A SEARCH ALGORITHM AND DATA STRUCTURE FOR AN EFFICIENT INFORMATION SYSTEM

by

Shou-chuan Yang

Data and Computation Center
University of Wisconsin
Madison, Wisconsin

Abstract

This paper describes a system for information storage, retrieval, and updating, with special attention to the search algorithm and data structure demanded for maximum program efficiency. The program efficiency is especially warranted when a natural language or a symbolic language is involved in the searching process.

The system is a basic framework for an efficient information system. It can be implemented for text processing and document retrieval; numerical data retrieval; and for handling of large files such as dictionaries, catalogs, and personnel records, as well as graphic informations. Currently, eight commands are implemented and operational in batch mode on a CDC 3600: STORE, RETRIEVE, ADD, DELETE, REPLACE, PRINT, COMPRESS and LIST. Further development will be on the use of teletype console, CRT terminal, and plotter under a time-sharing environment for producing immediate responses.

The maximum program efficiency is obtained through a unique search algorithm and data structure. Instead of examining the recall ratio and the precision ratio at a higher level, this efficiency is measured in the most basic term of "average number of searches" required for looking up an item. In order to identify an item, at least one search is necessary even if it is found the first time. However, through the use of the hash-address of a key or keyword, in conjunction with an indirect-chaining list-structured table, and a large available space list, the average number of searches required for retrieving a certain item is 1.25 regardless of the size of the file in question. This is to be compared with 15.6 searches for the binary search technique in a 50,000-item file, and 5.8 searches for the letter-table method with no regard to file size.

---

*This study was supported in part by the National Science Foundation and the University of Wisconsin.

Best of all, since the program can use the same technique for storing and updating informations, the maximum efficiency is also applicable to them with the same ease. Thus, it eliminates all the problems of inefficiency caused in establishing a file, and in updating a file.

I. MOTIVATION

In our daily life, there are too many instances of looking for some type of information such as checking a new vocabulary in a dictionary, finding a telephone number and/or an address in a directory, searching a book of a certain author, title, or subject in a library catalog card file, etc. Before the desired information is found, one has to go through a number of items or entries for close examination. The quantitative measurement is usually termed as the "number of searches", "number of lookups", or "number of file accesses" in mechanized information systems.

However, as King pointed out in his article in the Annual Review of Information Science and Technology, volume 3, (pp.74-75) that the most common measures of accuracy of an information system are the recall ratio and precision ratio. These two measures have come under considerable criticism for their indifference in retrieval characteristics, being misleading and producing varying results. They probably should be used primarily to highlight a system's unsatisfactory performance. From the failure analysis of Hooper, King, Lancaster and others, the reasons are: incorrect query formulation, indexing errors, mechanical errors, incorrect screening, etc.

In the same volume (p. 139), Shoffner commented on the evaluation of systems that "it is important to be able to determine the extent to which file structures and search techniques influence the recall, precision, and other measures of system performance". Not until very recently, file structure and search techniques were apparently unpopular topics among information scientists except Salton and a few others. Nevertheless, these topics have been attacked constantly

by system scientists for a much smaller size of file but the maximum efficiency is a vital factor for the total system. They are frequently discussed under the title of "symbol table techniques", or "scatter storage techniques" as used by Morris as the title of his article. In addition to the "number of searches" and the "number of lookups" other terminologies used by the system scientist for referencing the most basic measure are the "number of probes", the "number of attempts", and the "search length".

Ever since 1964 the author stepping into the computer profession noticed that the efficiency of a file handling system is always crippled by its file searching technique no matter how sophisticated the system. This was especially the case during 1965 and 1966 when the author was employed at the Itek Corporation on an Air Force project of a Chinese to English machine translation experiment. The best search technique used for dictionary lookups was the binary search which is still considered one of the best techniques available today.

For a large file with a huge number of records, entries or items, the binary search technique will still yield a substantial number of searches which is a function of the file size. The typical files are: dictionaries of any sort, telephone directories, library catalog cards, personnel records, merchandise catalogs, document collections, etc. For example, in a 50,000-entry file system the average number of searches for finding an entry is 15.6 calculated as $\log_2 N$. This figure will not be very satisfactory if frequent search inquiries to a file are the case. As a result to finding better search techniques, at least three kinds of search techniques or algorithms are found to be more satisfactory than the binary search. Namely they are: Lamb and Jacobson's "Letter Table Method", Peterson's "Open-Addressing Technique", and Johnson's "Indirect Chaining Method". They have a rather interesting common feature that the file size is no longer a factor in the search efficiency.

II. EFFICIENCY OF VARIOUS SEARCH ALGORITHMS

In order to have a gross understanding of various search algorithms, six of them are examined and compared in respect to their search efficiencies.

1. Linear Search

This is also called sequential search or sequential scan. The linear search of an unordered list or file is the simplest one, but is inefficient because the average number of searches for a given entry in a N-entry file will be N/2. For example, if N = 50,000, the average number of searches for a given entry is an enormous 25,000. It is assumed that the probability of finding a given entry in the file is one. The average number of searches in a linear search is calculated as:

$$S = \frac{N + 1}{2} \qquad \text{or} \qquad S = \frac{N}{2}$$

if N is a large number.

The linear search has to be performed in a consecutive storage area and this sometimes causes certain inconvenience if the required storage area is very large. The inconvenience can be avoided by using the last computer word (or some bits of it) to index the location of the next section of storage area used and thus form a single chain for searching. This variation of the linear search method is called the single chain method. It differs from the linear search in storage flexibility but is otherwise the same in the efficiency.

2. Directory Search

This is also called block search. With the aid of a directory which contains the addresses of every Bth entry of the ordered file, a better result can be achieved because the average number of searches is greatly reduced. For the best result, choosing the blocking factor B = 220 in the example above, the answer is 223.6 searches which is calculated as:

$$S = \frac{N + B^2}{2B} \qquad \text{or} \qquad S = \frac{N}{2B} + \frac{B}{2}$$

3. Binary Search

Using the binary search method will yield a more satisfactory result. The search starts with the midpoint of the file, and goes to the midpoint of the associated remaining half if a match fails. The comparison of their values will decide which half of the file should be tried next time. This process will be repeated until a match is found. The average number of searches in the example is calculated through the following formula as 15.6 searches:

$S = \frac{N + 1}{N} \log_2(N+1) - 1$ or $S = \log_2 N$ if N is a large number.

The Hibbard's Double Chain Method and Sussenguth's Distributed Key Method are compatible to the binary search in search efficiency but have a much better update efficiency because of the list-structured address-chaining mechanism. The respective calculations of the example are:

Hibbard: $S = 1.4 \log_2 N = 21.9$ and

Sussenguth: $S = 1.24 \log_2 N = 19.4$

4. Letter Table Method

This attractive method as suggested by Lamb and Jacobson in 1961 for the dictionary lookup in a machine translation system did not receive good attention for its possible applications in general information systems. The reasons could be the immediate response to the numerous letter tables after the second level which indicated its inefficiency in storage, and that no clear search efficiency and update efficiency were expressed.

Suppose only the twenty-six English letters are involved, in theory there are twenty-six tables at the first level, 26^2

tables at the second level, 26^3 tables at the third level, etc. The number of letter tables will in practice be reduced drastically after the second level because of the actual limitation in letter combination in forming a vocabulary. However, no studies of this sort are available for the calculation of storage requirement to disprove its storage inefficiency.

The average number of searches or the expected search length of this method can not be calculated as a function of the file or dictionary size. It is simply the average number of letters or characters of a certain language plus one space character or any other delimiter. For the English language, it is a favorable 5.8 searches (S = 4.8 + 1), with no concern of the file size. Its update efficiency is compatible with its search efficiency and may be estimated at less than twice the average number of searches.

In order to achieve the above efficiency, the letter tables at each level should be structured in alphabetic order, and every letter should be converted into a numeric value such as A = 1, B = 2, C = 3, ... , Z = 26 and the space delimiter = 0 or 27 through a simple table-lookup procedure. Those converted values would then be used as the direct-access address within each subset of alphabetic letters at each letter-table level. This discards the need for binary search within each subset of "brothers" as in the cases of Hibbard's and Sussenguth's searches.

5. Open Addressing Method

As early as 1957, Peterson introduced this method for random access storage addressing. This method is also called linear probing. It assumes the existence of a certain hash function to transform the key or keyword of an entry into a numerical value within the range of the table size which is predetermined as 2^M for any integer value of M. The table size should be large enough to accomodate all the entries

of the file. As in other methods, this method also assumes the probability of finding an entry in the file is equal to one.

Under these two assumptions, and if a good hash function is selected for a balanced distribution of hash values, the open addressing method will resolve the situation if more than one key is mapped into a particular slot in the table, and yields a very attractive average number of searches in most of the cases. The algorithm is best described in Morris' phrases:

> "The first method of generating successive calculated addresses to be suggested in the literature was simply to place colliding entries as near as possible to their nominally allocated position, in the following sense. Upon collision, search forward from the nominal position (the initial calculated address), until either the desired entry is found or an empty space is encountered--searching circularly past the end of the table to the beginning, if necessary. If an empty space is encountered, that space becomes the home for the new entry."

Peterson did some simulations of open addressing by generating random numbers and storing them into a 500-entry table, and the result of the average number of searches from nine different runs is compared with the calculation obtained through Morris' formula or Salton's formula (L is the loading factor or the percentage of table fullness at the time of search):

Morris: $$S = \frac{1 - \frac{L}{2}}{1 + L}$$

Salton: $$S = \frac{L}{2(1-L)} + 1$$

Table 1. Average number of searches in open addressing

Loading factor	Peterson	Morris/Salton
0.1	1.053	1.056
0.2	1.137	1.125
0.3	1.23	1.214
0.4	1.366	1.333
0.5	1.541	1.5
0.6	1.823	1.75
0.7	2.26	2.167
0.8	3.223	3.0
0.9	5.526	5.5
1.0	16.914	∞

It is thus clear that unless the table is nearly full, the average number of searches will be surprisingly small. For example, if the loading factor is equal or less than 0.9 the average number of searches will be an amazing 1.965. This can be achieved by allowing an extra ten percent of the table size. In this case, its storage efficiency will become less attractive. However, its search efficiency and update efficiency are excellent due to its extremely low average number of searches.

6. Indirect Chaining Method

Since this method makes the same two assumptions as open addressing method and is heavily dependent upon the hash addressing, a more descriptive name for this method is suggested as Hash-Addressed Indirect-Chaining Search (HAICS). Other names found in the literature are scatter index tables, direct chaining (a variation in chaining structure), closed addressing (direct and indirect chaining), and virtual scatter tables (matching additional hashed bits).

The HAICS method uses a structured four-field table, an additional non-addressable overflow area of the table or a separate

overflow table, and a free storage area called the available space list. It is aimed to fully utilize all spaces reserved for the table before using the overflow area and the free storage area. This method treats the addressable table area as end-rounded, i.e., the first address of the table is considered following the last address. When overflow occurs, the nonaddressable overflow area is made available as an extended table area. This is so arranged to achieve better storage efficiency since in most cases there is no need for the additional overflow area and thus it can be omitted at the beginning or added on when the need arises.

The HAICS chaining table has four fields: keyword (key or data), index, link and pointer. The keyword field is usually one computer word in size for accomodating symbols which identifies the entry. The index field should have enough bits in size to specify the largest relative address in the available space list, so that the variable length entry stored in the available space list could be indexed from this table. The link field is used to indicate the linkage to the next table address where information of entries of the same hashed value can be found. The pointer field is designated to contain the address of the first entry of entries with the same hashed value. Both the link field and the pointer field should have a field length in bits large enough to store the largest relative address of the addressable table area, i.e., the size of the addressable table.

Entries are entered at their hashed addresses first, and then upon collision allotted to the next (or surrounded) empty addresses. Their pointers and links are set up for the proper chaining. When an entry is being looked up, the first step is to check the pointer of the entry at hashed address, then go to the pointed address and start searching from this beginning entry. If it is not found, the entry pointed by the link of current entry is searched until

it is found or there is no further link. The latter case indicates that there is no match for this search. When the entry is found through keyword identification, the address stored in the index field will direct the actual entry storage in the available space list. The index field and the available space list are needed only if the entries are of variable length so that storage space can be conserved. In the case of fixed length entries, the available space list is no longer needed and the index field in the table should be changed into an entry field with the desired fixed length.

A great advantage of the hash addressing is that to update entries in a file requires no sorting or resorting of any kind. In the HAICS method, to delete an entry is to follow the algorithm until the entry is retrieved, and then to hook up the next entry in the chain to the previous entry. All the storage previously occupied by this entry is freed for later use. To add an entry will use the same algorithm to retrieve the last entry in a particular chain and then to set up the linkage to the next empty space in the table and have the information of the added entry stored there. The added entry itself is stored in some free storage area in the available space list being indexed in the chaining table.

This method was first introduced in 1961 by Johnson and its average number of searches is calculated simply as:

$$S = 1 + \frac{L}{2}$$

More interesting yet, this formula is still valid when the loading factor L is greater than one which means the number of entries exceeds the allotted table size and the information of overflow entries are kept in the overflow area while entries themselves are again placed in the available space list. The cost of overflow is increased linearly at merely 0.5 searches per 100%

increase of overflow. This provision virtually eliminates the fear of overflow which frequently causes almost unmanageable difficulties and at very high expense.

Before the table is full as in the usual case, the average number of searches of the indirect chaining method is a hard-to-believe 1.25 with a maximum of 1.5 when the table is about full. It is at these two figures and the above-mentioned update efficiency and overflow advantage that the author believes some storage inefficiency and programming complexity should be tolerated painlessly.

Table 2. Average number of searches in indirect chaining

Loading factor	Johnson
0.1	1.05
0.2	1.1
0.3	1.15
0.4	1.2
0.5	1.25
0.6	1.3
0.7	1.35
0.8	1.4
0.9	1.45
1.0	1.5
1.5	1.75
2.0	2.0
3.0	2.5
4.0	3.0

An overview of various search algorithms discussed above is given in the following table:

Table 3. Comparison of various search algorithms

Search method	Average number of searches	Sample S (N=50,000)	Search efficiency	Update efficiency	Storage efficiency
1. Linear search	$\frac{N}{2}$	25,000	Poor	Good	Excellent
Single chain	$\frac{N}{2}$	25,000	Poor	Good	Excellent
2. Directory search	$\frac{N}{2B} + \frac{B}{2}$	223.6 (B = 220)	Average	Average	Excellent
3. Binary search	$\log_2 N$	15.6	Good	Poor	Good
Double chain (Hibbard)	$1.4 \log_2 N$ (optimal)	21.9	Good	Good	Average
Distributed key (Sussenguth)	$1.24 \log_2 N$ (optimal)	19.4	Good	Good	Average
4. Letter table method (Lamb and Jacobson)	Average number of letters per word plus one space	5.8 (English)	Excellent	Good	Poor
5. Open addressing (Peterson)	$\frac{L}{2(1-L)} + 1$	1.965 ($L \leq 0.9$)	Excellent	Excellent	Good
6. Indirect addressing (Johnson)	$\frac{L}{2} + 1$	1.25 ($L \leq 1.0$)	Excellent	Excellent	Average

For most search algorithms not included in the above table, they are variations and combinations of the linear search, single chain, directory search, binary search, double chain and distributed key aimed at the improvement of a certain efficiency. For example, the double chain method itself is a combination of binary search, a variation of single chain, and the linear search, and it is aimed to improve the update efficiency of the binary search.

III. KEYWORD CONSTRUCTION AND HASH FUNCTION

It is understood from the previous comparison of various search algorithms that the turning point for the excellent performance in search and update efficiencies is at the hash addressing which is essentially a simple procedure applying a certain hash function upon a search key or keywrod. Since the same keyword will always be hashed into the same hash value for table addressing, the criteria for a keyword selection or construction to identify an entry is the characteristic of uniqueness. And, in order to minimize the undesired collision upon hashed addresses, a good hash function should be selected such that it would yield a balanced distribution of hashed values within the range of the table size.

1. Keyword Construction

Under the consideration of the programming and computing efficiency and of the storage efficiency, usually a keyword of one computer word size is more desirable, e.g., eight-character keyword in a 48-bit word machine.

In machine files such as dictionaries, thesauruses, keyword indices, and merchandise catalogs, the keyword is almost readily available for hashing. If the keyword is longer than the allowable number of characters, a simple word truncation at the right end or some word compression schemes can be used to reduce the word size to a desired amount of number of characters.

For example, the standard word abbreviation, and a simple procedure to eliminate all the vowels and one of the two same consecutive consonants in a word will all be acceptable for this purpose.

In cases of author indices and catalogs, membership rosters, alphabetical telephone directories, taxation records, census, personnel records, student files, and any file using a person's name as the primary source of indexing will have the convenience in using the last name plus one space character and the initials as the keyword. The word compression scheme is certainly applicable if it is necessary.

When title indices and catalogs, subject indices and catalogs, business telephone directories, scientific and technical dictionaries, lexicons and idiom-and-phrase dictionaries, and other descriptive multi-word information are desired, the first character of each non-trivil word may be selected in the original word sequence to form a keyword. For example, the rather lengthy title of this paper may have a keyword as SADSIRS. Several known information systems are named exactly in this manner such as SIR (Raphael's Semantic Information Retrieval), SADSAM (Lindsay's Sentence Appraiser and Diagrammer and Semantic Analyzing Machine), BIRS (Vinsonhaler's Basic Indexing and Retrieval System), and CGC (Klein and Simmons' Computational Grammar Coder).

An alternative to meet the need of the multi-word situation but with a possible improvement in the uniqueness of the resulting hashed value is to perform some arithmetic or logical manipulation on the binary representation of the multi-word. When the multi-word is stored in consecutive computer words, each binary representation of a computer word is treated as an individual constant. Then either an arithmetic operation (e.g., ADD, SUBTRACT, MULTIPLY, and DIVIDE) or a logical operation (e.g., AND, OR) is to be performed on these computer words to collapse them into one single computer word as the

keyword. The resulting keyword from this kind of manipulation is not human readable but will serve its purpose for hash addressing.

In some cases where a unique number is assigned to an entry, there is no need to hash this number provided that number is inside the range of the allotted table size. This is mostly seen when a record or document is arranged in its accession number or location index. Otherwise the number can be treated as letters and be constructed in one of the methods described above.

2. Hash Function

The different functions used for random number generations can also serve as the hash function if a likely one-to-one relation can be established between the keyword and the resulting random number. This is also subject to the restriction that only numbers inside the range of table size are acceptable. Frequently this method will not give a balanced distribution of table addresses and thus affect the search and update efficiencies.

The arithmetic or logical manipulation described above for handling multi-word items can also be used as a hash function. One method called division hash code is suggested by Maurer that the binary representation of a keyword is treated as an integer and divided by the table size. The remainder of this division is thus inside the range of the table size and is used as the hash value. As Maurer noticed this method has the disadvantage that sometimes it does not produce indices which are equally distributed.

Three methods of computing hash addresses with proven satisfactory results were described very neatly by Morris:

"If the keys are names or other objects that fit into a single machine word, a popular method of generating a hash address from the key is to choose some bits from the middle of the square of the key--enough bits to be used as an index to address any item in the table. Since the value of the middle bits of the square depends on all of bits of the key, we can expect that different keys will give rise to different hash addresses with high probability, more or less independently of whether the keys share some common feature, say all beginning with the same bit pattern.

"If the keys are multiword items, then some bits from the product of the words making up the key may be satisfactory as long as care is taken that the calculated address does not turn out to be zero most of the time. The most dangerous situation in this respect is when blanks are coded internally as zeros or when partial word items are padded to full word length with zeros.

"A third method of computing a hash address is to cut the key up into N-bit sections, where N is the number of bits needed for the hash address, and then to form the sum of all of these sections. The low order N bits of the sum is used as the hash address. This method can be used for single-word keys as well as for multiword keys. ..."

All these three method assume one slight restriction that the size of the table has to be a power of two because of the binary bit selection. Personally the author prefers the first method of these three due to the extremely simple programming involved. Depending on different machines, the main operation requires about five machine language instructions: load A register with the keyword, integer multiply with the keyword, left shift A and Q registers X bits so that the desired bits is at the left end of the Q register, clear A register, left shift A & Q registers again Y bits so that the desired bits are resided at the right end of the A register (CDC 3600 COMPASS).

If the second method described by Morris is used, the keyword construction for multi-word item can be eliminated if there is no risk of the kind described. The third method is more interesting because it has the generality of accepting both single-word and multi-word items but at a slight cost of some more programming which is to be offset by the cost of multi-word construction.

IV. HAICS DATA STRUCTURE

In response to the needs of search and update efficiencies, the data structure for the HAICS technique has to be organized in a much more sophisticated way with some additional storage requirement over the entries themselves. As previously described under the Indirect Chaining Method, it requires a fixed-size four-field addressable chaining table for bookkeeping the keyword and all the information for the chaining mechanism, a reserved free storage area called the available space list for storing the variable-length entries themselves, and an added-on non-addressable overflow table area for overflow chainings.

The overall HAICS data structure is quite list oriented but it is packed into the form of arrays for a more efficient indexing and searching procedure. A test program has been written in CDC 3600 Fortran (a variation of Fortran IV) for the convenience of adapting to other computers. The discussions following will frequently refer to the Fortran language and the list structure for a better clarification.

1. The Chaining Table

The four-field table can be easily set up as four single-dimensional arrays or as a four-dimensional array at the cost of several wasted bits in the computer word for storing the index to the available space list, the link to the next table address, or the pointer to indicate the beginning of a chained

sub-list. The savings are less computer word-packing and unpacking operations.

The positions of each four-field table item in relation to the first item,i.e., the hash addresses, can be viewed as the main list of the table. The linked entries of the same hashed address are treated as a sub-list. Since the relative position of each table item is identified with its hashed address, there is no need to set up an address index for each item. Besides, since each entry can be hash-addressed with an average number of searches at 1.25 and that most chains are not much longer than one or two entries, it is not necessary to have a backward link within a chain.

The layout of the chaining table is shown in Table 4 with some sample linkages indicated in hand-drawn circles and arrows.

Table 4. The chaining table

THE CHAINING TABLE --

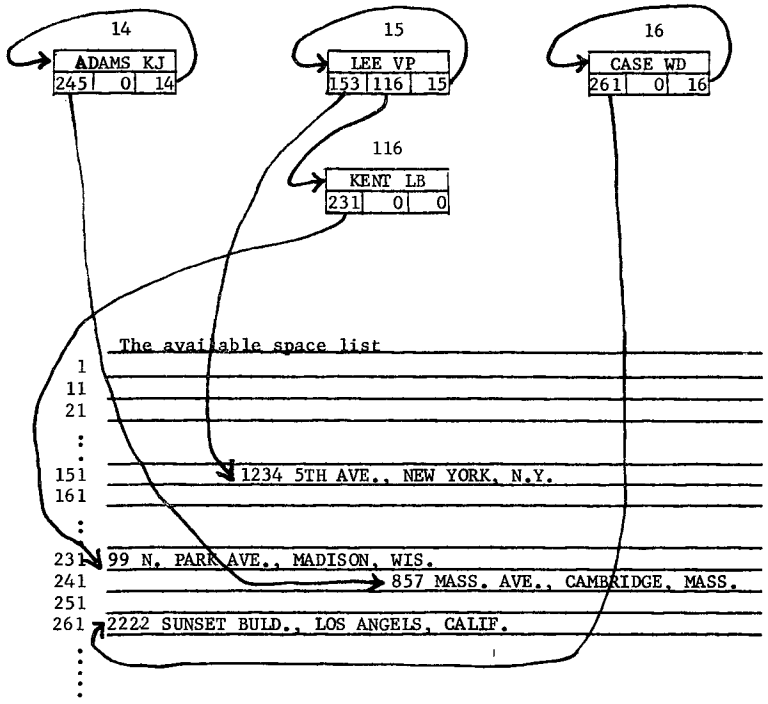
VALUE	KEYWORD	INDEX	LINK	POINTER
1	EARPHONE	296	0	1
2	ABSOCOD	23	17	2
3	EMERS R	377	0	0
4	EMERS HL	349	0	4
5	DASH	222	0	5
6		0	0	0
7		0	0	0
8	EAM	291	0	8
9	DATACIRC	278	0	9
10		0	0	0
11		0	0	0
12	BACKNOIS	99	13	12
13	CABINET	151	0	14
14	EMERS HS	335	0	0
15	EMERY DF	401	0	16
16	MACI	415	0	0
17	MAOIS AM	424	0	19
18	EMS	306	0	18
19	MASER	430	0	0
20	EMERS I	364	21	20
21	EMERS RE	376	22	0
22	MAS	407	0	0
23	DIADPD	1	25	23
24	ABSOADDR	11	28	24
25	ABSORBTN	37	0	32
26	BAFFLE	119	0	26
27		0	0	31
28	EMERS AT	315	0	0
29	EMERS CB	421	0	3
30	BABBLE	68	0	30
31	EMERS G	329	0	15
32	EMERS HC	357	0	0

The four-field table items can also be viewed as four-field notes or cells in a list structured presentation:

keyword		
index	link	pointer

An example of the list structured presentation of the HAICS system can be illustrated as follows:

Figure 1. List presentation of the HAICS data structure



2. The Available Space List

This can be a single one-dimensional array for the best storage efficiency in accomodating variable length entries. The beginning of an entry is indexed in the index field of the chaining table the relative position of the beginning computer word in the available space list. This is also shown in the examples of Figure 1. The ending of an entry can either be indicated by a special symbol such as two consecutive blanks or EOE as the abbreviation of end-of-entry, or calculated as the entry length and placed at the beginning of an entry.

Multi-dimensional arrays are usually wasteful for storing variable length entries. If the entries are of fixed length then, as described before, there is no need to have a separate available space list. These fixed length entries can be put directly in the enlarged index field of the chaining table for a more efficient processing.

3. The Overflow Table

The overflow table is structured the same way as the Chaining table except that it is not accessable through the hash function. It serves as an emergency device only after the chaining table is fully utilized and additional storage area is available at that time. When the overflow table is established to meet the emergency, its array names and the size are made available to the HAICS procedure as an extended area for the chaining table.

V. HAICS ALGORITHMS FOR STORING, RETRIEVING, UPDATING, AND UTILITY FUNCTIONS

The logical procedure of the HAICS technique is described in algorithms for easy adaption to procedure-oriented languages such as Fortran and Algol. Currently, eight commands have been implemented and operational on the CDC 3600 test program: STORE, RETRIEVE, ADD, DELETE, REPLACE, PRINT, COMPRESS, and LIST. They can be functionally classified into three groups: the main algorithms for STORE and RETRIEVE; the updating algorithms for ADD, DELETE, and REPLACE; and the utility algorithms for PRINT, COMPRESS, and LIST.

The two main algorithms are frequently utilized by other algorithms except PRINT.

These algorithms are presented in detail in the following:

1. Algorithm STORE (S)

This algorithm is to be used for establishing a HAICS file at the very beginning. It is assumed that the arrays for the chaining table and the available space list have been set up properly, and that keywords and the hash function have been constructed or selected appropriately through out the subsequent uses of this file.

S1 Clear the arrays for the Chaining Table and the Available Space List, and set up proper indices for these tables

S2 Compute the hash value of the given keyword, I = HASH(KEY)

S3 If POINTER (I) = 0 and KEYWORD (I) = 0, then KEYWORD (I) = KEY

S4 The first available address in the Available Space List, J, for storing the entry is placed as INDEX (I) = J

S5 The entry is stored in the Available Space List (ASL) sequentially starting at ASL(J) and with a special symbol EOE placed at the end of the entry in ASL, and exit on success.

S6 If POINTER (I) = 0 and KEYWORD (I) $\neq$ 0, then

S7 Search the keyword array downward and end-round until a KEYWORD (I) = 0 is found

S8 Set I = POINTER (I), KEYWORD (I) = KEY, and go to Step S4

S9 If a KEYWORD (I) = 0 can not be gound in the keyword array, a message is given to indicate the overflow of the Chaining Table and then exit on failure.

S10 If POINTER (I) $\neq$ 0, then I = POINTER (I)

S11 If LINK (I) = 0, then go to Step S7 for a KEYWORD (I) = 0, upon failure, go to Step S9; upon success, go to Step S12

S12 Set LINK (I) = I, KEYWORD (I) = KEY, and go to Step S4

S13 If LINK (I) $\neq$ 0, then set I = LINK (I), go to Step S11

S14 Repeat for additional entries starting at Step S2.

Examples in Tables 5 and 6 show the result of several stored entries under this algorithm.

Table 5. The chaining table after a STORE command

THE CHAINING TABLE --

VALUE	KEYWORD	INDEX	LINK	POINTER
1	EARPHONE	295 → A	0	1
2	ABSOCOD	23	0	2
3	EMERS R	377	0	0
4	EMERS HL	349	0	4
5	DASH	222	6	5
6	DATA	243	0	10
7	EMERS HH	342	0	7
8	EAM	291	0	8
9	DATACIRC	274	0	9
10	EAEC	385	0	0
11	EMERY DJ	395	0	0
12	BACKNOIS	93 → B	13	12
13	CABINET	151 → C	0	14
14	EMERS HS	335	0	0
15	EMERY DF	401	0	0
16		0	0	0
17		0	0	0
18	EMS	306	0	18
19		0	0	0
20	EMERS I	364	21	20
21	EMERS RE	378	0	0
22		0	0	0
23	UTADPD	1 → D	25	23
24	ABSOAUDR	11 → E	28	24
25	ABSORBTN	37	0	32
26	BAFFLE	119	0	26
27	CABLE	195	31	27
28	EMERS AT	315 → F	29	11
29	EMERS CB	321	0	0
30	BABBLE	68	0	30
31	EMERS G	329	0	15
32	EMERS HC	357 → G	0	0

Table 6. The available space list after a STORE command

```
THE AVAILABLE SPACE LIST --
ADDRESS                    INFORMATION
    1 (D)-> DATA TRANSMISSION AND DATA PROCESSING DICTIONARY. BY JAMES F. HO
    9 LMES     EOE (E)-> (ABSOLUTE ADDRESS) THE IDENTIFICATION OF A SPECIF
   17 IC REGISTER OR LOCATIPN IN STORAGE.     EOE     (ABSOLUTE CODING
   25 )CODING IN WHICH ALL ADDRESSES REFER TO SPECIFIC MACHINE REGISTE
   33 RS AND MEMORY LOCATIONS.EOE     (ABSORPTION) THE LOSS OF ENERGY I
   41 N THE TRANSMISSION OF WAVES OVER RADIO OR WIRE PATHS DUE TO CON
   49 VERSION INTO HEAT OR OTHER FORMS OF ENERGY.  IN WIRE TRANSMISSIO
   57 N, THE TERM IS USUALLY APPLIED ONLY TO LOSS OF ENERGY INEXTRANEO
   65 US MEDIA.       EOE     (1). AGGREGATE CROSSTALK FROM A LARGE NU
   73 MBER OF DISTURBING CHANNELS.     (2). UNWANTED DISTURBING SOUND
   81 S IN A CARRIER OR OTHER MULTIPLE CHANNEL SYSTEM WHICH RESULT FRO
   89 M THE AGGREGATE CROSSTALK OR MUTUAL INTERFERENCE FROM OTHER CHAN
   97 NELS.   EOE (B)-> (BACKGROUND NOISE) THE TOTAL SYSTEM NOISE INDEPEN
  105 DENT OF THE PRESENCE OR ABSENCE OF A SIGNAL.  THE SIGNAL IS NOT
  113 TO BE INCLUDED AS PART OF THE NOISE.     EOE     A SHIELDING STRU
  121 CTURE OR PARTITION USED TO INCREASE THE EFFECTIVE LENGTHOF THE E
  129 XTERNAL TRANSMISSION PATH BETWEEN TWO POINTS AS FOR TRANSMISSION
  137 PATH BETWEEN TWO POINTS AS FOR EXAMPLE, BETWEEN THE FRONT AND TH
  145 E BACK OF AN ELECTROACOUSTIC TRANSDUCER.EOE (C)-> EQUIPMENT--CASE
  153 DESIGNED TO HOUSE RELAYS AND OTHER APPARATUS.     KEY--CASE INST
  161 ALLED ON A CUSTOMER'S PREMISES, TO PERMIT DIFFERENT LINES TO THE
  169  CENTRAL OFFICE TO BE CONNECTED TO VARIOUS TELEPHONE STATIONS.
  177 IT HAS SIGNALS TO INDICATE ORIGINATING CALLS AND BUSY LINES.
  185  TEST--BOX CONTAINING APPARATUS FOR TROUBLE LOCATION AND ROUTINE
  193  MAINTENANCE.     EOE     AN ASSEMBLY OF ONE OR MORE CONDUCTORS, U
  201 SUALLY WITHIN AN ENVELOPING PROTECTIVE SHEATH, IN SUCH STRUCTURA
  209 L ARRANGEMENT OF THE INDIVIDUAL CONDUCTORS AS WILL PERMIT OF THE
  217 IR USE SEPARATELY OR IN GROUPS.  EOE     THREE UNIT LENGTHS OF SU
  225 STAINED SIGNAL, WHEN TRANSMITTED, A DASH WILL AUTOMATICALLY BE F
  233 OLLOWED BY ONE UNIT LENGTH OF SILENCE.  TERM USED IN RADIOTELEGR
  241 APHY.  EOE     (1). PLURAL TERM COLLECTIVELY USED TO DESIGNATE
  249 MATERIAL SERVING AS A BASIS FOR DISCUSSION,. MATERIAL MAY OR MAY
  257  NOT BE TECHNICAL IN NATURE.  THE SINGULAR OF DATA IS DATUM.  (2
  265 ). INFORMATION, PARTICULARLY THAT USED AS A BASIS FOR MECHANICAL
  273  OR ELECTRONIC COMPUTERS.     EOE     (DATA CIRCUIT) COMMUNICAT
  281 ION FACILITY PERMITTING TRANSMISSION OF INFORMATION IN DIGITAL F
  289 ORM.   EOE     ELECTRICAL ACCOUNTING MACHINE.  EOE (A)-> AN ELECT
  297 ROACOUSTIC TRANSDUCER INTENDED TO BE CLOSELY COUPLED ACOUSTICALL
  305 Y TO THE EAR.  EOE     (EMERGENCY MEDICAL SERVICE) 25 W MAIN,255
  313 -8567.  EOE (F)-> (EMERSON A T) 1008 RUTLEDGE,256-2587.     EOE
  321 (EMERSON CHAS B) 5314 MATHEWS RD MIDDLETON,238-5776.     EOE
  329 (EMERSON GAIL) 221 LAKE LAWN PL,257-6916.EOE     (EMERSON HARLAND
  337  S) 534 HILLTOP DR,233-0632.     EOE     (EMERSON HARRY H JR) 4626
  345  FSCH LA,249-3448.     EOE     (EMERSON HARRY L MRS) 2314 CHALET
  353  GARDENS RD,238-6067.  EOE (G)-> (EMERSON HUGH C) 1004 TOMPKINS DR
  361 ,222-1438.     EOE     (EMERSON IDA MRS) 419 JEAN,256-8128.
  369 EOE     (EMERSON RICHARD E) 1610 CAMERON DR,238-1126.     EOE
  377 (EMERSON RODNEY) 6266 ELMWOOD AV MIDDLETON,238-5769.     EOE
  385 (EMERY AIR FREIGHT CORP) 5300 S HOWELL AV MILW--MDSN NO.,255-8312
  393 .     EOE     (EMERY DANIEL J) 522 STATE,256-3305.     EOE
  401 (EMERY DONA F) 2506 MCDIVITT RD,256-1284.EOE
  409
  417
  425
  433
```

2. Algorithm RETRIEVE (R)

With a given keyword, this algorithm will retrieve the entry which is associated with keyword under the same assumptions made in Algorithm STORE.

R1 Compute I = HASH(KEY)

R2 If POINTER(I) = 0, exit on failure.

R3 If POINTER(I) ≠ 0, then I = POINTER(I)

R4 If KEYWORD(I) = KEY, then J = INDEX(I), move the entry in ASL starting from ASL(J) to a working area until an EOE is encountered, and exit on success.

R5 If KEYWORD(I) ≠ KEY, and if LINK(I) = 0, then exit on failure.

R6 If KEYWORD(I) ≠ KEY, and if LINK(I) ≠ 0, then I = LINK(I), go to step R4.

R7 Repeat for additional entries starting at Step R1.

Examples in Tables 5 and 6 will also illustrate this algorithm in actual applications. The execution of the RETRIEVE command will not change the contents of the Chaining Table and the Available Space List in any event.

3. Algorithm ADD (A)

This algorithm is used when an additional or new entry is put into the already established HAICS file. It is an operation of "adding" an entry to the end of a chain of its hashed address, rather than breaking up the chain and "inserting" the entry according to some order or hierarchy. This is so because each chain in the HAICS file is mostly very short with only one or two entries and the "inserting" will gain very little in search and update efficiencies.

This algorithm is different from Algorithm STORE in that no clear-up operations are performed on the arrays of the Chaining Table and the Available Space List. In addition, a relative address in the Available Space List is accepted as the first available address to store the added entries themselves.

A1	Set J = the given first available address
A2	If J $\leq$ size of ASL, go to Step S2 in Algorithm STORE and return to Step A4 upon exit from Algorithm STORE
A3	If J > size of ASL, exit on failure.
A4	Repeat for additional entries starting at Step A2

Tables 7 and 8 exhibit the results of the ADD command of some new entries upon Tables 5 and 6.

Table 7. The chaining table after an ADD command

THE CHAINING TABLE --

VALUE	KEYWORD	INDEx	LINK	POINTER
1	EARPHONE	296	0	1
2	ABSOCOD	23	17	2 ←3rd
3	EMERS R	377	0	0
4	EMERS HL	349	0	4
5	DASH	222	6	5
6	DATA	243	0	10
7	EMERS HH	342	0	7
8	EAM	291	0	8
9	DATACIRC	278	0	9
10	EAFC	385	0	0
11	EMERY DJ	395	0	0
12	BACKNOIS	99	13	12
13	CABINET	151	0	14
14	EMERS HS	335	0	0
15	EMERY DF	401 → B	0	16 ←2nd
16	MACI	415	0	0
17	MADIS AM	424 → C	0	19 ←4th
18	EMS	308	0	18
19	MASER	430 → D	0	0
20	EMERS I	364	21	20 ←1st
21	EMERS RE	370	22	0
22	MAS	407 → A	0	0
23	DTADPD	1	25	23
24	ABSOADDR	11	28	24
25	ABSORBTN	37	0	32
26	BAFFLE	119	0	26
27	CABLE	196	31	27
28	EMERS AT	315	29	11
29	EMERS CB	321	0	3
30	BABBLE	68	0	30
31	EMERS G	329	0	15
32	EMERS HC	357	0	0

Table 8. The available space list after an ADD command

```
THE AVAILABLE SPACE LIST --
ADDRESS                          INFORMATION

    1 DATA TRANSMISSION AND DATA PROCESSING DICTIONARY, BY JAMES F. HO
    9 LMES     EOE      (ABSOLUTE ADDRESS)THE IDENTIFICATION OF A SPECIF
   17 IC REGISTER OR LOCATIPN IN STORAGE.     EOE      (ABSOLUTE CODING
   25 )CODING IN WHICH ALL ADDRESSES REFER TO SPECIFIC MACHINE REGISTE
   33 RS AND MEMORY LOCATIONS.EOE      (ABSORPTION)THE LOSS OF ENERGY I
   41 N THE TRANSMISSION OF WAVES OVER RADIO  OR WIRE PATHS DUE TO CON
   49 VERSION INTO HEAT OR OTHER FORMS OF ENERGY.  IN WIRE TRANSMISSIO
   57 N. THE TERM IS USUALLY APPLIED ONLY TO LOSS OF ENERGY INEXTRANEO
   65 US MEDIA.       EOE      (1). AGGREGATE CROSSTALK FROM A LARGE NU
   73 MBER OF DISTURBING CHANNELS.      (2). UNWANTED DISTURBING SOUND
   81 S IN A CARRIER OR OTHER MULTIPLE CHANNEL SYSTEM WHICH RESULT FRO
   89 M THE AGGREGATE CROSSTALK OR MUTUAL INTERFERENCE FROM OTHER CHAN
   97 NELS.   EOE      (BACKGROUND NOISE)THE TOTAL SYSTEM NOISE INDEPEN
  105 DENT OF THE PRESENCE OR ABSENCE OF A SIGNAL.  THE SIGNAL IS NOT
  113 TO BE INCLUDED AS PART OF THE NOISE.     EOE      A SHIELDING STRU
  121 CTURE OR PARTITION USED TO INCREASE THE EFFECTIVE LENGTHOF THE E
  129 XTERNAL TRANSMISSION PATH BETWEEN TWO POINTS AS FOR TRANSMISSION
  137 PATH BETWEEN TWO POINTS AS FOR EXAMPLE, BETWEEN THE FRONT AND TH
  145 E BACK OF AN ELECTROACOUSTIC TRANSDUCER.EOE      EQUIPMENT--CASE
  153 DESIGNED TO HOUSE RELAYS AND OTHER APPARATUS.      KEY--CASE INST
  161 ALLED ON A CUSTOMER#S PREMISES, TO PERMIT DIFFERENT LINES TO THE
  169  CENTRAL OFFICE TO BE CONNECTED TO VARIOUS TELEPHONE STATIONS.
  177 IT HAS SIGNALS TO INDICATE ORIGINATING CALLS AND BUSY LINES.
  185  TEST--BOX CONTAINING APPARATUS FOR TROUBLE LOCATION AND ROUTINE
  193  MAINTENANCE.   EOE      AN ASSEMBLY OF ONE OR MORE CONDUCTORS, U
  201 SUALLY WITHIN AN ENVELOPING PROTECTIVE SHEATH, IN SUCH STRUCTURA
  209 L ARRANGEMENT OF THE INDIVIDUAL CONDUCTORS AS WILL PERMIT OF THE
  217 IR USE SEPARATELY OR IN GROUPS. EOE      THREE UNIT LENGTHS OF SU
  225 STAINED SIGNAL, WHEN TRANSMITTED, A DASH WILL AUTOMATICALLY BE F
  233 OLLOWED BY ONE UNIT LENGTH OF SILENCE.  TERM USED IN RADIOTELEGR
  241 APHY.    EOE      (1). PLURAL TERM COLLECTIVELY USED TO DESIGNATE
  249 MATERIAL SERVING AS A BASIS FOR DISCUSSION,. MATERIAL MAY OR MAY
  257  NOT BE TECHNICAL IN NATURE.  THE SINGULAR OF DATA IS DATUM.  (2
  265 ). INFORMATION, PARTICULARLY THAT USED AS A BASIS FOR MECHANICAL
  273  OR ELECTRONIC COMPUTERS.      EOE      (DATA CIRCUIT)COMMUNICAT
  281 ION FACILITY PERMITTING TRANSMISSION OF INFORMATION IN DIGITAL F
  289 ORM.    EOE      ELECTRICAL ACCOUNTING MACHINE.  EOE      AN ELECT
  297 ROACOUSTIC TRANSDUCER INTENDED TO BE CLOSELY COUPLED ACOUSTICALL
  305 Y TO THE EAR.    EOE      (EMERGENCY MEDICAL SERVICE)25 W MAIN,255
  313 -8567.  EOE      (EMERSON A T)1008 RUTLEDGE,256-2587.      EOE
  321 (EMERSON CHAS B)5314 MATHEWS RD MIDDLETON,238-5776.      EOE
  329 (EMERSON GAIL)220 LAKE LAWN PL,257-6916.EOE      (EMERSON HARLAND
  337  S)534 HILLTOP DR,233-0632.      EOE      (EMERSON HARRY H JR)4626
  345  ESCH LA,249-3448.      EOE      (EMERSON HARRY L MRS)2314 CHALET
  353  GARDENS RD,238-6067.   EOE      (EMERSON HUGH C)1004 TOMPKINS DR
  361 ,222-1432.      EOE      (EMERSON IDA MRS)419 JEAN,256-8128.
  369 EOE      (EMERSON RICHARD E)1610 CAMERON DR,238-1126.      EOE
  377 (EMERSON RODNEY)6266 ELMWOOD AV MIDDLETON,238-5769.      EOE
  385 (EMERY AIR FREIGHT CORP)5300 S HOWELL AV MILW--MDSN NO.,255-6312
  393 .      EOE      (EMERY DANIEL J)522 STATE,256-3305.      EOE
  401 (EMERY DONA F)2506 MCDIVITT RD,256-1284.EOE      (A)→(MADISON ADJUSTM
  409 ENT SYSTEM)303 PRICE PL, 238-2616.      EOE(B)→(MADISON ACCEPTA
  417 NCE CORP INC)1201 W BELTLINE HY, 257-1091.      EOE(C)→(MADIS A
  425 LMA MRS)3256 MILW, 244-7831.    EOE(D)→(MADISON AIR SERVICE)330
  433 2 NORTH STOUGHTON RD, 249-6478. EOE
```

4. Algorithm DELETE (D)

This is used to delete an entry from the HAICS file. This algorithm is heavily dependent upon the Algorithm RETRIEVE but instead of just retrieving the entry it traces back and deletes the entry itself from the Available Space List and all the pertinate information in the Chaining Table.

D1 Go to Step R1 in Algorithm RETRIEVE, return to Step D2 upon exit on failure from Algorithm RETRIEVE, or return to Step D3 upon exit on success from Algorithm RETRIEVE

D2 Exit on failure.

D3 Clear up the occupied section of the entry in ASL including the special symbol EOE at the end of the entry

D4 INDEX(I) = 0 and KEYWORD(I) = 0

D5 If POINTER(I) = I and LINK(I) = 0, then POINTER(I) = 0, exit on success.

D6 If POINTER(I) = I and LINK(I) $\neq$ 0, then POINTER(I) = LINK(I), LINK(I) = 0, exit on success.

D7 If POINTER(I) $\neq$ I and LINK(I) = 0, then trace back the previous link which contains I and set it to zero when it is found, otherwise trace back the original pointer and set it to zero, exit on success.

D8 If POINTER(I) $\neq$ I and LINK(I) $\neq$ 0, then trace back the previous link which contains I and replace it with LINK(I), exit on success.

D9 Repeat for additional entries starting at Step D1.

Sample results of the DELETE command executed upon Tables 7 and 8 are shown in Tables 9 and 10.

Table 9. The chaining table after a DELETE command

THE CHAINING TABLE --

VALUE	KEYWORD	INDEX	LINK	POINTER
1	EARPHONE	296	0	1
2	ABSOCOD	23	17	2
3	EMERS R	377	0	0
4	EMERS HL	349	0	4
5	DASH	222	0 (6)	5
6	DATA	243 (E)	0	0
7	EMERS HH	342 (C)	0	0
8	EAM	291	0	8
9	DATACIRC	274	0	9
10	EAFC	385 (A)	0	0
11	EMERY DJ	395 (B)	0	0
12	BACKNOIS	99	13	12
13	CABINET	151	0	14
14	EMERS HS	335	0	0
15	EMERY DF	401	0	16
16	MACI	415	0	0
17	MADIS AM	424	0	19
18	EMS	306	0	18
19	MASER	431	0	0
20	EMERS I	364	21	20
21	EMERS RE	375	22	0
22	MAS	407	0	0
23	DTADPD	1	25	23
24	ABSOADDR	11	28	24
25	ABSORBTN	37	0	32
26	BAFFLE	119	0	26
27	CABLE	196 (D)	0	31
28	EMERS AT	315	0	0
29	EMERS CB	321	0	3
30	BABBLE	68	0	30
31	EMERS G	329	0	15
32	EMERS HC	357	0	0

DATA
10
EAFC
7
EMERS HH
CABLE
27
11
EMERY DJ

Table 10. The available space list after a DELETE command

```
THE AVAILABLE SPACE LIST --
ADDRESS                    INFORMATION

    1 DATA TRANSMISSION AND DATA PROCESSING DICTIONARY, BY JAMES F. HO
    9 LMES     EOE      (ABSOLUTE ADDRESS)THE IDENTIFICATION OF A SPECIF
   17 IC REGISTER OR LOCATIPN IN STORAGE.     EOE      (ABSOLUTE CODING
   25 )CODING IN WHICH ALL ADDRESSES REFER TO SPECIFIC MACHINE REGISTE
   33 RS AND MEMORY LOCATIONS.EOE     (ABSORPTION)THE LOSS OF ENERGY I
   41 N THE TRANSMISSION OF WAVES OVER RADIO OR WIRE PATHS DUE TO CON
   49 VERSION INTO HEAT OR OTHER FORMS OF ENERGY.  IN WIRE TRANSMISSIO
   57 N, THE TERM IS USUALLY APPLIED ONLY TO LOSS OF ENERGY INEXTRANEO
   65 US MEDIA.      EOE      (1). AGGREGATE CROSSTALK FROM A LARGE NU
   73 MBER OF DISTURBING CHANNELS.     (2). UNWANTED DISTURBING SOUND
   81 S IN A CARRIER OR OTHER MULTIPLE CHANNEL SYSTEM WHICH RESULT FRO
   89 M THE AGGREGATE CROSSTALK OR MUTUAL INTERFERENCE FROM OTHER CHAN
   97 NELS.    EOE      (BACKGROUND NOISE)THE TOTAL SYSTEM NOISE INDEPEN
  105 DENT OF THE PRESENCE OR ABSENCE OF A SIGNAL.  THE SIGNAL IS NOT
  113 TO BE INCLUDED AS PART OF THE NOISE.     EOE      A SHIELDING STRU
  121 CTURE OR PARTITION USED TO INCREASE THE EFFECTIVE LENGTHOF THE E
  129 XTERNAL TRANSMISSION PATH BETWEEN TWO POINTS AS FOR TRANSMISSION
  137 PATH BETWEEN TWO POINTS AS FOR EXAMPLE, BETWEEN THE FRONT AND TH
  145 E BACK OF AN ELECTROACOUSTIC TRANSDUCER.EOE      EQUIPMENT--CASE
  153 DESIGNED TO HOUSE RELAYS AND OTHER APPARATUS.     KEY--CASE INST
  161 ALLED ON A CUSTOMER#S PREMISES, TO PERMIT DIFFERENT LINES TO THE
  169  CENTRAL OFFICE TO BE CONNECTED TO VARIOUS TELEPHONE STATIONS.
  177 IT HAS SIGNALS TO INDICATE ORIGINATING CALLS AND BUSY LINES.
  185  TEST--BOX CONTAINING APPARATUS FOR TROUBLE LOCATION AND ROUTINE
  193  MAINTENANCE.    EOE
  201
  209
  217                                                 THREE UNIT LENGTHS OF SU
  225 STAINED SIGNAL, WHEN TRANSMITTED, A DASH WILL AUTOMATICALLY BE F
  233 OLLOWED BY ONE UNIT LENGTH OF SILENCE.  TERM USED IN RADIOTELEGR
  241 APHY.    EOE
  249
  257
  265
  273                                                 (DATA CIRCUIT)COMMUNICAT
  281 ION FACILITY PERMITTING TRANSMISSION OF INFORMATION IN DIGITAL F
  289 ORM.     EOE      ELECTRICAL ACCOUNTING MACHINE.  EOE      AN ELECT
  297 ROACOUSTIC TRANSDUCER INTENDED TO BE CLOSELY COUPLED ACOUSTICALL
  305 Y TO THE EAR.    EOE      (EMERGENCY MEDICAL SERVICE)25 W MAIN,255
  313 -8567.  EOF      (EMERSON A T)1008 RUTLEDGE,256-2587.     EOE
  321 (EMERSON CHAS H)5314 MATHEWS RD MIDDLETON,238-5776.     EOE
  329 (EMERSON GAIL)220 LAKE LAWN PL,257-6916.EOE      (EMERSON HARLAND
  337  S)534 HILLTOP DR,233-0632.     EOE
  345                                                 (EMERSON HARRY L MRS)2314 CHALET
  353  GARDENS RD,238-6067.    EOE      (EMERSON HUGH C)1004 TOMPKINS DR
  361 ,222-1436.      EOE      (EMERSON IDA MRS)419 JEAN,256-8129.
  369 EOE      (EMERSON RICHARD E)1610 CAMERON DR,238-1126.     EOE
  377 (EMERSON RODNEY)6266 ELMWOOD AV MIDDLETON,238-5769.     EOE
  385
  393
  401 (EMERY DONA F)2506 MCDIVITT RD,256-1284.EOE      (MADISON ADJUSTM
  409 ENT SYSTEM)303 PRICE PL, 238-2616.     EOE      (MADISON ACCEPTA
  417 NCE CORP INC)1201 W BELTLINE HY, 257-1091.     EOE      (MADIS A
  425 LMA MRS)3256 MILW, 244-7831.    EOE      (MADISON AIR SERVICE)330
  433 2 NORTH STOUGHTON RD, 249-6478. EOE
```

(D) CABLE

(E) DATA

(A) EAFC (B) EMERY DJ (C) EMERS HH

5. Algorithm REPLACE (RP)

This is for the replacement of an entry itself in the Available Space List with the same keyword and linkages in the Chaining Table unchanged. Replacement entries longer than the original entries can be treated in a few different ways. The current algorithm will truncate the excessive end and give a message to indicate the situation. A remedy if desired then can be made through the deletion of the incomplete entry and the addition of the complete entry as a new entry. This algorithm will make use of the Algorithms RETRIEVE and STORE to find the desired entry and then replace the old contents with the new contents in the Available Space List.

RP1	Compute I = HASH(KEY)
RP2	If POINTER(I) = 0, exit on failure.
RP3	If POINTER(I) $\neq$ 0, then I = POINTER(I)
RP4	If KEYWORD(I) = KEY, then J = INDEX(I), clear the old entry in ASL starting from ASL(J) and including an EOE
RP5	The new entry is stored in ASL starting from ASL(J)
RP6	If the new entry plus an EOE can be accomodated in the old space, exit on success.
RP7	If the new entry plus an EOE can not be accommodated in the old space, then store the new entry up to the same length of the old entry and put an EOE at the end, exit on partial success.
RP8	If KEYWORD(I) $\neq$ KEY and LINK(I) = 0, then exit on failure.
RP9	If KEYWORD(I) $\neq$ KEY and LINK(I) $\neq$ 0, then I = LINK(I), go to Step RP4.
RP10	Repeat for additional entries starting at Step RP1

Table 11 will display the changes made through the use of this algorithm upon Table 10.

Table 11. The available space list after a REPLACE command

```
THE AVAILABLE SPACE LIST --
ADDRESS                          INFORMATION

    1 DATA TRANSMISSION AND DATA PROCESSING DICTIONARY, BY JAMES F. HO
    9 LMES     EOE      (ABSOLUTE ADDRESS)THE IDENTIFICATION OF A SPECIF
   17 IC REGISTER OR LOCATIPN IN STORAGE.      EOE      (ABSOLUTE CODING
   25 )CODING IN WHICH ALL ADDRESSES REFER TO SPECIFIC MACHINE REGISTE
   33 RS AND MEMORY LOCATIONS.EOE      (ABSORPTION)THE LOSS OF ENERGY I
   41 N THE TRANSMISSION OF WAVES OVER RADIO  OR WIRE PATHS DUE TO CON
   49 VERSION INTO HEAT OR OTHER FORMS OF ENERGY.  IN WIRE TRANSMISSIO
   57 N, THE TERM IS USUALLY APPLIED ONLY TO LOSS OF ENERGY INEXTRANEO
   65 US MEDIA.      EOE      (1). AGGREGATE CROSSTALK FROM A LARGE NU
   73 MBER OF DISTURBING CHANNELS.      (2). UNWANTED DISTURBING SOUND
   81 S IN A CARRIER OR OTHER MULTIPLE CHANNEL SYSTEM WHICH RESULT FRO
   89 M THE AGGREGATE CROSSTALK OR MUTUAL INTERFERENCE FROM OTHER CHAN
   97 NELS.    EOE      (BACKGROUND NOISE)THE TOTAL SYSTEM NOISE INDEPEN
  105 DENT OF THE PRESENCE OR ABSENCE OF A SIGNAL.  THE SIGNAL IS NOT
  113 TO BE INCLUDED AS PART OF THE NOISE.      EOE      A SHIELDING STRU
  121 CTURE OR PARTITION USED TO INCREASE THE EFFECTIVE LENGTHOF THE E
  129 XTERNAL TRANSMISSION PATH BETWEEN TWO POINTS AS FOR TRANSMISSION
  137 PATH BETWEEN TWO POINTS AS FOR EXAMPLE, BETWEEN THE FRONT AND TH
  145 E BACK OF AN ELECTROACOUSTIC TRANSDUCER.EOE      EQUIPMENT--CASE
  153 DESIGNED TO HOUSE RELAYS AND OTHER APPARATUS.      KEY--CASE INST
  161 ALLED ON A CUSTOMER#S PREMISES, TO PERMIT DIFFERENT LINES TO THE
  169  CENTRAL OFFICE TO BE CONNECTED TO VARIOUS TELEPHONE STATIONS.
  177 IT HAS SIGNALS TO INDICATE ORIGINATING CALLS AND BUSY LINES.
  185  TEST--AUX CONTAINING APPARATUS FOR TROUBLE LOCATION AND ROUTINE
  193  MAINTENANCE.    EOE
  201
  209
  217                                          THREE UNIT LENGTHS OF SI
  225 STAINED SIGNAL, WHEN TRANSMITTED, A DASH WILL AUTOMATICALLY BE F
  233 OLLOWED BY ONE UNIT LENGTH OF SILENCE.  TERM USED IN RADIOTELEGR
  241 APHY.    EOE
  249
  257
  265
  273                                          (DATA CIRCUIT)COMMUNICAT
  281 ION FACILITY PERMITTING TRANSMISSION OF INFORMATION IN DIGITAL F
  289 ORM.      EOE      ELECTRICAL ACCOUNTING MACHINE.  EOE      AN ELECT
  297 ROACOUSTIC TRANSDUCER INTENDED TO BE CLOSELY COUPLED ACOUSTICALL
  305 Y TO THE EAR.      EOE      (EMERGENCY MEDICAL SERVICE)25 W MAIN,255
  313 -8567.  EOE      (EMERSON A T)1234 RUTLEDGE,123-4567.      EOE
  321 (EMERSON CHAS B)5314 MATHEWS RD MIDDLETON,238-5776.      EOE
  329 (EMERSON GAIL)222 LAKE LAWN PL,222-2222.EOE      (EMERSON HARLAND
  337 S)333 HILLTOP DR,333-3333.      EOE
  345                                          (EMERSON HARRY L MRS)2314 CHALET
  353  GARDENS RD,238-6067.    EOE      (EMERSON HUGH C)1004 TOMPKINS DR
  361 ,222-1434.      EOE      (EMERSON IDA MRS)419 JEAN,256-8126.
  369 EOE      (EMERSON RICHARD E)1610 CAMERON DR,238-1126.      EOE
  377 (EMERSON RODNEY)6266 ELMWOOD AV MIDDLETON,238-5769.      EOE
  385
  393
  401 (EMERY DONA F)2506 MCDIVITT RD,256-1284.EOE      (MADISON ADJUSTM
  409 ENT SYSTEM)303 PRICE PL, 238-2616.      EOE      (MADISON ACCEPTA
  417 NCE CORP INC)1201 W BELTLINE HY, 257-1091.      EOE      (MADIS A
  425 LMA MRS)3256 MILL, 244-7831.    EOE      (MADISON AIR SERVICE)330
  433 2 NORTH STOUGHTON RD, 249-6478. EOE
```

6. Algorithm PRINT (P)

This is a simple algorithm for a utility function of arranging information in table form and printing out of the Chaining Table and the Available Space List as those of Table 4 to Table 14 in this paper.

P1 Set I = 1 and print titles for the Chaining Table

P2 Print I, KEYWORD(I), INDEX(I), and POINTER(I)

P3 I = I + 1, repeat Step P2 until the end of the table is reached

P4 Set J = 1 and print the titles for the Available Space List

P5 Print J, and ASL(J)

P6 J = J + 1, repeat Step P5 until the end of the Available Space List is reached, exit on success.

7. Algorithm COMPRESS (C)

This is an algorithm designed to serve as a "Garbage Collector" in the list processing languages for better storage efficiency. In practical applications, the Available Space List is a huge free storage area which can be on a secondary bulk storage device such as a drum or disk for random access. After several updating functions performed on the HAICS file, there will inevitably be some space groups residing in the middle of the used portion of the Available Space List. And eventually it will reach a situation that the end of the Available Space List is reached but with many space groups scattered in the middle.

To remedy this situation, a periodical operation of the COMPRESS command is desirable to repack the Available Space List for a better storage utilization. Many strategies or hierarchies can be used to achieve this purpose with some variations in computing efficiency. The current algorithm starts with the last entry in the Available Space List and moves it to the first accomodatable space group found from

the beginning of the List. The process is repeated until all the accomodatable space groups found are filled and thus a large space group is accumulated at the end of the Available Space List for subsequent additions of new entries.

C1 Search for the largest INDEX(I) in the Chaining Table, then set J = INDEX(I)

C2 Count the length of this last entry in ASL starting at ASL(J) until an EOE is found

C3 Check an internal table of space groups in ASL to find an accomodatable space group such that the number of spaces in a space group is greater than the length of the last entry, go to Step C6 if it is not found, go to Step C4 if it is found

C4 Move the last entry to the space group found, go to Step C5 if some spaces are left unused, otherwise exit on success.

C5 Store information of the space group to the internal table, exit on success.

C6 Search ASL from its beginning for a space group, go to Step C10 if it is not found, go to Step C7 if it is found

C7 If the space group found is accomodatable, then go to Step C4, otherwise go to Step C8

C8 Store information of the space group found to the internal table

C9 Search ASL continuously for a space group, go to Step C10 if it is not found before the search reached the original location of the last entry, go to Step C7 if it is found

C10 Exit on table not compressable.

C11 Repeat for additional compression upon exit on success by entering at Step C1.

A sample result of the COMPRESS algorithm upon Tables 9 and 11 is shown in the following Tables 12 and 13:

Table 12. The chaining table after a COMPRESS command

THE CHAINING TABLE --

VALUE	KEYWORD	INDEX	LINK	POINTER
1	EARPHONE	296	0	1
2	ABSOCOD	23	17	2
3	EMERS R	257	0	0
4	EMERS HL	349	0	4
5	DASH	222	0	5
6		0	0	0
7		0	0	0
8	EAM	291	0	8
9	DATACIRC	278	0	9
10		0	0	0
11		0	0	0
12	BACKNOIS	99	13	12
13	CABINET	151	0	14
14	EMERS MS	335	0	0
15	EMERY DF	251	0	16
16	MACI	211	0	0
17	MADIS AM	206	0	19
18	EMS	308	0	18
19	MASER	196	0	0
20	EMERS I	272	21	20
21	EMERS RE	265	22	0
22	MAS	243	0	0
23	UTADPD	1	25	23
24	ABSOADOR	11	28	24
25	ABSORBTN	37	0	32
26	BAFFLE	119	0	26
27		0	0	31
28	EMERS AT	315	0	0
29	EMERS CH	321	0	3
30	BABBLE	68	0	30
31	EMERS G	329	0	15
32	EMERS HC	342	0	0

Table 13. The available space list after a COMPRESS command

```
THE AVAILABLE SPACE LIST --
ADDRESS                    INFORMATION

   1 DATA TRANSMISSION AND DATA PROCESSING DICTIONARY, BY JAMES F. HO
   9 LMES     EOE      (ABSOLUTE ADDRESS)THE IDENTIFICATION OF A SPECIF
  17 IC REGISTER OR LOCATIPN IN STORAGE.     EOE      (ABSOLUTE CODING
  25 )CODING IN WHICH ALL ADDRESSES REFER TO SPECIFIC MACHINE REGISTE
  33 RS AND MEMORY LOCATIONS.EOE      (ABSORPTION)THE LOSS OF ENERGY I
  41 N THE TRANSMISSION OF WAVES OVER RADIO  OR WIRE PATHS DUE TO CON
  49 VERSION INTO HEAT OR OTHER FORMS OF ENERGY.  IN WIRE TRANSMISSIO
  57 N, THE TERM IS USUALLY APPLIED ONLY TO LOSS OF ENERGY INEXTRANEO
  65 US MEDIA.       EOE      (1). AGGREGATE CROSSTALK FROM A LARGE NU
  73 MBER OF DISTURBING CHANNELS.      (2).  UNWANTED DISTURBING SOUND
  81 S IN A CARRIER OR OTHER MULTIPLE CHANNEL SYSTEM WHICH RESULT FRO
  89 M THE AGGREGATE CROSSTALK OR MUTUAL INTERFERENCE FROM OTHER CHAN
  97 NELS.   EOE      (BACKGROUND NOISE)THE TOTAL SYSTEM NOISE INDEPEN
 105 DENT OF THE PRESENCE OR ABSENCE OF A SIGNAL.  THE SIGNAL IS NOT
 113 TO BE INCLUDED AS PART OF THE NOISE.     EOE      A SHIELDING STRU
 121 CTURE OR PARTITION USED TO INCREASE THE EFFECTIVE LENGTHOF THE E
 129 XTERNAL TRANSMISSION PATH BETWEEN TWO POINTS AS FOR TRANSMISSION
 137 PATH BETWEEN TWO POINTS AS FOR EXAMPLE, BETWEEN THE FRONT AND TH
 145 E BACK OF AN ELECTROACOUSTIC TRANSDUCER.EOE      EQUIPMENT--CASE
 153 DESIGNED TO HOUSE RELAYS AND OTHER APPARATUS.     KEY--CASE INST
 161 ALLED ON A CUSTOMER#S PREMISES, TO PERMIT DIFFERENT LINES TO THE
 169  CENTRAL OFFICE TO BE CONNECTED TO VARIOUS TELEPHONE STATIONS.
 177 IT HAS SIGNALS TO INDICATE ORIGINATING CALLS AND BUSY LINES.
 185  TEST--BOX CONTAINING APPARATUS FOR TROUBLE LOCATION AND ROUTINE
 193  MAINTENANCE.   EOE      (MADISON AIR SERVICE)3302 NORTH STOUGHTO
 201 N RD, 249-6478.  EOE      (MADIS ALMA MRS)3256 MILL, 244-7831.
 209 EOE      (MADISON ACCEPTANCE CORP INC)1201 W BELTLINE HY, 257-1109
 217 1.     EOE      THREE UNIT LENGTHS OF SI
 225 STAINED SIGNAL. WHEN TRANSMITTED, A DASH WILL AUTOMATICALLY BE F
 233 OLLOWED BY ONE UNIT LENGTH OF SILENCE.  TERM USED IN RADIOTELEGR
 241 APHY.   EOE      (MADISON ADJUSTMENT SYSTEM)303 PRICE PL, 238-261
 249 6.      EOE      (EMERY DONA F)2506 MCDIVITT RD,256-1284.EOE
 257 (EMERSON RODNEY)6266 ELMWOOD AV MIDDLETON,238-5749.     EOE
 265 (EMERSON RICHARD E)1610 CAMERON DR,238-1126.     EOE      (EMERSON
 273  IDA MRS)419 JEAN,256-8128.     EOE      (DATA CIRCUIT)COMMUNICAT
 281 ION FACILITY PERMITTING TRANSMISSION OF INFORMATION IN DIGITAL F
 289 ORM.   EOE      ELECTRICAL ACCOUNTING MACHINE.  EOE      AN ELECT
 297 ROACOUSTIC TRANSDUCER INTENDED TO BE CLOSELY COUPLED ACOUSTICALL
 305 Y TO THE EAR.   EOE      (EMERGENCY MEDICAL SERVICE)25 W MAIN,255
 313 -4567.  EOE      (EMERSON A T)1234 RUTLEDGE,123-4567.     EOE
 321 (EMERSON CHAS H)5314 MATHEWS RD MIDDLETON,238-5776.     EOE
 329 (EMERSON GAIL)222 LAKE LAWN PL,222-2222.EOE      (EMERSON HARLAND
 337  S)333 HILLTOP DR,333-3333.     EOE      (EMERSON HUGH C)1004 TOM
 345 PKINS DR,222-1438.     EOE      (EMERSON HARRY L MRS)2314 CHALET
 353  GARDENS RD,238-6067.   EOE
 361
 369
 377
 385
 393
 401
 409
 417
 425
 433
```

8. Algorithm LIST (L)

In contrast to the Algorithm PRINT, LIST will initiate an alphabetical sort on the keywords stored in the Chaining Table, and output an alphabetical list of all entries in the HAICS file. The final output of this algorithm performed on Tables 12 and 13 is illustrated in Table 14.

L1	Sort array KEYWORD(I) alphabetically and carry along the original sequence in the array, I, during the sort process
L2	Take the first original sequence number in the sorted keyword order, I
L3	Set J = INDEX(I), print the hash value, the keyword, the entry starting address in ASL, and the entry itself, exit on success.
L4	Repeat for the next keyword and its original sequence number in the sorted array until this array is exhausted.

LIST

Table 14. The alphabetical list after a LIST command

```
THE ALPHABETICAL LIST --

          HASH VALUE =   24          ENTRY ADDRESS IN ASL =   11
ABSOADDR  (ABSOLUTE ADDRESS)THE IDENTIFICATION OF A SPECIFIC REGIS
          TER OR LOCATIPN IN STORAGE.     EOE

          HASH VALUE =    2          ENTRY ADDRESS IN ASL =   23
ABSOCOD   (ABSOLUTE CODING)CODING IN WHICH ALL ADDRESSES REFER TO
          SPECIFIC MACHINE REGISTERS AND MEMORY LOCATIONS.EOE

          HASH VALUE =   25          ENTRY ADDRESS IN ASL =   37
ABSORPTN  (ABSORPTION)THE LOSS OF ENERGY IN THE TRANSMISSION OF WA
          VES OVER RADIO  OR WIRE PATHS DUE TO CONVERSION INTO HE
          T OR OTHER FORMS OF ENERGY.  IN WIRE TRANSMISSION, THE T
          ERM IS USUALLY APPLIED ONLY TO LOSS OF ENERGY INEXTRANEO
          US MEDIA.       EOE

          HASH VALUE =   30          ENTRY ADDRESS IN ASL =   98
BABBLE    (1). AGGREGATE CROSSTALK FROM A LARGE NUMBER OF DISTURBI
          NG CHANNELS.     (2). UNWANTED DISTURBING SOUNDS IN A C
          ARRIER OR OTHER MULTIPLE CHANNEL SYSTEM WHICH RESULT FRO
          M THE AGGREGATE CROSSTALK OR MUTUAL INTERFERENCE FROM OT
          HER CHANNELS.   EOE

          HASH VALUE =   12          ENTRY ADDRESS IN ASL =   99
BACKNOIS  (BACKGROUND NOISE)THE TOTAL SYSTEM NOISE INDEPENDENT OF
          THE PRESENCE OR ABSENCE OF A SIGNAL.  THE SIGNAL IS NOT
          TO BE INCLUDED AS PART OF THE NOISE.     EOE

          HASH VALUE =   26          ENTRY ADDRESS IN ASL =  119
BAFFLE    A SHIELDING STRUCTURE OR PARTITION USED TO INCREASE THE
          EFFECTIVE LENGTHOF THE EXTERNAL TRANSMISSION PATH BETWEE
          N TWO POINTS AS FOR TRANSMISSIONPATH BETWEEN TWO POINTS
          AS FOR EXAMPLE, BETWEEN THE FRONT AND THE BACK OF AN ELE
          CTROACOUSTIC TRANSDUCER.EOE

          HASH VALUE =   13          ENTRY ADDRESS IN ASL =  151
CABINET   EQUIPMENT--CASE DESIGNED TO HOUSE RELAYS AND OTHER APPAR
          ATUS.     KEY--CASE INSTALLED ON A CUSTOMER*S PREMISES,
          TO PERMIT DIFFERENT LINES TO THE CENTRAL OFFICE TO BE CO
          NNECTED TO VARIOUS TELEPHONE STATIONS.  IT HAS SIGNALS T
          O INDICATE ORIGINATING CALLS AND BUSY LINES.     TEST--A
          CK CONTAINING APPARATUS FOR TROUBLE LOCATION AND ROUTINE
          MAINTENANCE.    EOE

          HASH VALUE =    5          ENTRY ADDRESS IN ASL =  222
DASH      THREE UNIT LENGTHS OF SUSTAINED SIGNAL, WHEN TRANSMITTED
          , A DASH WILL AUTOMATICALLY BE FOLLOWED BY ONE UNIT LENG
          TH OF SILENCE.  TERM USED IN RADIOTELEGRAPHY.    EOE

          HASH VALUE =    9          ENTRY ADDRESS IN ASL =  278
DATACIRC  (DATA CIRCUIT)COMMUNICATION FACILITY PERMITTING TRANSMIS
          SION OF INFORMATION IN DIGITAL FORM.     EOE

          HASH VALUE =   23          ENTRY ADDRESS IN ASL =    1
DTADPD    DATA TRANSMISSION AND DATA PROCESSING DICTIONARY, BY JAM
          ES E. HOLMES     EOE

          HASH VALUE =    8          ENTRY ADDRESS IN ASL =  291
```

```
EAM         ELECTRICAL ACCOUNTING MACHINE.  EOE

          HASH VALUE =    1          ENTRY ADDRESS IN ASL =  296
EARPHONE  AN ELECTROACOUSTIC TRANSDUCER INTENDED TO BE CLOSELY COU
          PLED ACOUSTICALLY TO THE EAR.   EOE

          HASH VALUE =   28          ENTRY ADDRESS IN ASL =  315
EMERS AT  (EMERSON A T)1234 RUTLEDGE,123-4567.   EOE

          HASH VALUE =   29          ENTRY ADDRESS IN ASL =  321
EMERS CR  (EMERSON CHAS B)5314 MATHEWS RD MIDDLETON,238-5776.
          EOE

          HASH VALUE =   31          ENTRY ADDRESS IN ASL =  329
EMERS G   (EMERSON GAIL)222 LAKE LAWN PL,222-2222,EOE

          HASH VALUE =   32          ENTRY ADDRESS IN ASL =  342
EMERS HC  (EMERSON HUGH C)1004 TOMPKINS DR,222-1438.      EOE

          HASH VALUE =    4          ENTRY ADDRESS IN ASL =  349
EMERS HL  (EMERSON HARRY L MRS)2314 CHALET GARDENS RD,238-6067.
          EOE

          HASH VALUE =   14          ENTRY ADDRESS IN ASL =  335
EMERS HS  (EMERSON HARLAND S)333 HILLTOP DR,333-3333.     EOE

          HASH VALUE =   20          ENTRY ADDRESS IN ASL =  272
EMERS I   (EMERSON IDA MRS)419 JEAN,256-8128.     EOE

          HASH VALUE =    3          ENTRY ADDRESS IN ASL =  257
EMERS R   (EMERSON RODNEY)6266 ELMWOOD AV MIDDLETON,238-5769.
          EOE

          HASH VALUE =   21          ENTRY ADDRESS IN ASL =  265
EMERS RE  (EMERSON RICHARD E)1610 CAMERON DR,238-1126.    EOE

          HASH VALUE =   15          ENTRY ADDRESS IN ASL =  251
EMERY DF  (EMERY DONA F)2506 MCDIVITT RD,256-1284,EOE

          HASH VALUE =   18          ENTRY ADDRESS IN ASL =  388
EMS       (EMERGENCY MEDICAL SERVICE)25 W MAIN,255-8567.  EOE

          HASH VALUE =   16          ENTRY ADDRESS IN ASL =  210
MACI      (MADISON ACCEPTANCE CORP INC)1201 W BELTLINE HY, 257-109
          1.      EOE

          HASH VALUE =   17          ENTRY ADDRESS IN ASL =  244
MADIS AM  (MADIS ALMA MRS)3256 MILW, 244-7831.    EOE

          HASH VALUE =   22          ENTRY ADDRESS IN ASL =  243
MAS       (MADISON ADJUSTMENT SYSTEM)303 PRICE PL, 238-2616.
          EOE

          HASH VALUE =   19          ENTRY ADDRESS IN ASL =  196
MASER     (MADISON AIR SERVICE)3302 NORTH STOUGHTON RD, 249-6478.
          EOE

END OF ISRU RUN
```

VI. DISCUSSION

1. Sample Statistics of the Main and Update HAICS Algorithms

For the purpose of demonstrating the actual performance of the HAICS main and update algorithms, the statistics gathered from the test run (which also produced Tables 5-11) are listed below in Tables 15 and 16 and are the basis for a preliminary discussion.

Table 15. Sample statistics of the main HAICS algorithms

Sample entry sequence	STORE		RETRIEVE	
	Number of searches of current entry	Accumulative average number of searches	Number of searches of current entry	Accumulative average number of searches
1	0	0	1	1.0
2	0	0	1	1.0
3	0	0	1	1.0
4	1	0.25	1	1.0
5	0	0.2	1	1.0
6	0	0.167	1	1.0
7	0	0.143	1	1.0
8	1	0.25	2	1.125
9	0	0.222	1	1.111
10	0	0.2	1	1.1
11	1	0.273	1	1.091
12	0	0.25	1	1.083
13	0	0.231	1	1.077
14	0	0.214	2	1.143
15	0	0.2	1	1.133
16	1	0.25	1	1.125
17	2	0.359	1	1.118
18	1	0.389	3	1.222
19	0	0.368	2	1.263
20	0	0.35		
21	0	0.333		
22	0	0.318		
23	0	0.304		
24	1	0.333		
25	0	0.32		
26	0	0.308		
27	0	0.296		
28	0	0.286		

Table 16. Sample statistics of the update HAICS algorithms

Sample entry sequence	ADD		DELETE		REPLACE	
	Number of searches of current entry	Accumulative average number of searches	Number of searches of current entry	Accumulative average number of searches	Number of searches of current entry	Accumulative average number of searches
1	2	2.0	1	1.0	1	1.0
2	0	1.0	1	1.0	2	1.5
3	1	1.0	1	1.0	1	1.333
4	0	0.75	1	1.0		
5			2	1.2		

The STORE efficiency, i.e., the accumulative average number of searches for the STORE algorithm, as shown in Table 15 reveals that starting with an empty chaining table, it is a low 0.286 at 87.5% fullness of the table. Most entries are entered into this table with no search at all which implies a good balanced distribution of keyword hash values.

The ADD efficiency is a function of the STORE efficiency. And in this sample's statistics the ADD efficiency obtained through the addition of four entries to make a full chaining table, is in fact the same as if these four entries are placed at the end of the STORE command. Thus the ADD efficiency of 0.75 for four entries can be combined with the STORE efficiency for twenty-eight entries and the result is a 0.344 of STORE efficiency for a full 32-entry chaining table. It is noted that the ADD efficiency is always greater than (or equal to) the STORE efficiency due to the non-emptiness of the chaining table.

The RETRIEVE efficiency is always identical with the search efficiency as indicated in Table 3 which is an average of 1.25 for the indirect chaining method. The accumulative average number of searches does fall into the range between the minimum of 1.0 and the maximum of 1.5 which is a 1.263 at 59.4% table fullness.

Both the DELETE and REPLACE efficiencies are functions of the RETRIEVE efficiency or the search efficiency. The sample statistics of accumulative average number of searches of 1.2 for deleting five entries and of 1.333 for replacing three entries gives some indication that the DELETE and REPLACE efficiencies are compatible with the RETRIEVE efficiency.

As mentioned before, the above discussion is preliminary and even premature. The statistics in Tables 15 and 16 do not cover some unusual circumstances although it is a typical example of several regular test runs. To support, or oppose, the above discussion will demand several further extensive tests of each of these five efficiencies under a controlled and isolated environment.

2. A Framework for Information Systems

The HAICS method is a basic framework aimed to improve the total efficiency of an information system. It can be programmed in a number of languages from the fundamental machine language or assembly language of a particular family of computers to the high-level procedure-oriented languages such as Fortran and Algol which are acceptable to most of the computers.

With an amazing 1.25 average number of searches per entry, this method will certainly make natural language processing not much worse than the numerical computation. It is ready to be implemented for text processing and document retrieval; numerical data retrieval; and for handling of large files such as dictionaries, catalogs, and personnel records, as well as graphic informations. In the test program coded in Fortran and a machine language COMPASS, eight commands as described before are currently implemented and operational in batch mode on a CDC 3600. Further development will be on the use of teletype console, CRT terminal, and plotter under a time-sharing

environment for producing immediate responses. This is under the ideal of placing the most complete encyclopedia or a tailored index-reference work under one's fingertip.

Specifically, the dictionary lookup operation as the principal operation of an information system, is no longer a lengthy and painful procedure and thus a barrier in natural language processing. Linguistic analysis may be provided with a complete freedom in referring back and forth any entry in the dictionary and the grammar, and the information gained at any stage of analysis can be stored and retrieved in the same way. Document retrieval may go deeper in content analysis and providing a synonym dictionary for some better query descriptor transformations and matching functions. As Shoffner noted, "it is important to be able to determine the extent to which file structure and search techniques influence recall, precision, and other measures of system performance." This paper tends to support Shoffner's statement by presenting an analysis of current search techniques and a detailed description of the HAICS method which is a possible framework for most information systems.

REFERENCES

Becker, Joseph, and Hayes, Robert M. Information Storage and Retrieval: tools, elements, theories. Wiley, New York, 1963.

Bobrow, D. G. "Syntactic Theory in Computer Implementations," Automated Language Processing, edited by Harold Borko. Wiley, New York, 1967, pp.217-251.

Borko, Harold. "Indexing and Classification," Automated Language Processing, edited by Harold Borko. Wiley, New York, 1967, pp.99-125.

Bourne, Charles P. Methods of Information Handling. Wiley, New York, 1963.

Hayes, David G. Introduction to Computational Linguistics. American Elsevier, New York, 1967.

Johnson, L. R. "Indirect Chaining Method for Addressing on Secondary Keys," Communications of the ACM, 4(May,1961), pp.218-222.

King, Donald W. "Design and Evaluation of Information Systems," Annual Review of Information Science and Technology, Volume 3, edited by Carlos A. Cuadra. Encyclopedia Britannica, Chicago, 1968, pp.61-103.

Knuth, Donald E. The Art of Computer Programming, Volume 1/ Fundamental Algorithms. Addison-Wesley, Reading, Massachusetts, 1968.

Lamb, Sydney M. and Jacobsen, William H., Jr. "A High-Speed Large-Capacity Dictionary System," Readings in Automatic Language Processing, edited by David G. Hays. American Elsevier, New York, 1966, pp.51-72. Also in Mechanical Translation, 6 (November, 1961), pp. 76-107.

Lee, T. C.; Wang, H. T.; and Yang, S. C. "An Experimental Model for Chinese to English Machine Translation." Paper presented at the Annual Meeting of the Association for Machine Translation and Computational Linguistics, San Françisco, 1966.

Lee, T.C.; Wang, H. T.; Yang, S. C.; and Farmer, E. Linguistic Studies for Chinese to English Machine Translation. Itek Corporation, Lexington, Massachusetts, 1965. Also available from ERIC Document Reproduction Service as ED 010 872.

Maurer, W. D. Programming: an introduction to computer languages and techniques. Holden-Day, San Francisco, 1968.

Meadow, Charles T. The Analysis of Information Systems: A Programmer's Introduction to Information Retrieval. Wiley, New York, 1967.

Morris, Robert. "Scatter Storage Techniques," Communications of the ACM, 11(January, 1968), pp.38-44.

Pendergraft, E.D. "Translating Languages," Automated Language Processing, edited by Harold Borko. Wiley, New York, 1967, pp.291-323.

Peterson, W. W. "Addressing for Random-Access Storage," IBM J. Res. Dev. 1(April, 1957), pp.130-146.

Salton, Gerard. Automatic Information Organization and Retrieval. McGraw-Hill, New York, 1968.

Sedelow, Salley Yeates, and Sedelow, Walter A., Jr. "Stylistic Analysis," Automatic Language Processing, edited by Harold Borko. Wiley, New York, 1967, pp.181-213.

See, Richard. "Machine-Aided Translation and Information Retrieval," Electronic Handling of Information: Testing & Evaluation, edited by Allen Kent; Orrin E. Taulbee; Jack Belzer; and Gordon D. Goldstein. Thompson, Washington, D.C., and Academic Press, London, 1967, pp.89-108.

Shoffner, Ralph M. "Organization, Maintenance and Search of Machine Files," Annual Review of Information Science and Technology, Volume 3, edited by Carlos A. Cuadra. Encyclopedia Britannica, Chicago, 1968, pp.137-167.

Simmons, R.F. "Answering English Questions by Computer," Automated Language Processing, edited by Harold Borko. Wiley, New York, 1967, pp.253-289.

Travis, Larry E. "Analytic Information Retrieval," Natural Language and the Computer, edited by Paul L. Garvin. McGraw-Hill, New York, 1963, pp.310-353.

Venezky, Richard L. "Storage, Retrieval, and Editing of Information for a Dictionary," American Documentation, 19 (January, 1968), pp.71-79.

Wegner, Peter. Programming Languages, Information Structures, and Machine Organization. McGraw-Hill, New York, 1968.

Wyllys, Ronald E. "Extracting and Abstracting by Computer," Automated Language Processing, edited by Harold Borko. Wiley, New York, 1967, pp.127-179.

Yang, S. C. "Automatic Segmentation and Phrase-Structure Parsing: a Simple Chinese Parser," Thought and Word, 6 (January, 1969), pp.324-331.