XII. STATISTICAL PHRASE PROCESSING

G. Shapiro

1. Introduction

The SMART system has capabilities for processing so-called "statistical phrases." These consist of sets of concept numbers, such that the components of each phrase are filed under a single concept number (the concept number of the phrase). A statistical phrase is said to occur in a sentence if all of its components occur at least once. Unlike the syntactic phrases described by Lemmon in a later section of this report, the occurrence of a statistical phrase in a sentence implies no syntactic relation among its components. It should be noted that, though concept clusters are produced by a special algorithm while statistical phrases are determined by a lookup process in a man-made dictionary, the procedure for using cluster or statistical phrase occurrences in a sentence is identical.[†] For this reason, subroutine PHROCC (described in Part 3 of this section) handles both of these expansion options. Part 2 of this section describes the programs used in writing and maintaining the statistical phrase file on the library tape.

[†] If either the statistical phrase or cluster options are used, the concept numbers of the phrases found are added to the concept vectors of the documents, and are taken into account (with the weight prescribed) in all subsequent manipulations of these vectors.

2. Maintenance of the Statistical Phrase File

During an update run, subroutine MAKE writes the statistical phrase file of the library tape. Up to six components are permitted for a phrase (in practice most phrases have only two). The input card format for MAKE is as follows:

```
Column 1        5 6 7        11 12 13        17 18 19        23 24 25 ...
         P No.    λ   COMP 1     λ     COMP 2     λ     COMP 3     λ    ...
```

where P No. is the concept number of the phrase and COMP I, I = 1,...,6 are the component concept numbers (all in integer format). λ represents an empty field. The list of component concept numbers is assumed to terminate when the first zero entry is encountered; if no zero entry is found, six components are assumed. No phrase may have fewer than two components. The integers may be justified anywhere in the appropriate field. If any alphabetic characters appear or if there are fewer than two components, the card is ignored during processing and a message and description of the offending cards are written on the output tape. The input deck must be terminated by a card with the characters NOMOR in columns 1 through 5.

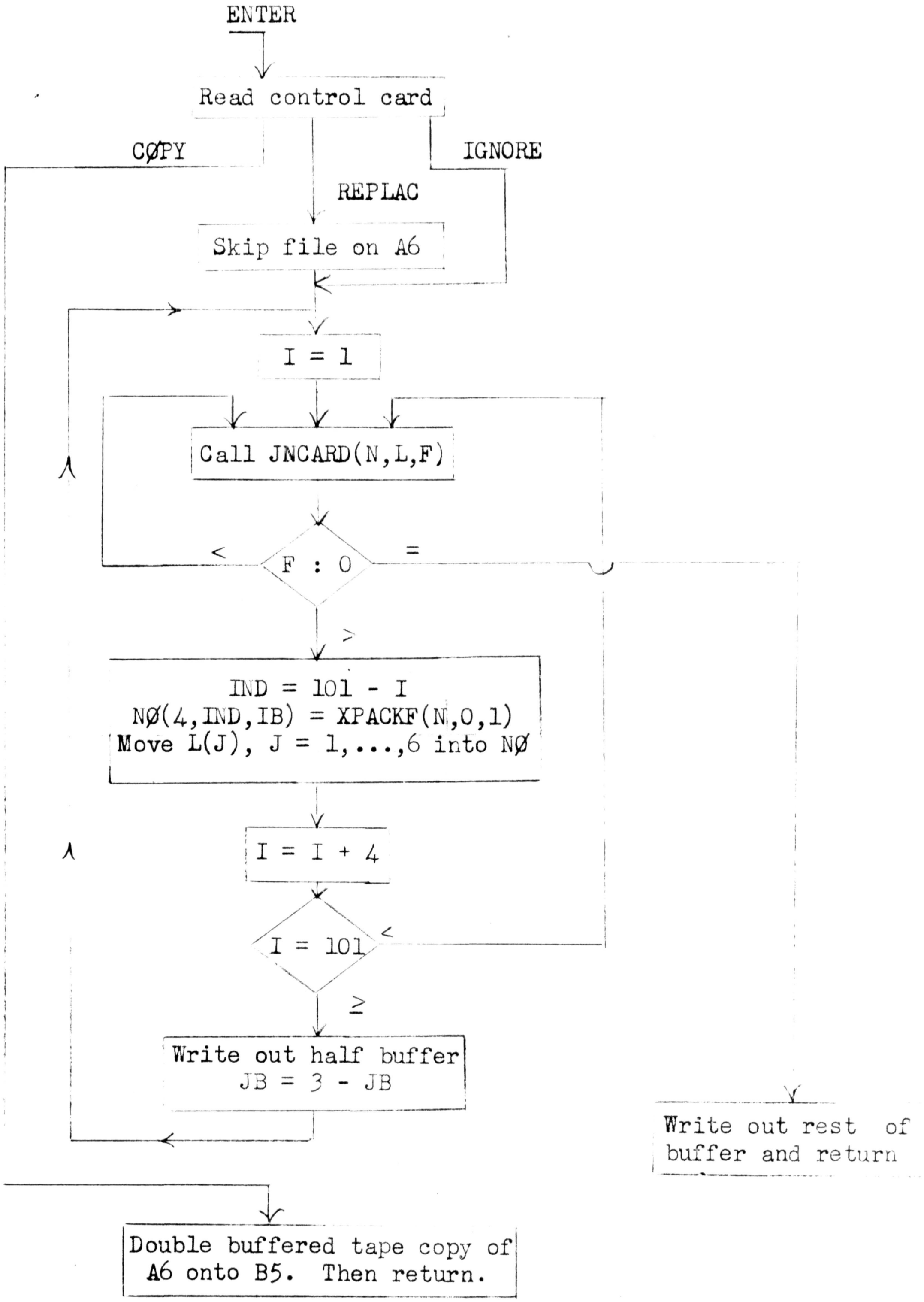
MAKE operates in three different modes, the mode depending on the first card of the input deck. If the first card contains (in columns 1 through 6) the word REPLAC, the succeeding cards are written onto the library tape B5, and a file on A6, the old library tape, is skipped in order to position A6 correctly for the other programs in the update link. If the first card is IGNORE, MAKE acts exactly as it would in the REPLAC case but does not skip the file on A6 (in case the entire update run is from cards and tape

A6 has not been mounted). If the first card is COPY, MAKE copies the statistical phrase file from A6 onto B5. In this case, the input deck should consist only of the COPY card; the NOMOR card must not appear.

On tape, each phrase takes up four words, of which the first contains the phrase concept number in the decrement and a weight of 1 in the address (the weight is used to make the statistical phrase processing compatible with the cluster processing done by PHROCC, since the clusters may have weights other than 1). The next three words contain the list of component concept numbers packed two to a word, with all unused components being set to zero. One hundred phrases (400 words) are packed in one physical tape record. The last record of the statistical phrase file (and only the last) may contain fewer than 100 items.

MAKE reads the first card of the input deck and, depending on the option specified, transfers to one of two sections of the code. In the REPLAC or IGNORE cases, MAKE continues reading cards from the input tape A2. This reading is actually done by a routine named JNCARD which reads the card in alphabetic format, checks to make sure that the card is valid, and then converts the information to binary integers using a FAP-coded routine XDTB. Alternate half buffers of 100-item length are filled and written out using the trap-controlled input-output routine INOUT (called through interface routines WAIT, INPUT, and OUTL). In the COPY case, a simple double buffered tape copy, again using INOUT, is performed.

Flowchart 1 describes the operation of MAKE. JNCARD(N,L,F) returns the first word of the input card to N and the next six words into the array L. If F is less than zero on return, the card was invalid; if F = 0, the card contained NOMOR; if F is positive, the card was valid. The array NO



Subroutine MAKE

Flowchart 1

(dimensioned 4 by 100 by 2) contains both havles of the buffer used for output. IB is an index which indicates which half of the output buffer is currently being filled.

3. Finding Statistical Phrase and Cluster Occurrences

Subroutine PHROCC is the bookkeeping routine for the statistical phrase finding package in link 1 of SMART. The actual algorithm for finding statistical phrases is coded in subroutine WPHROC which is called by PHROCC. The functions of PHROCC are described first.

PHROCC is called by the main program of the lookup link once for each core-load of documents, as described in Sec. V of this report by M. Cane. PHROCC first determines whether statistical phrase or cluster expansion, or both, are requested, and sets several flags accordingly. If both are requested, the statistical phrase expansion is performed before the cluster expansion. Otherwise, the library tape is positioned before the appropriate file. PHROCC also determines whether the first document in this core-load is a continuation of the last document of the preceding core-load by interrogating the flag LASINC which is set nonzero by the link 1 supervisor if and only if the last document of the preceding batch was incomplete. In this case, there may be partial results from the last phrase lookup waiting on a scratch tape. If there are, PHROCC reads these in and initializes the array JP, JQ, and KQ accordingly (these are described below). In any event, PHROCC initializes these arrays.

The main purpose of PHROCC, however, is to control the double-buffered trap-controlled input with which the statistical phrase dictionary is read in from the library tape. The half-buffer size is 400 words, corresponding to the physical record length on tape. PHROCC also unpacks the packed information from the library tape into FORTRAN arrays, thereby speeding up the time-consuming search loops in WPHROC. WPHROC performs the phrase lookups for one dictionary buffer and one document each time it is called, and is therefore called many times from the inside of a loop in PHROCC. Before calling WPHROC for a given document, PHROCC produces an array IST, in which the Ith entry is an index in the array IWDLST pointing to the beginning of the Ith sentence of the current document. IWDLST is a 10 by 1000 FORTRAN array containing in each 10-word item all the information for a particular word in one of the documents of the current core-load (see Sec. V of this report).

After calling WPHROC for each document in the current core-load for one given buffer load of the dictionary, PHROCC reads in the next buffer load of the dictionary (and switches buffer halves). When the entire dictionary has been processed, PHROCC proceeds to unwind and possibly to write out the results of the WPHROC manipulations.

If the SYNTAX option had been requested and a statistical phrase lookup was just performed, then WPHROC will have produced a list of sentences in which statistical phrases were found. PHROCC must write out these sentences on tape A7, to be used as input for the syntax link since, in the present version of SMART, syntactic phrases are sought only in sentences in which statistical phrases had previously occurred. This saves much time in that sentences in which there are no statistical phrases, and which therefore could not possibly contain syntactic phrases are not processed by the time-consuming syntactic analyzer. The whole entry in IWDLST (rearranged slightly) is written out for each such sentence. LEM1(ID) contains the number of sentences in the IDth document which are to be written out. LEM2(ID,J), J = 1,...,LEM1(ID) contains, packed in one word, the beginning and end indices in IWDLST of the Jth sentence of document ID which is to be written out. If any sentences of a document are to be written out, then the twelve word BCD identifier of this document is written out first.

At this point, PHROCC unwinds the chained lists JQ and KQ in which WPHROC has entered the phrases found together with their weights. JP(ID) will contain a pointer to the location in the array JQ in which the first phrase of document ID has been placed. KQ is a pointer array for JQ. KQ(M) is the index in JQ of the next phrase found in the document in which the phrase in JQ(M) was found. The chained list for a particular document is sorted in numerical order of the phrase concept number. No phrase occurs more than once in the list for any one document (except that occurrences in the title are treated distinctly from those in the body of a document and are distinguished by a different tag field). If one phrase was found several times in a document, the weight is set accordingly by WPHROC. The list for each document terminates with the word 077777777777_8. PHROCC places the entries from JQ into the FORTRAN array MARK; between the lists for two successive documents the word 077777777777_8 is written (even if these lists are null). If the flag INCPLG is nonzero, then the last document of the current core-load is incomplete, in which case PHROCC sets a pointer in such a way that the program VECFM (see Sec. V) ignores the part of MARK corresponding to the last document. In this case, PHROCC writes this section of MARK onto a scratch tape. This tape is read in when PHROCC is called for the next core-load of documents and is used to initialize JQ correctly.

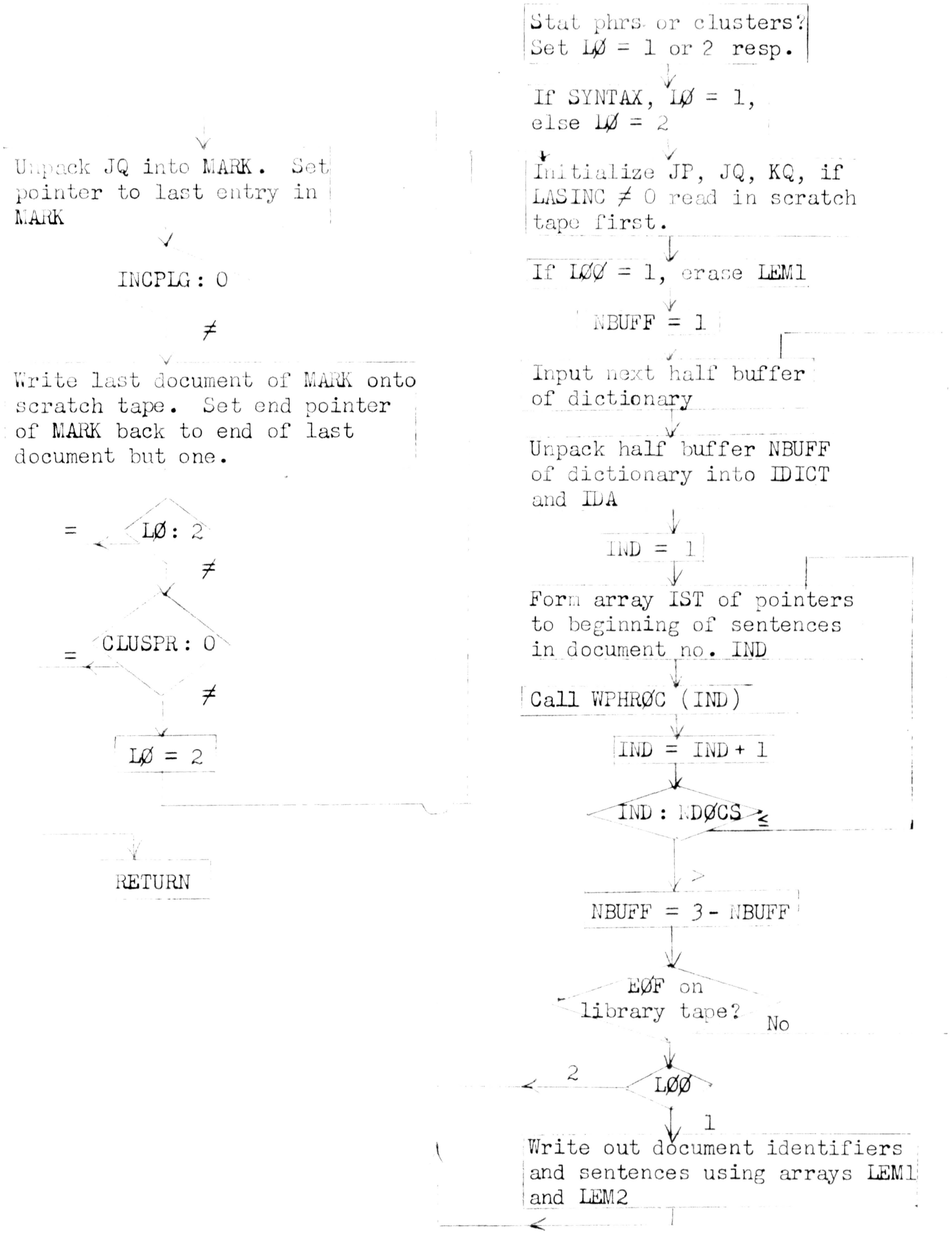
If both statistical phrase and clustering options were requested, PHROCC performs the lookup for both before returning to the calling program. Except for the fact that SYNTAX does not affect the clusters (i.e., nothing is written out on A7), and that the clusters always have negative concept numbers, the code is the same in either case.

Flowchart 2 describes the operation of PHROCC. NBUFF (which is either 1 or 2) indicates which half of the dictionary input buffer is currently being used.

Consider now the subroutine WPHROC. The data left for the calling program (of PHROCC) in the array MARK must be in the following format

S	12	3 17	18-20	21 35
		CON NO.	T	WEIGHT

where S = + or - according to whether statistical phrases or clusters are determined, CON NO. is the concept number of the phrase, and T = 6 if the phrase was found in the first sentence (i.e., title) of the document, and = 2 otherwise. WEIGHT is 12 times the number of occurrences of the phrase times the weight of the phrase (which is always 1 for statistical phrases, but may be some other integer in the case of a cluster).


Stat phrs or clusters? Set LØ = 1 or 2 resp.
If SYNTAX, LØ = 1, else LØ = 2
Initialize JP, JQ, KQ, if LASINC ≠ 0 read in scratch tape first.
If LØØ = 1, erase LEM1
NBUFF = 1
Input next half buffer of dictionary
Unpack half buffer NBUFF of dictionary into IDICT and IDA
IND = 1
Form array IST of pointers to beginning of sentences in document no. IND
Call WPHRØC (IND)
IND = IND + 1
IND : NDØCS
≤
>
NBUFF = 3 - NBUFF
EØF on library tape?
No
LØØ
2
1
Write out document identifiers and sentences using arrays LEM1 and LEM2
Unpack JQ into MARK. Set pointer to last entry in MARK
INCPLG : 0
≠
Write last document of MARK onto scratch tape. Set end pointer of MARK back to end of last document but one.
=
LØ : 2
≠
=
CLUSPR : 0
≠
LØ = 2
RETURN

Subroutine PHRØCC

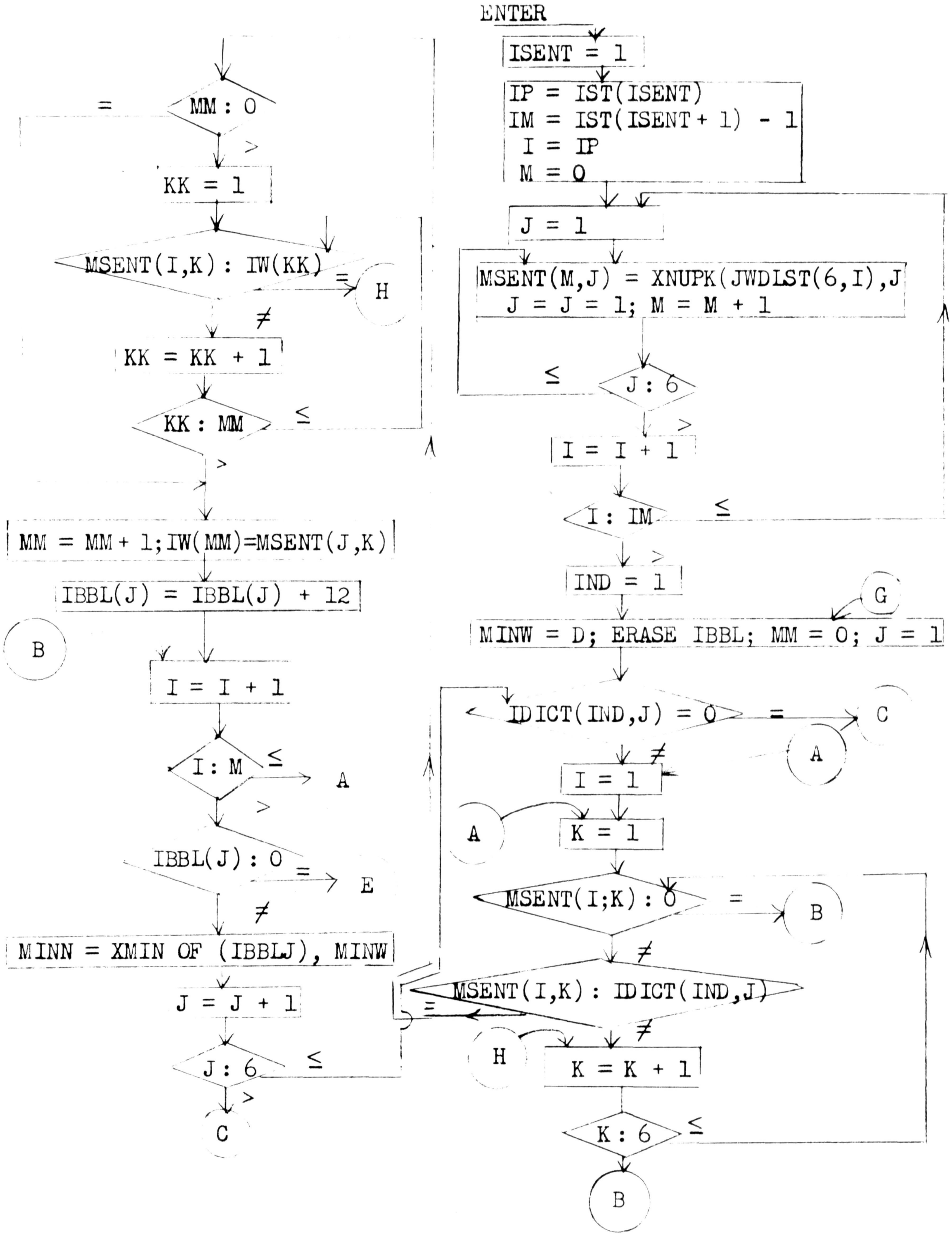
Flowchart 2

This is also the format used in JQ (except for the sign which would confuse attempts to order JQ; the sign is added later by PHROCC).

The number of occurrences of a phrase in a sentence is defined to be the minimum of the number of occurrences of each of its components in the sentence. Since many words have more than one associated concept number, a check is made to ensure that different components of a phrase are not found in the same word of a sentence (by keeping a list in the array IW of those words in which phrase components have already been found). One would, presumably, not wish to count such an event as an occurrence of a phrase.

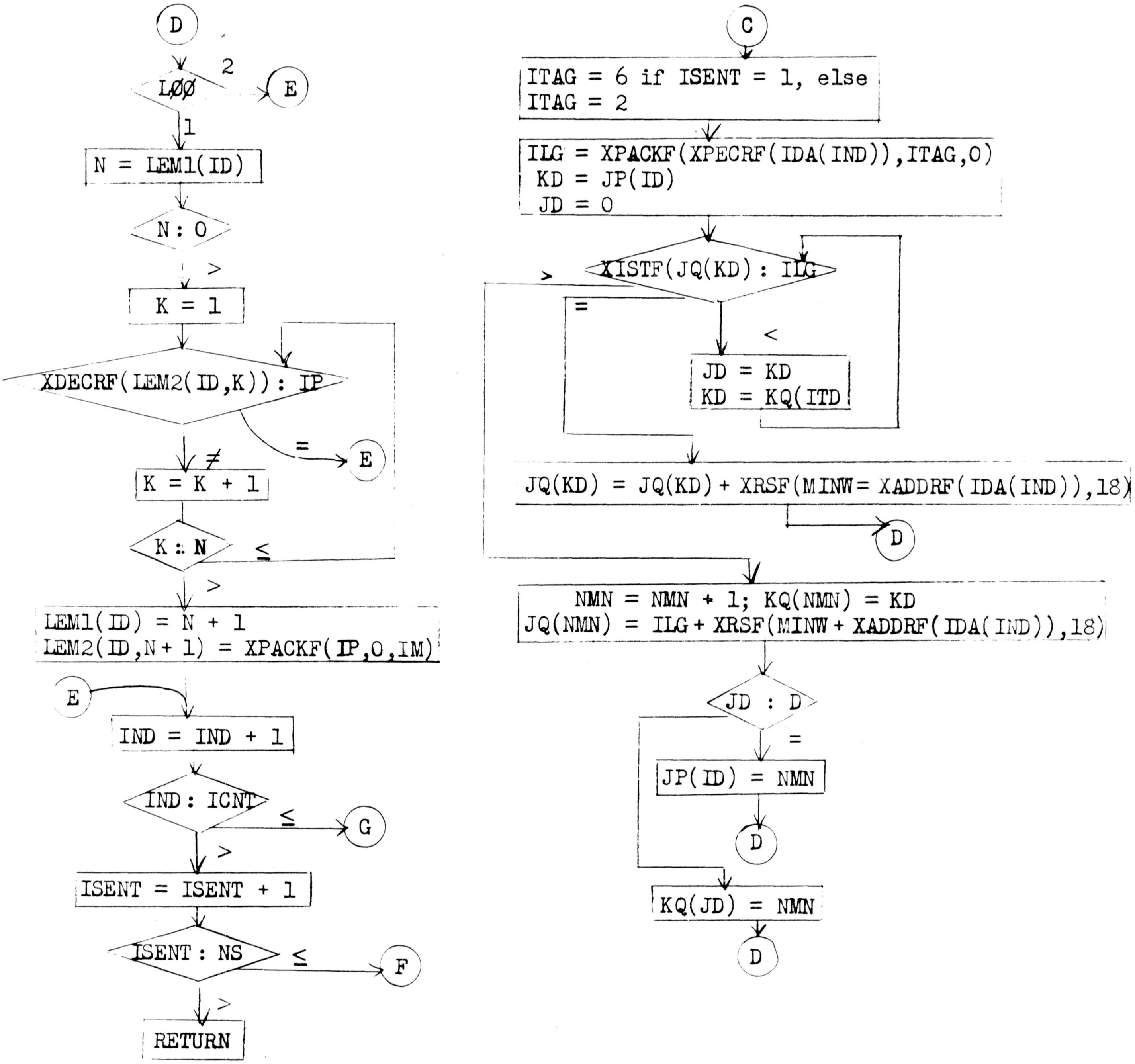
In order to save the time of unpacking words from IWDLST each time a test is to be made, WPHROC works one sentence at a time, first unpacking the words of the current sentence into an array called MSENT. Each phrase in the current buffer load of the dictionary (which has been unpacked into the array IDICT by PHROCC) is then searched for in the sentence under consideration. The method is completely straightforward: the first component of the phrase is matched against each component of each word in the sentence; the second component is then matched, and so on. The search terminates when either a zero component concept number of the phrase is reached, or six components have been processed, or when there are no matches for the present component of the phrase.

If a particular phrase is found in a sentence, the appropriate entry is made in the chained list JQ; if the phrases had been previously found in this document, no new entry need be made, rather the weight of the old entry is increased. JQ is always kept in order, eliminating the necessity for performing a sort before returning the information to either PHROCC or VECFM.



Subroutine WPHRØC (ID)

Flowchart 3



Flowchart 3 (continued)

If the syntax option is to be used, and WPHROC is currently processing statistical phrases, the appropriate entries are made in LEM1 and LEM2, provided that the current sentence had not been previously entered (i.e., even if many phrases are found in one sentence, the sentence is written out on A7 only once).

Flowchart 3 describes subroutine WPHROC in detail: NS is the number of sentences in the current document and is set by PHROCC. JNUPK(A,I) gives the decrement or address (according as I is even or odd) of the (I-1)/2 word after A. XMINOF(A,B) returns the minimum of its arguments. XPACKF(I,J,K) packs FORTRAN integers I,J,K, respectively, in the decrement, tag and address of a word; XDECRF(A) and XADDRF(A) return the decrement and address, respectively, of their arguments as a FORTRAN integer. XISTF(A) masks out the 15 low-order bits of its argument. XRSF(A,I) shifts A right 18 places. ID is the current document number (in this core-load) and is an argument used by WPHROC. IDA(I) contains the first word of the four-word entry for the Ith phrase in the current dictionary buffer, and IDICT(I,J) J = 1,...,6 contains the Jth component of the Ith phrase of the current buffer. These last two arrays are prepared by PHROCC.