows into consecutive words of IGROUP, packing three 12-bit row numbers into one word. It also puts a pointer in the ISTART entry for that row, indicating the first word in IGROUP for this row (see Fig. 3). This procedure continues until the rows related to every row in the matrix are listed in IGROUP (the whole correlation matrix being too large to store in memory).

For many purposes, the statistical processing is now terminated. To obtain request answers, for example, the program can now produce the list of related documents for each request from the IGROUP matrix. Similarly, in concept-concept expansion, the program looks up the concepts related to each concept in the request vector and adds them to that vector.

It is possible, however, to go one step further and try an elementary clustering procedure. The list of "relations" may be considered as a kind

row i \ row j	1	2	3	4	5	6	7	8	
1	-	.5	.0	.9	.7	.1	.4	.3	
2	.5	-	.9	.0	.8	.0	.8	.7	...
3	.0	.9	-	.0	.8	.0	.4	.3	
⋮									

(a) Correlation vectors r_{ij}

apply cutoff .6

row	related to
1	4 5
2	3 5 7 8 ...
3	2 5
⋮	

(b) List of row relations

	ISTART
001	1
002	2
003	4
⋮	⋮

	IGROUP		
1	4	5	0
2	3	5	7
3	8	0	0
4	2	5	0
5			

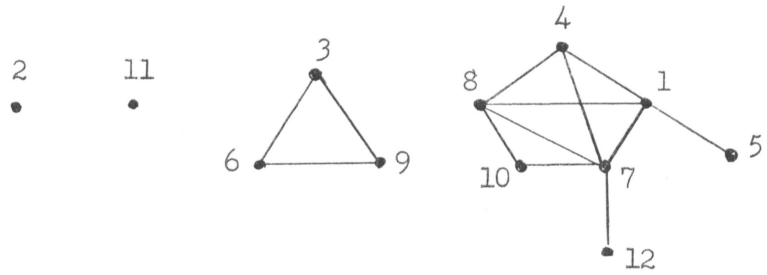
(c) ISTART - IGROUP

Formation of IGROUP

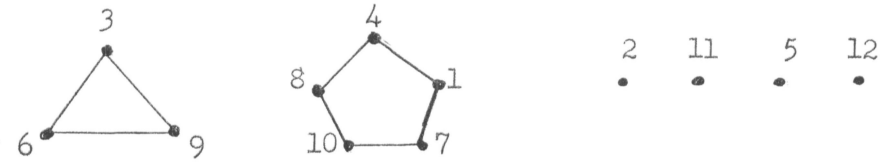
Figure 3

row	related to
1	4 5 7 8
2	
3	6 9
4	1 7 8
5	1
6	3 9
7	1 4 8 10 12
8	1 4 7 10
9	3 6
10	7 8
11	
12	7

(a) Relation list



(b) Relation graph



(c) Final clusters

(a) 3-6-9
(b) 1-4-7-8-10
(c) leftover single items: 2,5,11,12

Formation of Clusters

Figure 4

of graph, with lines connecting all related points. Subroutine CLUMP accumulates maximal connected subgraphs from the connection data. Another program, CDENSE, reduces the size of each cluster by throwing out the "ends" of all branches (i.e., those points with only one connection to another member of the subgraph (see Fig. 4)). WCLUST will now print out the cluster; alternatively, CLSMAP converts term clusters into a form usable for further processing.

In the concept-concept and document-document correlations, no use is currently made of the clusters except to print them. In the term-term correlation, however, the clusters can be used as phrases, and occurrences of these clusters may be counted as pseudo-concepts. This is done by setting up an array CLNUMS which indicates, for each term, the clusters to which it belongs. The text in IWDLST is scanned, and for each sentence the sum of the occurrences of members of each cluster is compiled in LITOCC. The entry of each cluster in LITOCC is now divided by the cluster size to determine the number of cluster occurrences, which are accumulated for the document in KLSOCC. This operation is performed by KCOUNT. Meanwhile, MAPKLS still retains the information necessary to interpret the cluster counts; i.e., the list of concept numbers in each cluster. This system may not be of much use when short documents are used, since all intradocument statistical techniques may then be of questionable applicability.

3. Request Processing

The SMART system, after processing all documents presented on the input tape, proceeds with an interdocument operation specified by the programmer. The major operation consists in the answering of requests and in the performance of the various request alterations.

The basic data useful for request answering is accumulated during Phase II, the document absorption stage. During this stage the program compiles the document-concept matrix, DOCWRD, which includes the concept-vector representation for all documents. This matrix is used together with an auxiliary array called MSTART. MSTART contains one word per document; this word is a FORTRAN integer giving the subscript I of the first element in DOCWRD associated with the given document. Reference to DOCWRD(I) provides the first element of the document-concept vector, consisting of one machine word divided into three fields. The first field gives the weight of this vector element, the second field stores the concept number of this term (the column identifier of the element). The weight is generally equal to the frequency of occurrence of that concept number in the text. Concepts which do not occur in a given text are omitted from the vector, so that no zero weights are included. The LOGICAL VECTORS (LV) option forces all weights to become equal to 1 as the matrix is formed, thus suppressing the frequency counting and correlating options. The final, third field of the DOCWRD item gives the subscript of the next word in DOCWRD associated with this concept vector. By following the pointers in the third field, the entire vector may be readily obtained. A zero pointer indicates the end of the vector.

Other arrays used in processing requests include NAME1 and NAME2, which provide the 12 BCD characters used to identify documents externally. Internally, the program uses consecutive numbers assigned to the documents as they are read in. These numbers are replaced on printouts by the corresponding 6-character words from NAME1 and NAME2. The identifiers stored in these vectors are supplied with the document at readin time. Another important array is DFLAG, which contains one word for each document. This word stores a nonzero address for each request document, a zero address for nonrequests.

The first step in request processing is a straightforward correlation, using the same programs that serve for the term-sentence correlation. Since the DOCWRD-MSTART combination uses the same format as the FREQ-IROWST matrices, the identical programs suffice for both correlations; it is then necessary only to change the addresses of the arrays to permit all internal operations to be carried out correctly. For printout purposes, sense light 4 is turned on; this informs the program that Phase III is being executed and that row identifiers now represent documents and column identifiers concepts. MAX, the parameter indicating the largest row identifier, is also altered to represent the highest document number instead of the largest concept number.

Correlation is under the control of subroutine DCORR, which calls ROWSUM and ROCOR to perform the actual correlations. Three modes are again available, specified by the logical parameter CORMD3 and using the numerical cutoff CUTOF3. Correlations are printed by subroutines WCORR and WDCRND;

document-document relations are printed by WGROUP. To save time in correlating, ROCOR correlates only those documents that have been marked as requests if sense light 3 is on. Sense light 3 is set automatically unless the programmer specifies an option such as DOCUMENT CORRELATIONS (DC), DOCUMENT RELATIONS (DR), or CLUSTER DOCUMENTS (KD). If only request correlations are wanted REQUEST CORRELATIONS (RC) prints them. Request relations - listing those documents considered "related" to a given request - produces request "answers"; these "answers" are printed by ANSWER REQUESTS (AR), probably the most common specification in the entire system.

Since the initial answers are frequently unusable for one reason or another, provision is made for altering requests and recomputing the correlations in the hope of altering the list of answers. Two basic schemes are provided for the request alteration: concept-concept correlation and hierarchical expansion. As currently programmed, these options modify only the request vectors; this is, however, a logical decision and the programming changes required to operate on all documents are slight.

Concept-concept expansion is a statistical technique. The system attempts to add to a given request those concepts which, although originally omitted, normally co-occur in the document concept vectors with the concepts actually present in the request. The statistical processing is handled by the same subroutine package that is used in the two other correlation systems. The input matrix is the transpose of the document-concept matrix, and gives for each document the occurrence frequencies of each concept.

Concepts are represented by the row identifiers, and documents by column markers. The matrix is obtained by transposition of DOCWRD. Subroutine TRANS performs the transposition; it uses the areas IWDLST, IROWST, and FREQ as storage space and generates an IWLST-IROWST pair of arrays identical in format to FREQ-IROWST or DOCWRD-MSTART. Subroutine CONCOR directs the correlation of this matrix by ROCOR and ROWSUM. Modes and cutoffs are controlled by parameters CORMD2 and CUTOF2. The programmer can obtain, as intermediate output, the total number of occurrences of each concept by using the CONCEPT FREQUENCIES (CF) option; similarly, the correlations can by obtained by specifying CONCEPT CORRELATIONS (CC), or the concept-concept relations by a CONCEPT RELATIONS (CR) specification.

The actual expansion is performed by the use of the table of concept-concept relations (see Flowchart 8). Subroutine SMEAR, using DFLAG, selects all request vectors and calls subroutine LINEUP to extract them from the chained structure and line them up in convenient format. Core area IVCORR is used for the concepts and MAINSZ for the associated weights. SMEAR then runs down the list of concepts in each request. Every concept is looked up in the concept relations table (which is again kept in ISTART-IGROUP matrices, as are the term-term relations and the document-document relations), and its related concepts are added to the request vectors. If the related concepts are already in the vector, the respective weights are incremented. Subroutine FILLIN is used to restore the vectors into the chained format of DOCWRD for recorrelation. This process is specified by the SMEAR CONCEPTS (SC) options. The vectors are printed before and after expansion if the SMEARED VECTORS (SV) option is given.

The second expansion method uses the hierarchical system. Subroutine TREES plays the same role here as SMEAR does in the concept-concept expansion. It first calls each request in turn; the hierarchical programs are then called to perform the vector alterations; finally, the adjusted vectors are returned into DOCWRD (using FILLIN). The HIERARCHICAL VECTORS (HV) specification controls the printing of "before and after" vectors. The expansion is called if the specification HIERARCHICAL EXPANSION (HE) is given (see Flowchart 7).

The actual expansion is performed by the programs described in Sec. V of this report. Subroutine FOREST is called first to read the hierarchy from the library tape. Then one of four entry points is called, depending on the setting of the logical parameter GOTREE. The possible values of GOTREE are ROOT, which causes expansion by parents (subroutine CLIMB); BRANCH, indicating expansion to sons (subroutine CHILD); FILIAL, denoting expansion by brothers (subroutine FILIAL); and REFER, producing expansion by cross references (CROSS). These subroutines are called for each request. An additional option decides whether the new concepts provided by the hierarchy will be added to the existing concepts, or will replace them. This specification is the REPLACE BY HIERARCHY (RH) option. The decision as to which hierarchical option is to be called is actually made in CHIEF. TREES calls a general entry point named GEORGE, located in the main program, which is set to CLIMB, CHILD, CROSS, or FILIAL, as required (see Fig. 5).

Concept	1	2	3	4	5	6
Weights	0	0	2	0	1	0

(a) Original request

Relations of concepts (from IGROUP)

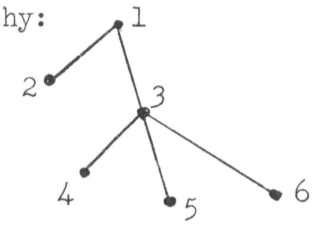
concept	related to
1	2
2	1
3	4
4	3
5	3
6	

Resulting request

Concept	1	2	3	4	5	6
Weight	0	0	3	2	1	0

(b) Concept-concept expansion

Hierarchy:



Expand by parents; resulting request

Concept	1	2	3	4	5	6
Weight	3	0	6	2	1	0

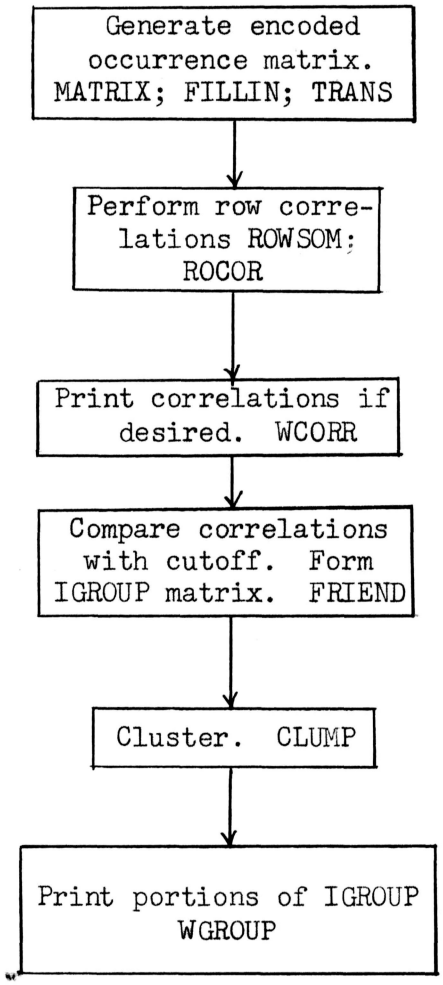
(c) Hierarchical expansion

Request Alteration

Figure 5

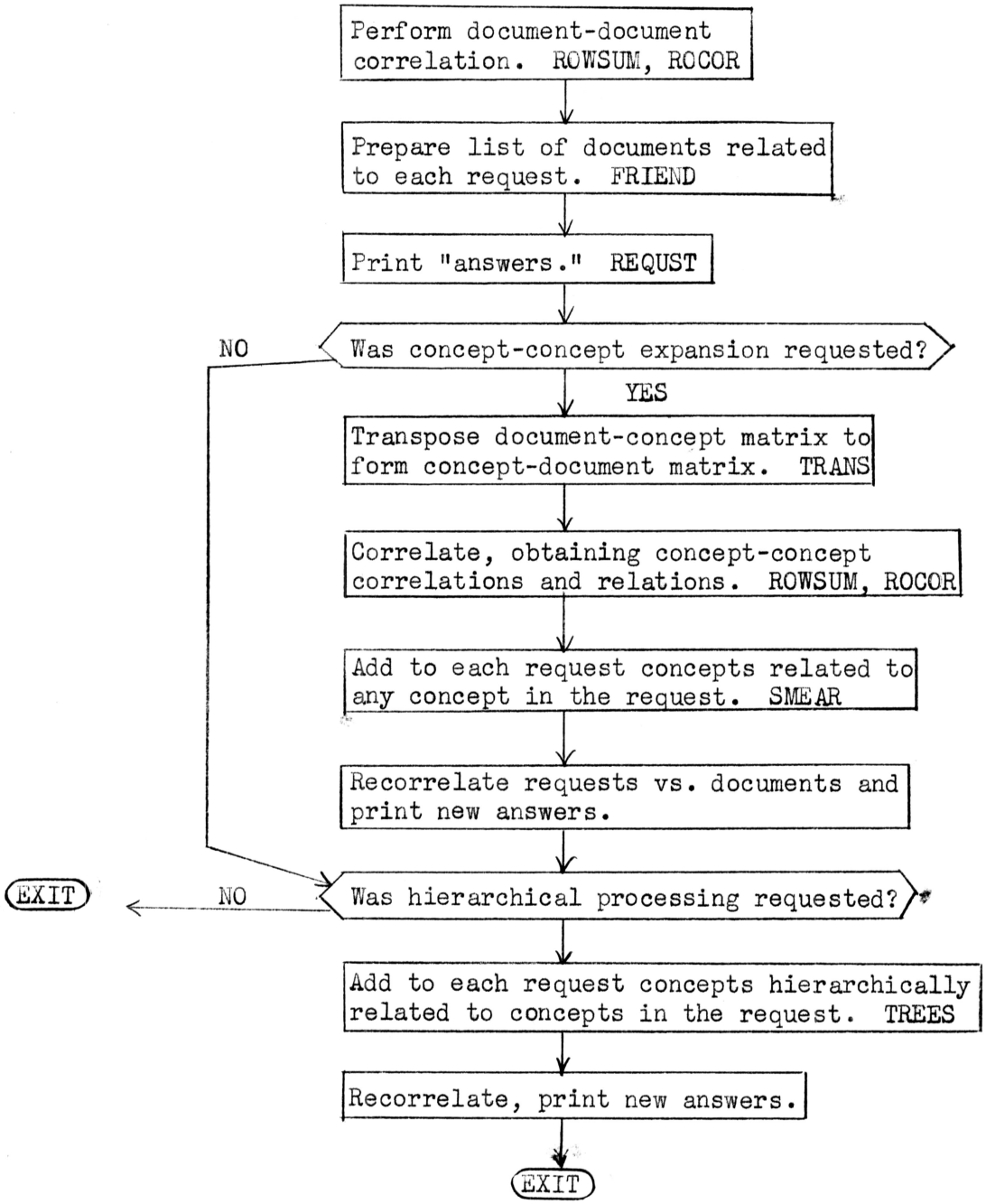
After expanding all requests, DCORR is again called to recorrelate the request documents. If a correlation printing option is given, the request correlations are printed again. The subroutine REQUST is then called to print the new answers. These may now be compared with the previous set, and the effectiveness of the expansion method may be studied.

APPENDIX



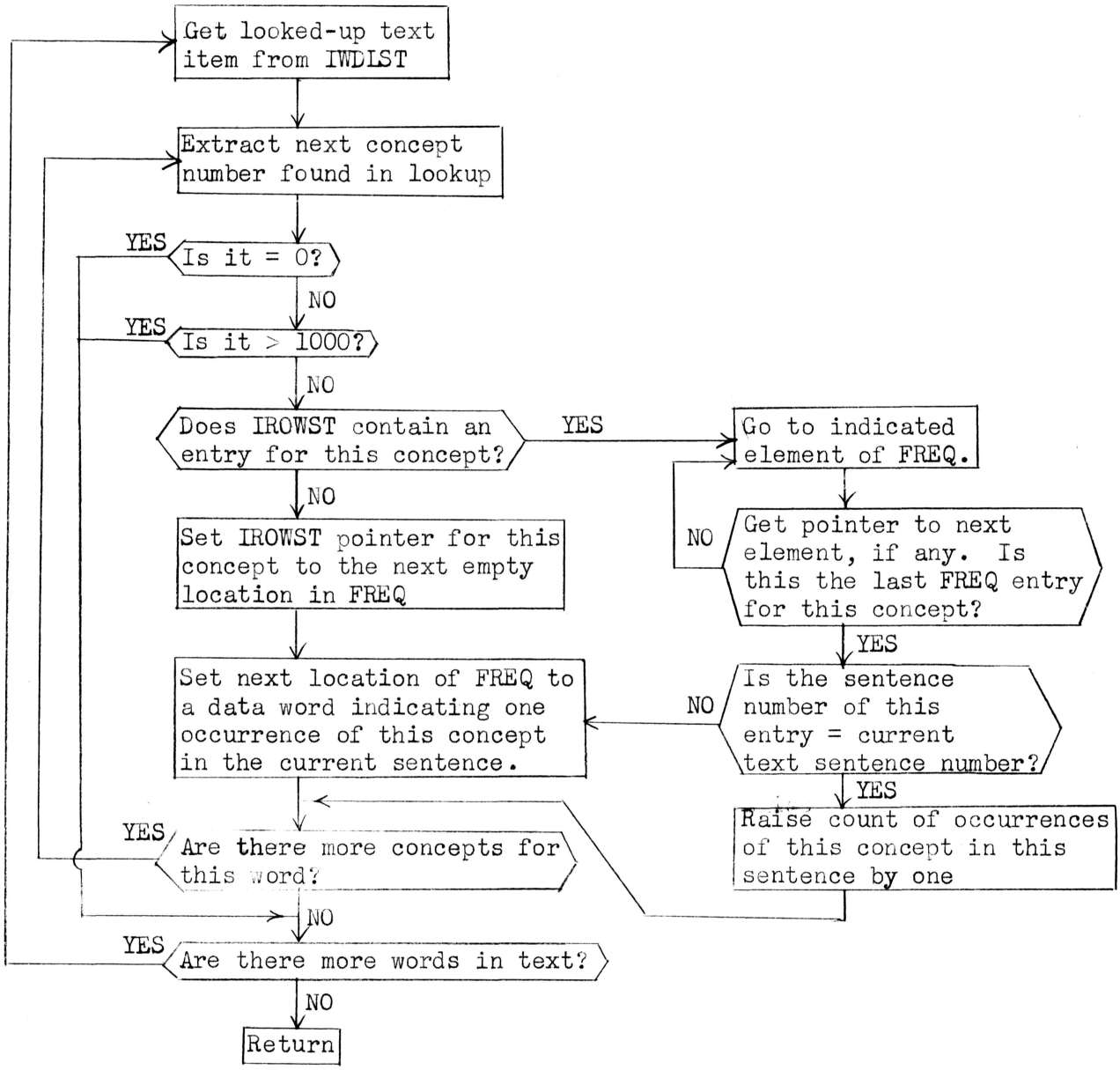
Statistical Processing

Flowchart 1



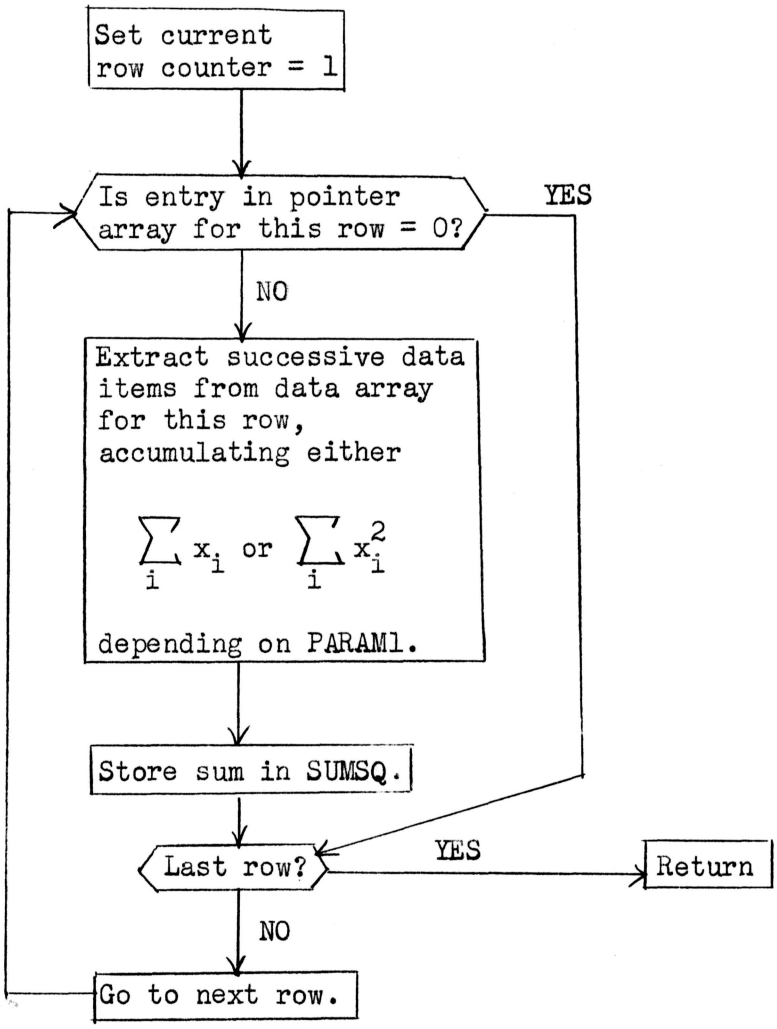
Request Processing

Flowchart 2



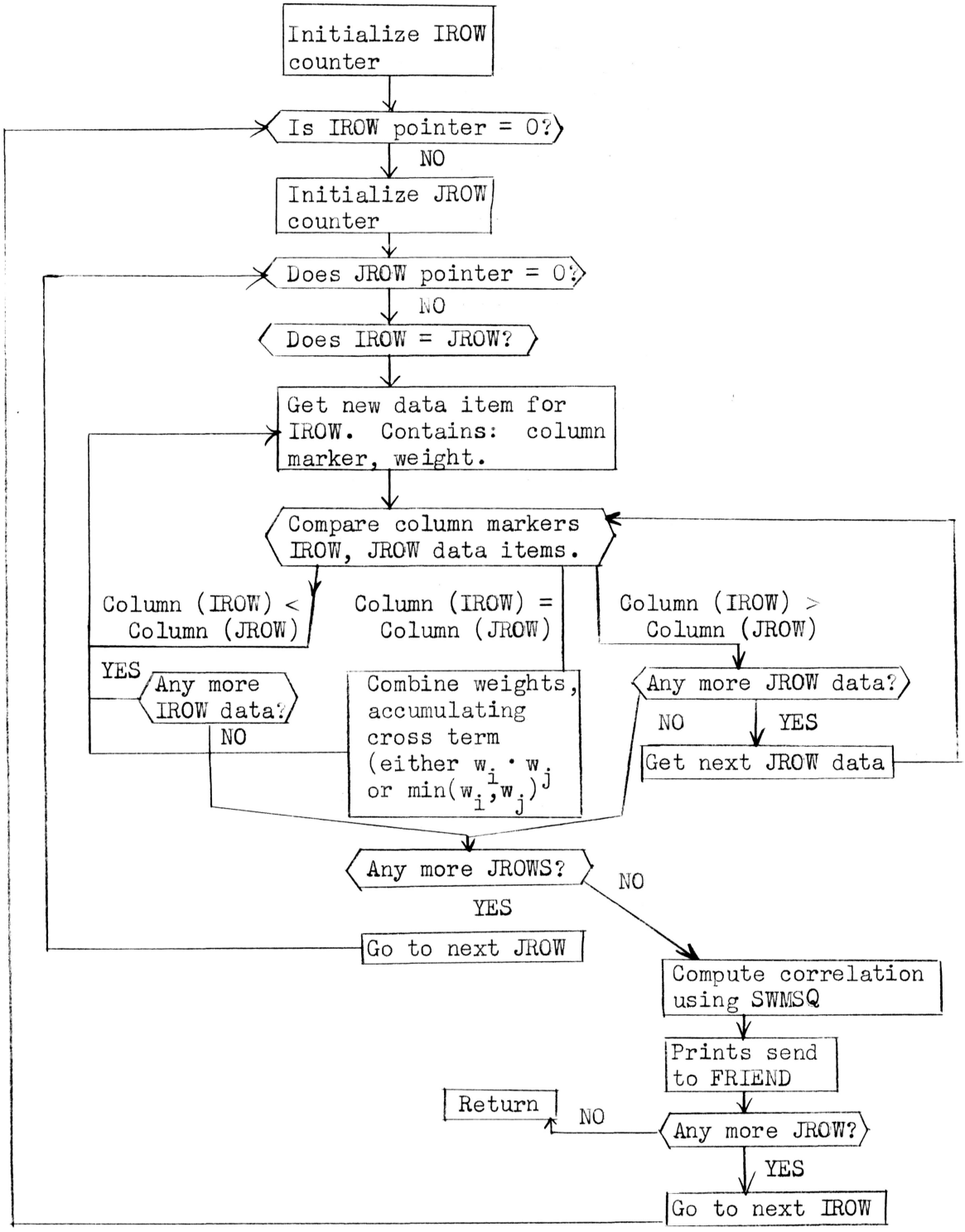
MATRIX

Flowchart 3



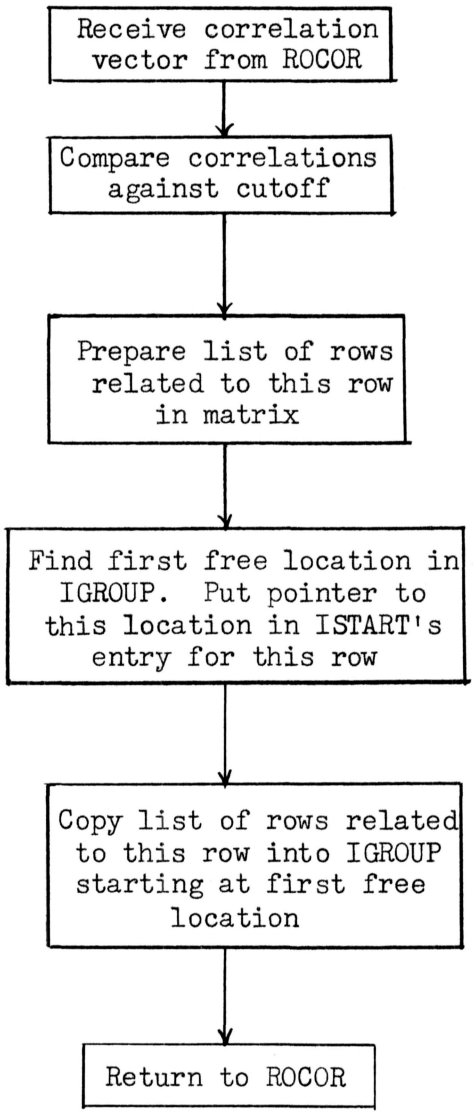
ROWSUM

Flowchart 4



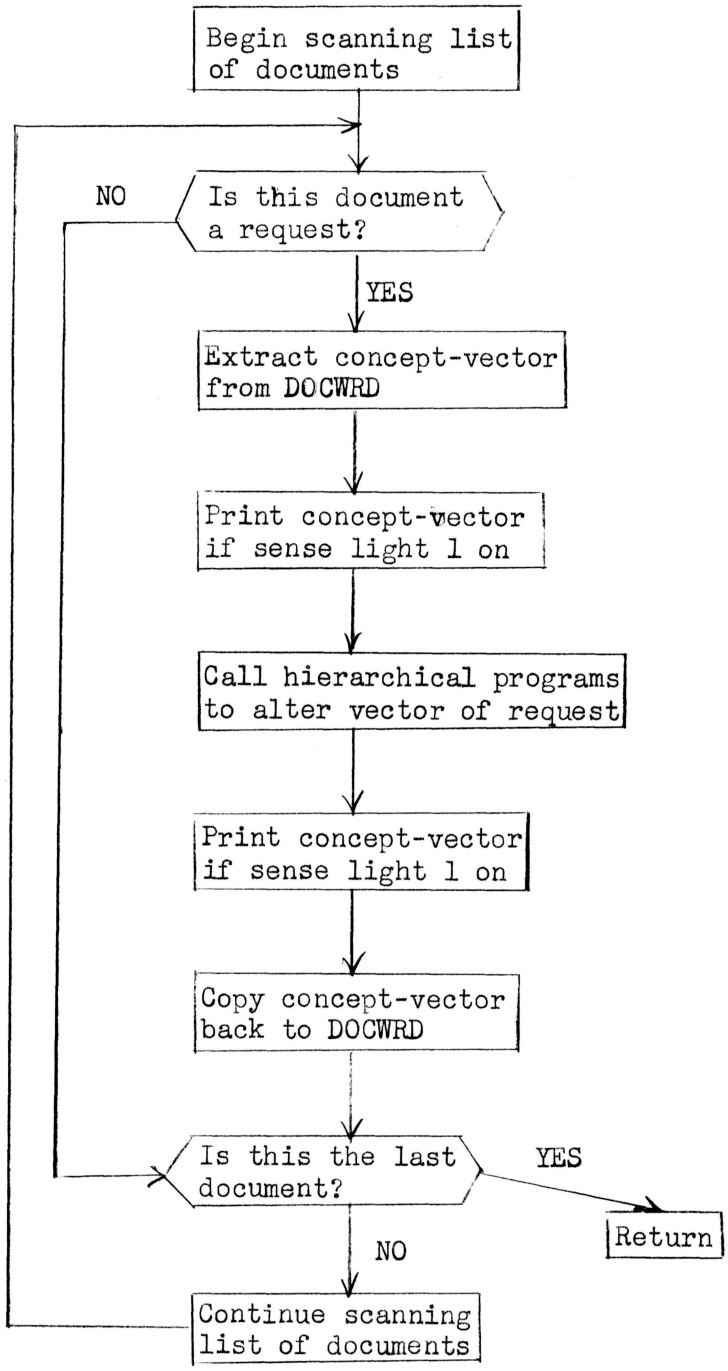
ROCOR

Flowchart 5



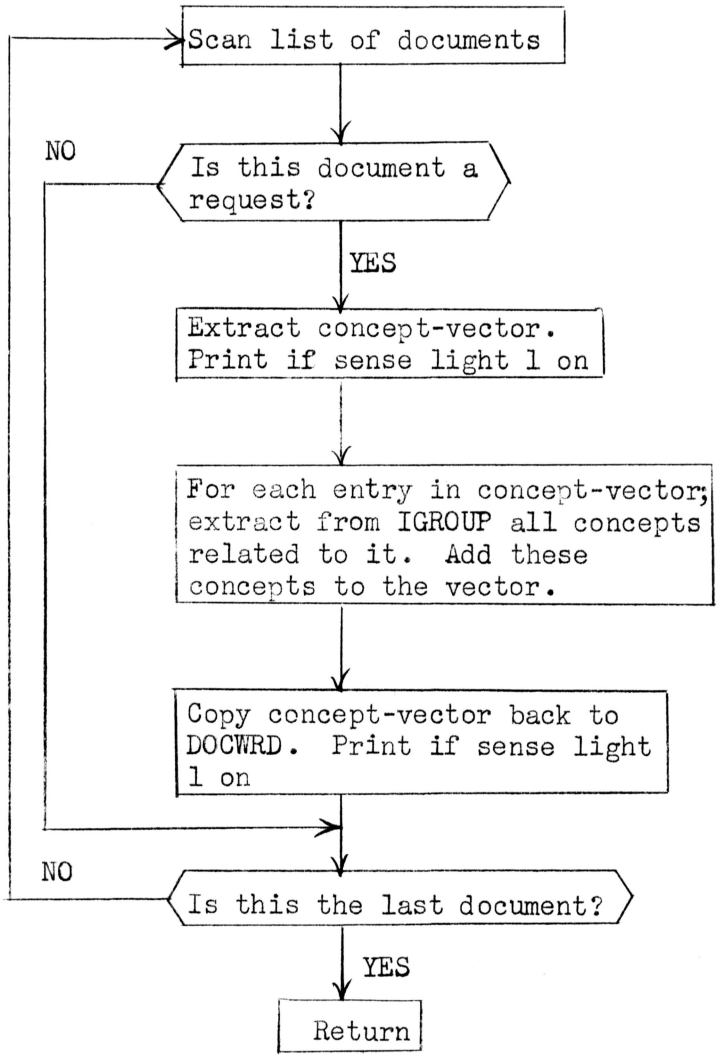
FRIEND

Flowchart 6



TREES

Flowchart 7



SMEAR

Flowchart 8