XI. HOUSEKEEPING ROUTINES

Michael Lesk and Tom Evslin

1. SMART System Routines

SMART, like any large system, contains a number of purely internal subroutines whose functions do not logically contribute to the retrieval process and can largely be ignored by the user. These include, among others, the output tape editing system, a system for punching looked-up documents onto binary cards, and several others. The binary card system is perhaps the most important and will be discussed first.

Because of the need to call another chain link to perform lookups, the phase II processing of SMART, described in Sec. II of this report, is very expensive. A minimum of ten seconds is required to process a document through phase II. If syntactic analysis is requested, or if term clustering is to be applied to a relatively long document, another minute may be added to that time. It is then desirable to provide a method for preserving the results of phase II so that they may also be used in phase III. The chosen method is to punch the results of phase II into binary cards which may be reloaded at close to tape speed.

The information to be saved is contained in two arrays, KLSOCC and MAPKLS. KLSOCC contains the occurrences of each "cluster," where a cluster is either a group of terms or a single concept. The

correspondence between clusters and concepts is given by MAPKLS. Each entry in MAPKLS contains the cluster number of this entry, and the component concept numbers. Thus, each document is in a form compatible with any other document looked up in the same thesaurus, regardless of the phase II options used. These arrays, together with two constants defining their length, are punched into binary cards. The deck is produced by the PUNCH DOCUMENT DATA (PU) specification, which causes CHIEF to call LSTOUT; LSTOUT writes the binary decks or "lists" on the punch tape.

The format of these binary decks is as follows: First, the program punches two flip cards giving the twelve characters of the document name. Then, a *LIST card suitable for reintroducing the document during a subsequent run is punched. This card is identical with the *TEXT card which identifies the text when it is first read in, except that *TEXT is replaced with *LIST, and the date is inserted in columns 73-80. The document itself follows on binary cards with 5-7-9 punches in column 1. The binary cards are checksummed and serialized and should therefore not be tampered with by the programmer. One exception to this rule is allowed: The flip cards for identification may be inserted anywhere among the special binary cards of the list; alternatively the flip cards may be removed. The first binary card contains the two constants giving the length of KLSOCC and MAPKLS; the second card begins the listing of MAPKLS. When MAPKLS is terminated, KLSOCC occupies the remaining cards. A short abstract is represented by three to seven cards. A longer text by perhaps 20 cards.

The cards are read by subroutine LISTIN, which thoroughly checks the input decks. At least ten documents a second can be read in the binary mode. Note that the phase II options for such documents are those specified at look-up time and not those given at the time the binary decks are read in. Binary decks from different thesaurus should not be mixed because the concept numbers used may no longer correspond.

All input and output operations are checked carefully. Because of the mixed binary and BCD data on the input tape, all input subroutines are associated with the main link (after the update), and the look-ahead coding (which signals the mode of the next card on the input tape) is left in location LUKAHD as specified in the last part of Sec. II. The input subroutines are NAMES, which reads the list of identifiers for the thesaurus categories; CHIEF, which reads control cards; SEGMNT, which reads input texts and specification cards; and LISTIN, which reads in binary documents.

Print output is provided entirely through the FORTRAN write routines (STH) and (IOH). Packed records are used to save time. A subroutine called LINE keeps track of the number of lines written on each page, and calls HEADR when necessary to start a new page. In general, each new write subroutine calls HEADR to rehead the page when writing is started, and LINE is called before each write to see if enough space is still left on the current page.

Other auxiliary routines include:

ERROR - prints error messages for all main link programs;

RANNO - generates random numbers;

FLIP – punches identification flip cards;

SHORT – packs and unpacks portions of machine words for FORTRAN subroutines;

DOVER and SDOVER – conserves storage space by adding subroutines not needed in the run to the free storage chain of DOCWRD;

TAPENO – translates special code words into tape instructions. (To permit the specifications of logical tape addresses, FAP subroutines do not contain actual tape instructions at assembly time, but only code words indicating the type of command and the logical tape address; TAPENO translates these codes into real machine instructions);

WKUNO – copies the syntactic analysis output from A6 to A3 so that the programmer can look at the results of the analysis for each sentence;

WMODAL – writes the list of "frequent cluster sentences" on A6 for the syntactic analyzer preprocessor;

ENDEND – prints a termination message, empties buffers, and calls EXIT.

2. Memory Clean-up Procedures

DOVER and SDOVER are routines used to clear away unneeded subroutines at execution time, thus providing additional space in memory.

The DOVER routine contains a list of BCD subroutine names called SUBTBL. This list is broken into four parts, each part including those

subroutines that are never needed in later parts. For example, subroutines in part three may be used in any of the first three parts but are never used in the fourth.

The last card in SUBTBL is labeled ENDTBL. Following ENDTBL, a list appears containing the bits in the FLGWRD which determine whether a given subroutine is to be executed during the current run. This list must be stored in the same order as the subroutines in SUBTBL with which they correspond. A subroutine which is always needed up to and during execution of its phase is identified by a keyword with all its bits equal to one.

SDOVER is a routine called at load time. It picks up each item in SUBTBL and searches the loading tables to find the matching entry. If no entry is found, the item in SUBTBL is replaced by zeros. If the entry is found, the item in SUBTBL is replaced in the following manner: its address contains the first location of the subroutine and its decrement contains the first location not used by the subroutine.

DOVER has four entry points: SUBTBL, DSHIFT, DSTATE, and DOVER. SUBTBL is the first location of the list of subroutine names which SDOVER replaces with location data. DSTATE contains a one, two, three, or four indicating the part of SUBTBL currently in use. A call to DSHIFT results in a shifting of all the keywords following ENDTBL by fifteen bit positions to the right, so that the matching operation with FLGWRD, which has also been shifted, can be continued.

Entry DOVER results in an attempt to free the core occupied by subroutines found to be unneeded. Each nonzero item in SUBTBL is

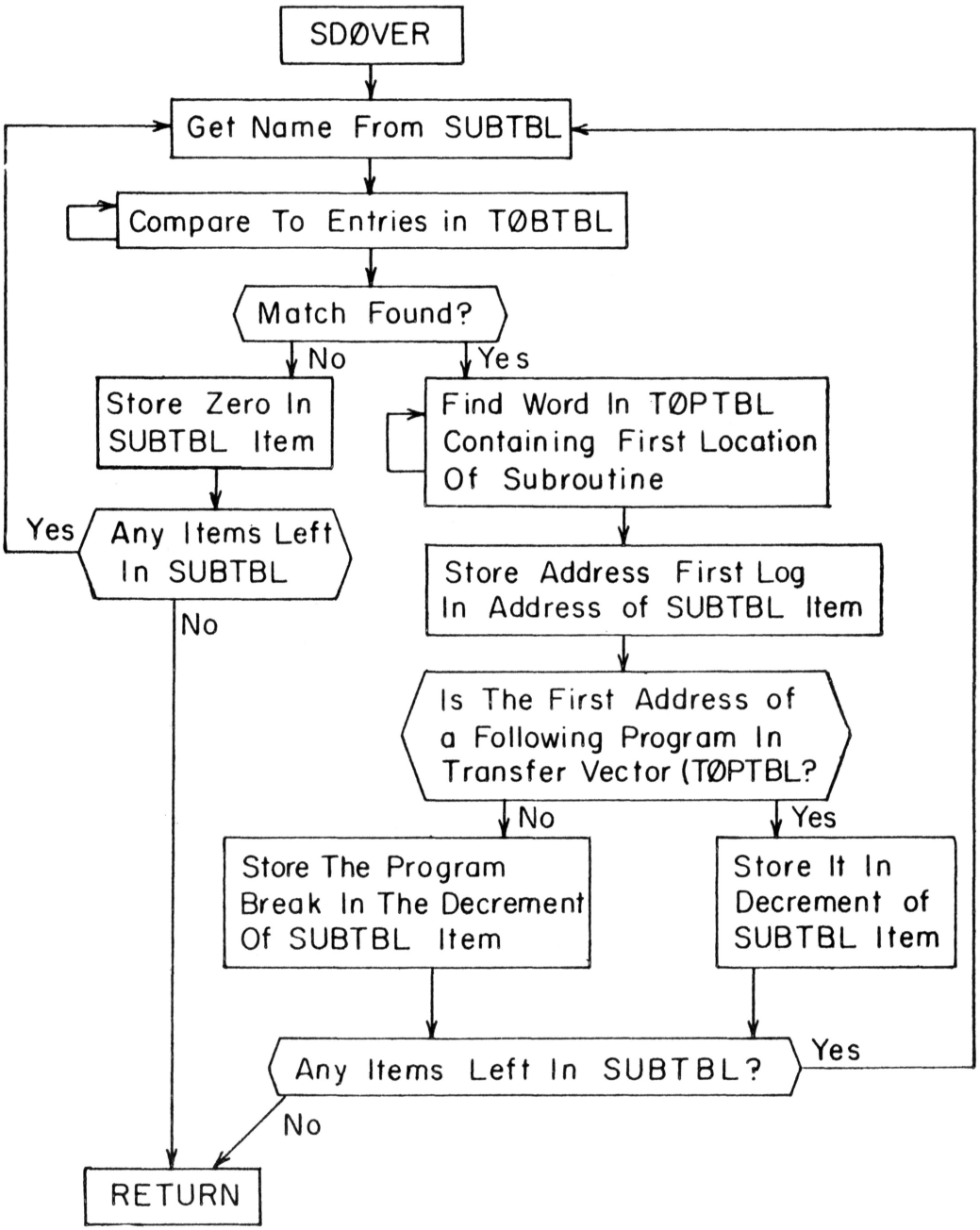
examined. If the entry is included in the present or following part, the word corresponding to it in the key bit list is also examined. Each bit which is "on" in the keyword must be "off" in the FLGWRD if the subroutine is to be deleted during its own or a preceding phase. All subroutines appearing in SUBTBL in a phase preceding the current phase will be deleted regardless of the contents of their keywords since they cannot possibly be of any further use.

When a subroutine is deleted, each word is cleared and its decrement is replaced by the address of the following word in a one dimensional FORTRAN array called LDOC. This means that most words contain the array address of the word below them in core. The first location of each subroutine will contain either the array address of the last location of the next subroutine cleared, or a zero if this is the last subroutine to be cleared by the current call to DOVER.

The array address of the first location cleared in the above manner will be returned to the decrement of the word addressed by the first parameter in the calling sequence to DOVER. All of these subroutines are written in FAP. Flowcharts 1 and 2 show the organization of SDOVER and DOVER respectively.

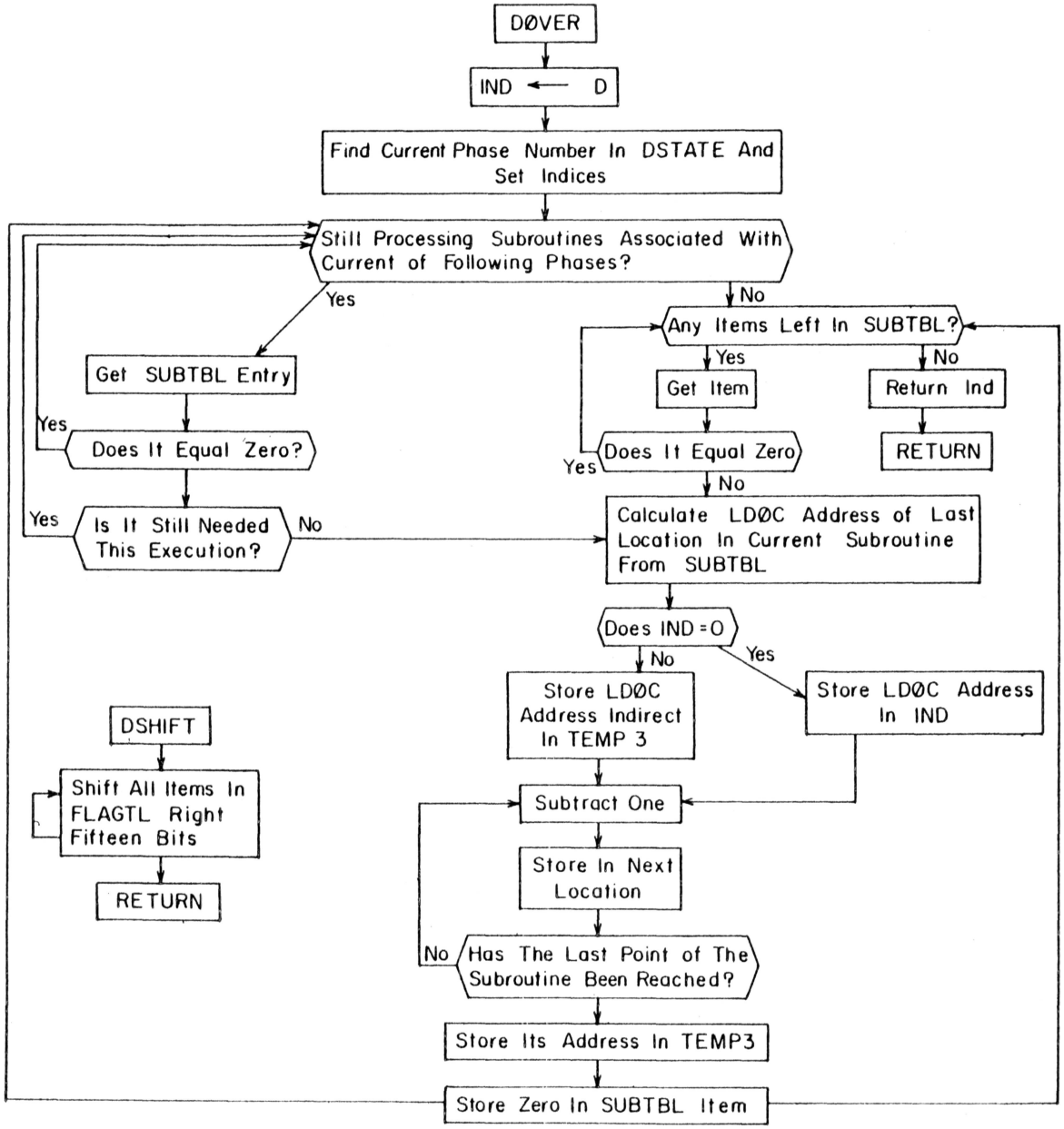
3. Formation of the Vacuous Thesaurus

For any document collection, it is possible to generate a null thesaurus automatically. This null thesaurus assigns to each noncommon word a unique concept number, and may be used for document processing in place of the thesaurus described by Harris in Sec. III.



Organization of SDØVER Routine

Flowchart 1



Organization of DØVER Routine

Flowchart 2

The null thesaurus is generated by a separate program, THES. THES is given a document collection and a list of common words, and generates a thesaurus for that collection. Cards in the correct format for the thesaurus updating program (Sec. IV) are punched out, along with a list of identifiers for the concept numbers. A frequency count of the collection is produced as a by-product of the program.

The common word list starts with the first card following the *DATA card and is terminated by a card containing slashes in columns 1-6. Each common word has a unique number greater than 1,000. Following this may be the optional *FREQ* card, the aforementioned characters to appear in columns 1-6.

This card causes a list of all noncommon words appearing in the collection to be printed out on tape A3 along with the frequency of their occurrences. In addition, if a left-justified number appears in columns 7-12 of this card, only those noncommon words with a frequency greater than or equal to this number will be included in the punched output, although in no case will more than 1,000 distinct noncommon word stems be punched.

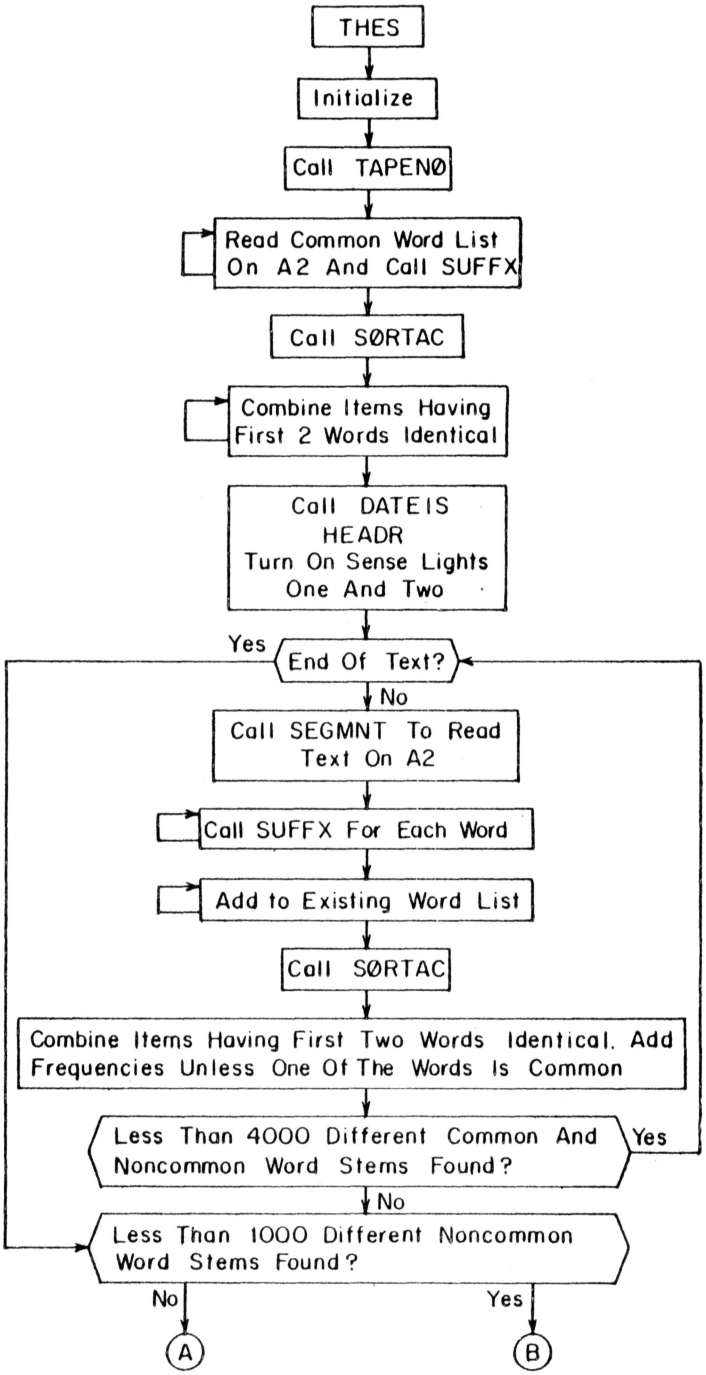
Following the *FREQ* card, the document collection itself must appear. Each document starts with a *TEXT card and is terminated by the appearance of the next *TEXT card, or the *STOP card which must follow the last document. Documents are limited to 1,000 words in length. Only the first 1,000 words of overlength documents are processed by THES and an error message results. It is desirable that documents be as close to the 1,000 word maximum as possible to save processing time.

After each document is read, all word suffixes are removed by a right-to-left suffix scan. Common words are also suffixed. Next an alphabetic sort is performed by SORTAC (WDSORT). All words whose first twelve characters after suffix splitting are identical are considered to be the same word. Any word which occurs in the common word list is ignored. If two matching words are found, both of which are noncommon, their frequencies are added and one of the entries is deleted. (Note: such pairs as "perfected" – suffixed perfect...ed – and "perfecting" – suffixed perfect...ing – are treated as one entry, their frequencies are combined, and one entry is deleted. This corresponds to the manner in which words are entered in the normal thesaurus.

Documents cease to be processed when the program has found more than 4,000 distinct common and noncommon word stems, or when a *STOP card is encountered. The number of documents processed is printed out in the former case.

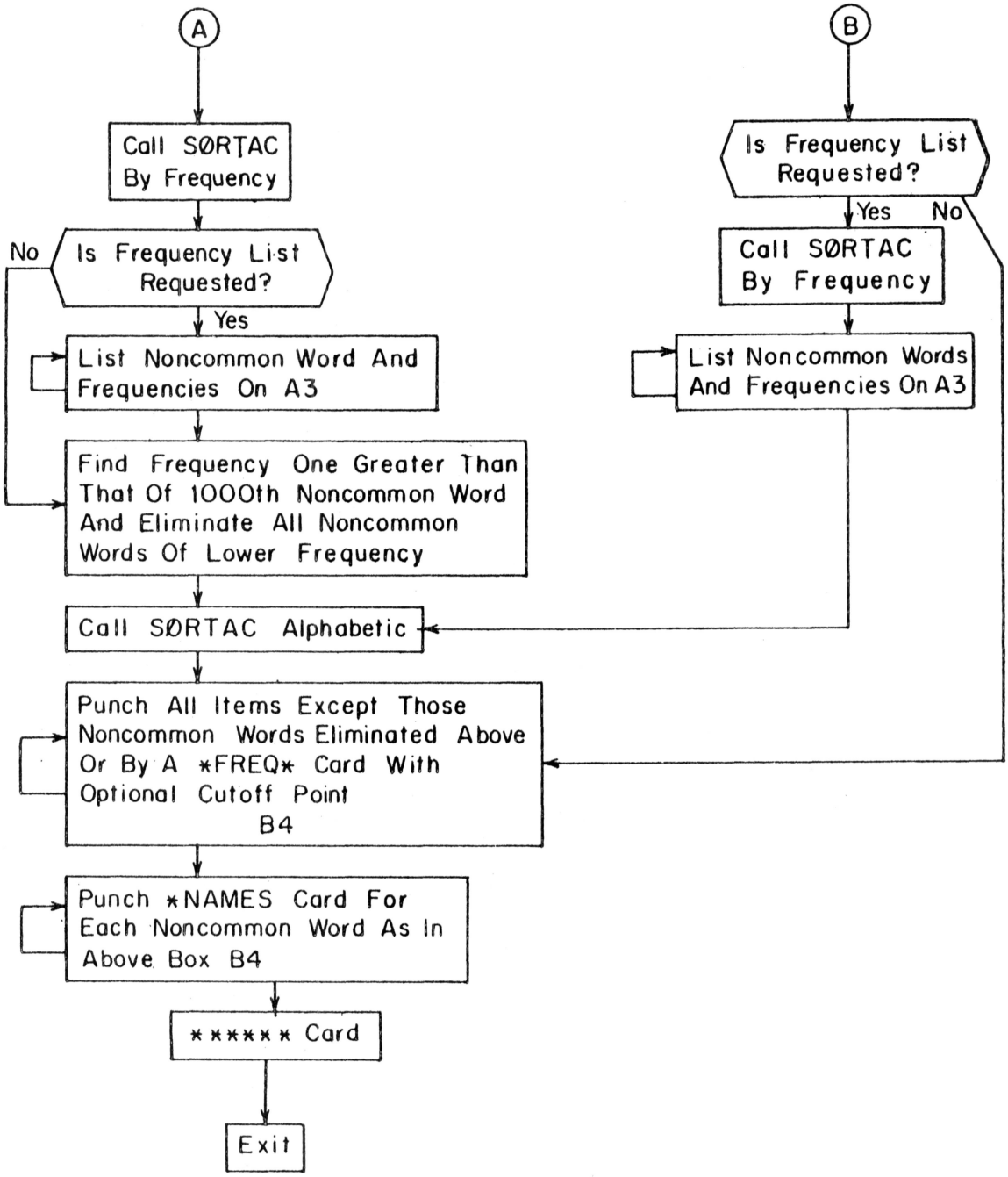
If more than 999 distinct noncommon stems are found, only those noncommon words whose frequency is greater than the frequency of the 1,000th most frequent noncommon word appears in the punched output. Also, if the optional number appears in columns 7-12 of the *FREQ* card, only those of the above words of frequency greater than or equal to this number are punched.

The written output on A3 consists of the following: a listing of all documents processed, a list of all noncommon word stems and their



Formation of Vacuous Thesaurus

Flowchart 3



Flowchart 3 (continued)

frequencies if a *FREQ* card was encountered, either a statement that the entire document collection has been processed or a count of the number of documents which have been handled, a statement of the least frequency a noncommon word may have in order to appear in the punched output, and a statement of the highest concept number assigned.

Concept numbers are assigned to noncommon words in alphabetical order starting with concept number 1. The first four 6-column fields of each of the first batch of cards contain the stem and suffix. The fifth 6-column field contains the concept number. Common words retain their original concept number, although some of these words may disappear because of apparent identities after suffixing.

Immediately following the cards containing the above data a set of cards is present having *NAMES in columns 1-6, and the first six columns of each suffixed noncommon word in columns 7-12. The last card has asterisks in columns 1-6. Punched output appears on B4.

This is a main program and not part of the SMART system proper. Flowchart 3 describes this program.