JOINT COMMITTEE REPORT ON

LARGE SCALE MEMORY ADDRESSING

June 20, 1958

San Jose Product Planning Technical Report #3

ABSTRACT

In applying random access storage devices to in-line data processing, the problem of addressing, that is, the association of item identifications with machine addresses, becomes more acute as storage capacity increases. Although workable solutions exist for present storage systems, they will not be adequate in the future. The increased capacities of devices in development and the broadened range of future applications demand an addressing technique more effective than what is now available.

To investigate the addressing problem, a Joint Committee was formed. During the month of its existence, The Committee reviewed the known addressing techniques and attempted to define functionally an optimum addressing system. For the evaluation of a present or future technique, the Committee adopted the four basic criteria of minimum retrieval time, maximum storage utilization, ease of file maintenance, and minimum external manipulation.

Of the three classes of known addressing techniques (direct addressing, mathematical manipulation, and table look-up), only the table look-up method offers general purpose applicability and reasonably measures up to the basic evaluation criteria. However, the method is not widely used today due to limitations in the existing random access storage systems. The Committee favors the development of an auxiliary storage device to implement this technique and proposes three such approaches for table look-up as an interim solution to the addressing problem.

A description of functional characteristics of an optimal technique termed "associative addressing" was attempted. Although the association process cannot be specified at the present time, the desired end result is to retrieve instantaneously any item in storage by one of its identifiers without any manual or programmed effort. Such a technique should permit full utilization of storage space and eliminate file maintenance and external manipulation.

In conclusion, the Committee recommends hardware development to automatize the table and storage search, mathematical research to derive suitable address conversion processes, and a systems survey to define precisely the requirements of potential file applications.

TABLE OF CONTENTS

1. INTRODUCTION

In-line accounting methods have only recently been automated by the introduction of large random access memories in the multi-million character range. Because of this newness, our present in-line accounting systems know-how and experience are quite limited in comparison with well-established batch accounting methods.

A major problem encountered in using these memories is addressing, i.e., the placement and retrieval of items stored. Three basic techniques have been used with present systems: direct addressing, mathematical manipulation, and table look-up. Although workable, each of these techniques has various disadvantages.

Technological advances now being developed will greatly broaden the application range of data processing equipment. One of these applications, Integrated Data Processing, brings up many unexplored file areas related to addressing, e.g., multiple item files, multiple identifiers, and output sorting. This extension of systems and applications has created a sense of urgency for further investigation of addressing techniques.

To initiate the investigation of addressing, a Joint Committee was formed May 19, 1958, with the following participants:

W. G. Dye	Product Planning	San Jose
W. R. English	Product Planning	Poughkeepsie
H. Jeans	Applied Programming	San Jose
A. D. Lin	Product Development	San Jose
E. E. Lindstrom	Applied Science	San Francisco Office
F. B. Wood	Research Laboratory	San Jose

The investigation continued through June 20, 1958.

The purpose of this Committee effort was to:

1. Define the problem of addressing.
2. Evaluate existing addressing techniques.
3. Suggest new techniques.
4. Propose areas for further intensive study.

The scope of effort was:

1. Confined to addressing techniques for large capacity random access mechanical memories (5 million to one billion character capacity).
2. Concerned primarily with a functional approach, not with determining the operating characteristics (speed, capacity, etc.) of the storage devices.
3. Directed toward general purpose solutions.
4. To evaluate possible solutions with respect to storage devices in development.
5. To functionally describe any hardware for addressing as required for a desirable technique.

The basic problem in addressing can be stated simply as the association of the item identification and the machine address. An item cannot be placed in or recovered from the storage until this association is completed. The three aspects of this problem are:

1. Transformation - conversion of the item identifier to an intermediate code.
2. Mapping - the optimal distribution of these codes within the storage device.
3. Retrieval - the recovery of an item in whole or part as desired.

Other factors to be discussed in the following sections will reflect the effectiveness of addressing techniques.

2. EVALUATION FACTORS

In evaluating addressing techniques, certain criteria must be established. Basically, the effectiveness of an addressing technique can be measured in terms of (1) time required for item retrieval, (2) percentage of storage capacity utilized for item files, (3) effort required in maintaining the item files to assure retrieval efficiency, and (4) external manipulation to set up and apply the technique.

Time Required for Item Retrieval - The item retrieval time consists of the time that the identification of an item to be retrieved from the storage is made available, the time that the identification is associated with a machine address, and the time that the item is located in the storage and completely read out for processing.

It is preferable that the address association process be independent of the data processor and the storage device, and its time minimized. The storage, independent of the processor, should take no more than one "seek and read" to retrieve the required item. Thus, the processor is involved only in the transfer of the identification and the item itself.

Storage Utilization - It is desirable that the machine file be used to full capacity for item storage. In practice, a given addressing technique may reduce the available storage capacity by requiring the storage of tables and programs essential to item retrieval, or by inherently excluding certain addresses for possible association with the identifications. Also, some addressing techniques become less efficient with storage utilization above a certain percentage.

File Maintenance - File maintenance is required because of changes in the item file. When such changes in the file are accumulated over a period of time, a given addressing technique may become less effective. Thus, it may be necessary to rearrange active items in the storage, purge inactive items, and revise the technique. Ideally, an addressing technique should alleviate, if not completely eliminate, the effort required for file maintenance.

External Manipulation - External manipulation includes the initial effort in setting up the best possible addressing technique for a particular application and the manual operations necessary to implement the technique in routine processes. It is desirable that the initial set-up of a technique be simple and straight-forward for customers' personnel to apply, and identifications not be manually transformed in a normal retrieval process.

Secondary Factors - In addition to the above four basic criteria, some other factors must also be considered such as item retrieval by an identification other than the one used in normal processing, reference to one or more items as required in the processing of a given item, and output sorting in a sequence different from the item sequence in the storage. These secondary factors are described in detail in Appendix 2. They are peculiar to large business files, and any addressing techniques must be able to cope with them adequately.

3. PRESENT ADDRESSING TECHNIQUES

Present methods of addressing large capacity random access storages fall into one of three primary categories: direct addressing, mathematical manipulation and table look-up. Generally, a combination of the three methods is devised to suit a particular application.

Direct Addressing - In direct addressing, the item identification (or part of it) is the same as the machine address where the item is stored. With its machine address directly available, the item can be retrieved with one access to the storage, and no time is required for address conversion in the processor or storage. Storage utilization and file maintenance depend upon the degree of control over available addresses. A tight control (manual or automatic) assures high storage utilization and minimum file maintenance.

Since existing item identifications may not be directly used as machine addresses, it may be necessary to reassign addresses as identifications or append one to the other. Such a conversion may not be feasible in practice. The direct addressing method does not allow the retrieval of any item by an identification other than its machine address. In case the item size is increased, in excess of the allotted machine record length, the excess part may not be addressed directly.

Addressing by Mathematical Manipulation - The item identification is transformed into a machine address by a suitable mathematical process usually carried out by the processor. As duplicate addresses may result from manipulating the identifications, multiple items may be stored in locations linked to the one address. The search for a given item from one location to the next may be time-consuming in the storage and impede the efficiency of item retrieval. This indirectly limits the storage utilization to a certain percentage of its capacity.

The main drawback of this technique is the complex initial effort required in analyzing the set of identifications and selecting the appropriate method of manipulation. When the number of items in the file is changed to the extent that the distribution of their identifications is altered, it may be necessary to modify the manipulation method in order to maintain the retrieval efficiency.

An item can only be directly retrieved by its primary identification. Since items in the file are not stored in any predetermined sequence, batch processing in identification sequence does not necessarily minimize the access time. However, if items with the same address are loaded into the storage according to their activities (with the most active one loaded first), the average retrieval time can be significantly reduced.

Addressing by Table Look-up - The machine address of an item is determined by searching a table of item identifications and their corresponding addresses. The time required for table searching depends on the table organization and equipment characteristics. Once its address is found, the item can be readily retrieved from the storage.

The table look-up method has these points of advantage: (1) it is of general applicability to any item identification set; (2) setting up the table is direct and free from extensive pre-analysis; (3) the table usually requires only a small fraction of storage space as compared with the item file, hence, correspondingly less time for search. However, when the table storage approaches item storage, the method obviously loses effectiveness.

With multiple entry table or separate tables, it is possible to retrieve items by any one of several identifications. When the item length is not the same as the machine record length, the table function can specify the former (in addition to the machine address) to facilitate retrieval.

As this method requires the storage of tables, storage utilization is inherently penalized. Furthermore, tables must generally be maintained in good order to allow additions, deletions, and error-free references.

Combination Techniques - In practice, a combination of the three primary addressing techniques is often worked out to take advantage of the distribution of existing identifications and file activities, and to meet the requirements of a particular application. Such combinations tend to be tailor-made and are frequently lacking in general applicability. It is beyond the scope of this report to enumerate all of the possible combination methods.

Summary of Present Techniques - A detailed appraisal of the three primary addressing techniques is presented in Appendix 3. The discussion here and in the appendix supports the contention that, of these techniques, only table look-up offers general applicability and reasonable satisfaction of the basic evaluations factors cited in Section 2.

To date, however, addressing by table look-up has been infrequently used. Its full potential has not been exploited because of certain factors. If main memory is used to house the table(s), planning for 100% data usage of storage is impossible. Also, main memory and the processor may be tied up while table search is being made. The search routine usually associated with this technique is the sequential scanning of the table from beginning to end, resulting in an average search time equal to half the time required for a complete table scan. Changes in the table generally involve rearranging the complete table.

The answer to these objections may be met by providing a separate storage device to house the table with independent logic for high-speed searching and file maintenance independent of a processor. Table search time may be substantially reduced by good organization so that convergence is made to a coarse area before sequential scan is begun. Other system stratagems would shorten the time further.

It has been mentioned elsewhere in this report that until a breakthrough in the addressing art occurs, every effort should be made to maximize the effectiveness of our known techniques. The point of emphasis here is this: with special-purpose hardware and system sophistications, table look-up addressing may be brought to light in its true potential of general applicability and effectiveness.

The three interim proposals for table look-up addressing in a later section illustrate how this may be done.

4. OPTIMAL ADDRESSING TECHNIQUE

A group of functions to fulfill the addressing requirements of the future are presented here. These functions, although strictly "blue sky," satisfy the criteria established by the Committee and should help stimulate further thinking that may provide a solution to addressing files of the future.

The approach has been termed "associative addressing." The association process functionally supplies information leading to the retrieval of an item from a machine file when one of the item's identifiers is provided. Each individual may think of this association as a different process and what this process is remains to be determined. One might think of this as the process that takes place in the human brain or as a conditioned reflex (given a stimulus such as an identification, a response instantaneously links and provides the desired information).

The association process accomplishes the transformation and mapping aspects of addressing leading to the automatic retrieval of the desired information from a machine file. An instantaneous association will reduce the retrieval time to that inherently required by the machine file. Ultimately, this may approach the time required to transfer the information from the machine file - the machine's data flow rate.

Association links are made as items are added and eliminated when items are deleted. Upon requesting an item the association links indicate the space occupied by the item. Of course, the association process must be provided with the identifier type (primary, secondary #1, #2, etc.) in addition to the identifier to retrieve the appropriate item. For multiple items with the same secondary identifier, such as a particular part type, or due date, the association would sequentially retrieve all the items with that particular identifier.

The association process must be able to determine available space (space not in use for storing items). Thus, space made available when an item is deleted can be used immediately for another item. Additionally, the

associative framework should be capable of expansion so that reference to modular or interchangeable types of files would not require reshuffling of items within the system.

To add an item to the machine file association links would be created between an available location and the item's identifiers. Of the locations available, one would be automatically selected according to the item's activity, size, and possibly the item's content and classification. Therefore, placement of an item in the machine file would be determined by the above item characteristics and not in a fixed or rigid manner according to one of the item's identifiers.

"Associative addressing" provides a dynamic system capable of handling the ever-changing data encountered in business. As a direct result of this dynamic "associative addressing" storage utilization would be dependent entirely upon the number of items stored and would not be limited by retrieval techniques. Item file maintenance problems as encountered today would be completely eliminated for the organization and distribution of items in a machine memory would no longer be an addressing consideration. Additionally, no external manipulation for initial set-up or the operation of the retrieval system would be required.

"Associative addressing" will provide the functions required to mechanize many applications not currently practical on data processing equipment. For the present, however, no method for implementing the "associative addressing" function has been evolved. It is to be hoped that future mathematical, systems, and engineering effort will provide solutions permitting practical realization of "associative addressing."

5. PROPOSED INTERIM TECHNIQUES

Table Look-up Methods - Three table look-up methods are proposed as an interim solution to the problem of addressing large capacity mechanical storages. (The methods proposed do not fall within the restrictions imposed upon IBM on the use of certain table look-up [indexing] procedures. The Patent Department may be contacted for a list of these restrictions.) The three methods are: Multiple Level Table Look-up, Modified Binary Search of a Sorted Table, and Indexing Scheme using an Unsorted Table. A detailed description of these methods appears in Appendix 4.

The Multiple Level Table Look-Up requires the search of a small first level table to reach a part of the second level table from which the storage address of a group of items is found. The table arguments represent ranges of item identifications and are in sequential order. However, the items stored at a given address are not sorted and must be scanned to yield the required item.

The Modified Binary Search of a Sorted Table takes advantage of the binary search to reach a specific portion of the table. A linear search of that portion determines the range the given identification falls in. During the search process, a binary machine address is automatically generated. A search of a short table of identifications in that location addresses the required item.

In the Indexing Scheme Using an Unsorted Table, arguments exist as ranges with their high and low limits. A "row" contains the unsorted arguments within a range. At the common end of each row is placed the highest value for that row. These values, consecutively organized, constitute the "column" down which a coarse search proceeds until a "less than" comparison initiates a fine search across the row. An inclusion response derives the bulk memory address for retrieval of the item.

All three proposed methods use a separate table storage device (a drum or disk) operating independently of the main storage and processor. The main file storage is required to have the ability for rapid scanning of

stored items. Thus, the table searching, storage scanning and processing time can be overlapped, resulting in efficient item retrieval.

As several items in the same range may be stored in one location (or locations linked to the given one), the file is loosely organized, reducing the size of the table and the need for frequent maintenance. The table of identification ranges can be so arranged that minimum access in switching from one section to the next will be realized.

The table look-up methods are inherently general purpose in application, irrespective of identification structure and distribution. They reasonably meet the basic criteria set forth previously and satisfy the many requirements of a business file system.

Mathematical Techniques - Address conversion by mathematical manipulation may be described as a transformation procedure which when applied to a primary identifier results in an index number equivalent to an address or location on a machine file. There are several possible approaches to this basic problem, the most common being one of the various pseudo-randomizing techniques. It would appear that some additional research should be undertaken in this area because the theoretical environment for randomizing is not always present, i.e., the resultant set of numbers may not be wholly described or the machine file may prevent fully developed choices.

One area of study that suggests a more vigorous investigation is the theory of numbers. This branch of mathematics is the study of integers and relations between them. An alphanumeric identification system could easily be converted to all numerical so each identification would be an integer. Then the modified identifier could be divided by an integer and sorted into groups by the resultant remainder. An analysis of the frequency of occurrence of different remainders when divided by different integers might lead to a simple transformation formula. Another approach would be to operate on the binary representation of the identifier. If all identifiers differ in at least a certain minimum number of binary digits, this representation would have some features similar to a Hamming error-correcting code. If cases like this are found, error correcting

logic such as used with the Hamming error-correcting codes could affect a transformation of the set of identifiers into a more uniformally distributed set.

The rules of formal logic offer a second possibility for research. At present, no general syntax language is available to completely express a group of propositions for addressing. The immediate advantage of a solution in this area is that the data transformed need not be restricted to numerical data. Cryptographers quite universally include numbers, letters, and special symbols as input but develop an output symmetric in form but random in appearance. An expanded command structure large enough to execute logical instructions would be necessary for solutions in this area.

The most frequently considered attack upon the problem is through the use of statistics. The solution procedure is divided into two general parts. The first demands a proper definition of the distribution of the data being placed on the machine file. The second requires an efficient sampling scheme designed to monitor the changing data so that the most effective use of the file is being realized at any given time. The current practice of assuming a normal distribution and imposing a pseudo-randomizing scheme to the data may not always be proper for the data being processed. If some careful analysis of the data in question is made before the addressing is done, it might be discovered that certain kinds of data are more properly described by a multi-modal distribution, i.e., one that generates clusters of frequently addressed material around two or several peaks. The methodology of sequential sampling would seem to offer the most fruitful approach to describe an answer to the problem of changing data affecting a previously defined distribution. By so examining all of the changing data, it would probably be possible to construct a sampling scheme whereby an immediate warning is given when the old distribution goes out of control. The immediate problem in this area is that these techniques could become so sophisticated that the time necessary to produce results could adversely affect the processing of an application (unless the arithmetic unit is quite facile).

The three areas discussed above (i.e., number theory, formal logic, statistics) are common in their approach in that they all prescribe a micro-inspection of the file to attain their final result. A macro-inspection method would result if the file was viewed as a map and material on this space was located by area or shape rather than numerical address. At the present time, topological or geometrical transformations are not directly available to simply express such relationships. As a completely separate view of the problem, however, this method of analysis might be considered. This could conceivably result in affecting the ultimate form of the file as well as the addressing scheme.

For a full description of these techniques, see Appendix 5.

6. CONCLUSIONS AND RECOMMENDATIONS

The Committee, in examining the addressing art, has reviewed its present state and has looked into its desired capabilities for the future.

For the present, the Committee finds existing addressing techniques workable but pedestrian. They tend to be special purposed; they demand a high degree of file organization; they can exploit the information content of a file only from a single approach; they do not allow full storage utilization. To date, technological and system know-how do not permit these conditions to be otherwise. Application personnel should recognize that this impasse exists. Until the awaited break-through arrives, their most rewarding efforts will be in the direction of evolutionary improvements on current techniques, rather than in attempting radical solutions for which there is no realistic support.

For the future, the Committee envisions greatly broadened application capabilities of large memories made possible through improved addressing techniques. A break-through in addressing must not only overcome the weaknesses of present techniques, but must fulfill new application requirements, including output sorting, easy and flexible file reorganization, multiple item files, and item retrieval based on any of several identifiers. The manner in which these requirements will be satisfied remains to be determined.

For the interim, the Committee recognizes that addressing must depend upon the three current classes of techniques: Direct Addressing, Mathematical Manipulation, and Table Look-up. Of these, only table look-up is general purpose in applicability and reasonably satisfies the basic evaluation factors of retrieval time, storage utilization, file maintenance, and external manipulation. This is not to exclude the possibility that some clever combinations of the several techniques may not offer a highly effective scheme for some applications.

In the hardware area, the Committee recommends a vigorous Product Development program to develop general purpose addressing facilities using the table look-up approach. These facilities should provide:

1. A separate memory device to hold the table for addressing as well as appropriate logic to search this table independently of the processor and main storage.
2. Scanning facilities in the main storage to independently search a storage area, for retrieval of a particular item on the basis of its identifier.
3. Automatic chaining so that a main storage search can be extended to any location desired.

In the research area, the Committee recommends a program including:

1. Analysis of typical identification sets for key parameters and characteristics by which any identification set may be readily classified.
2. Determination of mathematical transformations applicable to identification sets in general and, if possible, to those amenable to the above classification process.
3. Exploration of physical phenomena and organization of elements using the phenomena to uncover practical hardware methods of performing identification transformations.

In the systems area, the Committee recommends a Product Planning study to more precisely define the system requirements of potential large file applications. Although some work in this area has been done for storage devices in development, many new requirements as mentioned above remain to be deduced. These new system requirements will greatly influence the development of addressing techniques for future random access storage devices.

APPENDIX 1

TERMINOLOGY

The purpose of this section of the report is to clarify the meaning of some of the words used in the report. In relation to information retrieval, the definition of the words, in some cases, has a different implication than in normal data processing terminology.

Access - This is the physical motion to position a read/write device with respect to the storage medium.

Address - The "address" of a record defines its physical location in storage. In most files, the address is numerical and is related to the geometry of the file.

Binary Search - Divide a sorted table (or file) into two halves; determine by one comparison in which half the required entry is; sub-divide that half into two halves; continue the "compare and divide" process until the entry is located. For an N entry table, at most $\log_2 N$ comparisons are required to locate the entry.

Bucket - A number of records associated in one area of a machine file - generally determined by the geometry of the file so that the entire area (bucket, e.g., disk track, or tape strip) can be read with one physical access operation.

File or Item File - A file consists of a group of related items; usually all of the items in a particular category.

Identification - The identification of an item is part of the indicative data in that item. It is used to define the item or relate it to something else. The item identification is a function of the system, not the machine. In some cases, the machine can control the numbers used for the identification but generally, this is not possible or desirable. (See Appendix 7)

Item - An item is a group of fields that contain information about a particular phase of a business. For example, an item might contain all the information about an insurance policy, a part, or a man. No machine restrictions, such as length, number of fields, field length, etc., are implied in the use of the word "item."

Linear Search - Search the table from one end to the other; scrutinize each table entry until the required one is found. An item file may be regarded as a table with items as entries.

Record - A record is a part of the machine file. It is normally the information that is transferred to the processor as the result of a single read instruction. The size of the record may vary in different machine files. In some cases, the record size is fixed; in other proposed files the record size is flexible, that is, the size of the records in a particular area of the file may be assigned any length within certain limits.

Scan - Scan is the physical motion of the storage medium in relation to the read/write device (after the read/write device is positioned) to reach the information desired.

Seek - Both access and scan operations combined are called "seek."

APPENDIX 2

DETAILED CONSIDERATIONS IN APPRAISING ADDRESSING TECHNIQUES

As mentioned in Section 2 of this report, there are four basic criteria - time, storage utilization, file maintenance, and external manipulation, that can be used in the evaluation of addressing techniques. Furthermore, certain characteristics of the items and machine files, and certain file functions desirable in a business data processing system affect the efficiency of an addressing technique. These are termed as secondary factors which may temper the final outcome of the appraisal. These factors and the four basic criteria are discussed in detail below.

2.1 Basic Criteria

The four basic criteria are used as yardsticks to measure the effectiveness of an addressing technique. They are the following:

A. Time Required for Item Retrieval

The item retrieval time is considered to be the average time from the instant an item is requested to the instant it is completely read out from the storage device. This time can be divided into three intervals as follows:

(1) Processor Transfer Time - the time required for the processor to make the item identification available. At the end of this interval, the identification is available for conversion into a machine address, if necessary.

(2) Association Time - the time required to associate the identification with an address. The association process, e.g. mathematical manipulation or table look-up, may involve the processor and/or storage or auxiliary devices. In case the identification supplies the address directly, no association is necessary. At the end of this interval, the machine address of the desired item is provided to the storage.

(3) Seek-Read Time - the time required for the storage to seek and completely read out the record(s) desired. At the end of this interval, the item is available to the processor.

In item retrieval by primary identification, it is preferable that the processor be involved only in the transfer of the item identification and the item itself, but not in the association process. For the storage device, preferably no more than one seek-read time should be required. A ratio of the item retrieval time to the seek-read time can be used to gauge the efficiency of addressing techniques.

B. Storage Utilization

The percentage of storage capacity utilized in storing item files may vary with the addressing technique adopted. A given technique may reduce the available storage capacity by requiring the storage of tables and programs in the machine file, or by inherently excluding certain addresses for possible association with the identifications.

It is desirable that the entire machine file can be used for item storage with no loss due to addressing techniques. In practice, however, a storage utilization of no less than 80% can be tolerated. (This can be achieved on 305 RAMAC with the randomizing method.)

C. File Maintenance

For business files, additions and deletions usually occur. Over a period of time, the net effect may be an increase in the number of items in the file (and/or in the length of some items), probably accompanied by a decrease in efficiency of item retrieval. Thus, periodically, it may be necessary to relocate the active items in the storage, purge the deleted and aged (inactive) items from the file and perhaps revise the addressing technique, in order that items can be retrieved with expected efficiency from the storage. The need of file maintenance is dependent upon file stability, i.e., the frequency of additions and deletions.

Ideally, an addressing technique should eliminate the necessity of file maintenance; for practical purposes, file maintenance should be easily accomplished by means of a stored program, with infrequent reorganization of the item file and no major revision of the addressing technique.

D. External Manipulation

External manipulation consists of the initial effort in selecting an addressing technique and the manual operations necessary to implement the technique in routine processes. To adopt an addressing technique for a particular application, usually the item identifications are thoroughly analyzed to determine the applicable techniques and these techniques are evaluated in terms of retrieval time, storage utilization, file maintenance, etc. Although the initial set-up of the best possible addressing technique may be a one-time operation, it could be time-consuming and costly.

As an addressing technique is limited by the capability of the available equipment, manual operations may be required to facilitate machine processes. For example, item identifications may be manually converted into machine-addresses for processing purposes or secondary identifications into primary ones by which items can be readily retrieved from the file.

It is desirable that the initial set-up of an addressing technique should be simple and straight-forward for customers' personnel to apply. It should not require lengthy manual or machine computations. For item retrieval by primary identifications in a normal process, no manual operation of these identifications should be involved.

2.2 Secondary Factors

The evaluation of an addressing technique can be mainly carried out in terms of the four basic criteria mentioned above. However, it will also be tempered by some secondary factors. These factors concern either the characteristics of a business file that may affect the retrieval efficiency, or the functions of a business data processing system that must be accomplished with any addressing technique.

Into the latter category fall item retrieval by secondary identifications, link reference from an item to one or more items, output in a sequence different from the one existing in the machine file, and batch processing of input transactions. Of the former are item length (fixed or variable), identification structure (numerical or alphanumeric, long or short), duplicate identifications in multiple item files, and file activities unevenly distributed among items in the file.

These secondary factors are discussed in detail below:

A. Item Length

Items in a file may not be of the same length as the machine record, and their size may not remain the same throughout the active part of their life. In case the machine record size is fixed for that part of the storage where an item is stored, and it is not equal to the item length, the retrieval operation should specify the item size in terms of record size without any loss in the retrieval efficiency. For example, it may be specified by the program processing the item or included as a part of the table function value.

B. Identification Structure and Address Distribution (See also Appendix 7)

The identification structure should be considered in the initial selection of an addressing technique. A set of short, evenly distributed numerical identifications may lend itself to direct addressing, while long unevenly distributed alphanumeric identifications must resort to table look-up or some form of

mathematical manipulation. In particular, the size of a table is dependent upon the identification length, and affects the amount of time required in a table search (to compare the identifications and to scan the table entries).

As machine addresses may not be consecutive in future storages with flexible record length, the uneven address distribution increases the complexity in the mapping of index codes within a storage device.

C. Multiple Item Files

There may be several item files in the same storage, perhaps with duplicate identifications. For example, inventory file, accounting file, and personnel file may be stored in the same device, and a part number, account number, and employee number may be exactly the same number (say 1234). Normally, duplicate identifications are handled by the particular program processing the item or by external manipulation to indicate the type of transaction or item. For item retrieval from multiple files, a number of different techniques or modifications of a given technique may be required.

D. Item Activities

It is usually the case that a small part of items account for a major portion of file activities. Thus, it is preferable to organize items in a file or their identifications in a table according to their activity so as to minimize the retrieval time for high activity items.

Where multiple access mechanisms are installed in one area of the storage device and a single access mechanism in another area, the most active items should be stored in the former area. For any technique with multiple items stored in a particular address and its overflow area, it is desirable to arrange the items in a sequence of descending activity so that less references will be made to those overflowed items.

E. Secondary Identifications

To facilitate data processing, item files are generally organized according to a primary identification. However, in practice, it may be necessary to retrieve an item by its secondary identification which may be any portion of that item. For example, a personnel file may be organized by employee number, but retrieval of an employee's record by either his name or social security number may not be infrequent.

Normally a primary identification indicates a unique item and storage location, while a secondary identification may refer to multiple items. Thus, a part number is a unique identification, but part type or assembly number identifies more than one part.

Although item retrieval by its primary identification should be performed in minimum time, facility should be provided to retrieve an item by its secondary identification. The process may be less efficient, but should be independent of processor operation.

F. Link Reference

This implies reference from one item to another (or several other items). For example, when a part is found to be out of stock in inventory updating, it may be necessary to refer to the substitutes. The item initially retrieved may supply an identification or machine address by which the related items can be individually retrieved; or reference may be made to a table which lists the identifications or addresses of all of the items in the same category (e.g., items of the same due date, assembly number, or part type).

G. Output Sorting

For output, items need sometimes be arranged in a sequence different from the one existing in the file or the one of the input transactions. For example, warehouse picking tickets must be printed by bin locations within each warehouse while incoming orders may be in random sequence, or a sales analysis report by district, product line, and salesman must be prepared from a file organized sequentially by salesman. It is preferable that output sorting can be achieved without rearranging items within the storage.

H. Batch Processing

An addressing technique should primarily lend itself to in-line processing, but also be able to take advantage of existing file organization for batch processing. For example, in inventory control, report of inventory status of items in file sequence can be periodically produced, while daily updating of inventory will be random in nature.

APPENDIX 3

REVIEW AND APPRAISAL OF PRESENT ADDRESSING TECHNIQUES

3.1 Direct Addressing

In direct addressing, the item identification is the same as the machine address. Therefore, with its identification available, the item can be directly retrieved from the storage.

The original identification may be directly used as the mach ine address (e.g., a four-digit account number may serve at the same time as the machine address where the account record is stored). If this is not possible, assignment of machine addresses as item identifications or appending to each identification its corresponding address will permit direct addressing in item retrieval.

An appraisal of this technique in terms of the four basic criteria and secondary factors is summarized as follows:

A. Retrieval Time

Since machine address is directly available, no time in the processor or storage is required for address conversion. Normally, item retrieval involves one single access to the storage.

B. Storage Utilization

This technique does not affect storage utilization. High storage utilization is possible through external control of identification assignments.

C. File Maintenance

Address assignments require the maintenance of a list of available addresses or identifications. From this list, identifications are selected for new items added to the file. In case an item grows in size in excess of allotted machine record length, the excess part will be linked to the item, but may not be retrieved directly.

D. External Manipulation

From the user's point of view, initial conversion of identifications to machine addresses may not be desirable. If the machine address is not included in the identification, manual operation is required to append machine address for direct addressing.

E. Item Length

Item and record lengths have no significant effect on retrieval efficiency. It is assumed that the ratio of item size to record size is taken into consideration in the initial assignment of addresses.

F. Identification Structure and Address Distribution

Since the address is the same as or a part of the identification, their length and distribution will not affect retrieval efficiency.

G. Multiple Item Files

No duplicate identification in multiple item files is allowed.

H. Item Activities

Item retrieval by direct addressing is independent of item activity and its organization within the storage.

I. Secondary Identifications

An item can only be directly retrieved by its primary identification.

J. Link Reference

Link reference is feasible, provided the address of the item referred to is given.

K. Output Sorting

Output sorting is feasible by conventional sorting, file searching, or use of specialized tables. Logic may be built in the assignment of identifications (machine addresses) to facilitate output in a given sequence.

L. Batch Processing

Batch processing in machine address sequence and in-line processing are possible.

3.2 Addressing by Mathematical Manipulation

The machine address of an item can be derived by mathematical manipulation of its identification. Mathematical manipulation implies randomizing, digit selecting, performing simple arithmetic on the identification, or a combination of these. The purpose of the manipulation is to transform the set of identifications into a set of addresses.

As a one-to-one transformation does not necessarily exist, duplicate addresses may result from manipulating the identifications. In such a case, multiple items may be stored in the same location (called a bucket) and overflow of the location be referred to through the chaining of one or more storage locations.

An appraisal of this technique is presented in the following:

A. Retrieval Time

Mathematical manipulation usually requires processor time in carrying out the transformation and storage time in searching for the individual item in a bucket and possibly seeking the overflow bucket.

B. Storage Utilization

The storage capacity cannot be fully utilized in view of the following:

(1) Item retrieval reaches a point of diminishing efficiency (increasing number of seeks) when the storage is packed to a certain percentage (roughly over 80%) of its capacity.

(2) Certain storage locations may not be efficiently utilized due to an uneven distribution of converted addresses.

C. File Maintenance

Additions (within the limitation of B(2) above) and deletions can be adequately handled. For file growth exceeding the number of items originally contemplated, or for alteration in identification distribution, it is necessary to modify the manipulation method.

D. External Manipulation

Initial effort in analyzing the set of identifications and selecting the method of manipulation may be excessive. However, no manual operation is required to translate identifications into machine addresses.

E. Item Length

The item and record lengths have no significant effect on retrieval efficiency. In case the two are not equal, the technique will be varied to cope with items stored as more than one machine record or multiple items stored in one machine record.

F. Identification Structure and Address Distribution

The technique is not affected by identification length or address length, although the selection of an optimal method depends upon a thorough analysis of identification and address distribution.

G. Multiple Item Files

Multiple item files necessitate the modification of manipulation method according to the type of input identification.

H. Item Activities

Items within a bucket and its overflow area can be arranged according to their activity so that average retrieval time can be minimized.

I. Secondary Identifications

An item can only be directly retrieved by its primary identification. It seems improbable that a different mathematical manipulation for a secondary identification can be devised so that the identifications of an item can be separately transformed into the same machine address.

J. Link Reference

Link reference is feasible by specifying item identification and its type. These will be manipulated to arrive at the machine address.

K. Output Sorting

Output sorting is feasible by conventional sorting, file searching, or use of specialized tables.

L. Batch Processing

Since items with their addresses generated through mathematical manipulation are not stored in any predetermined sequence, batch processing of these items does not seem more advantageous than in-line processing.

3.3 Addressing by Table Look-Up

By searching a table of item identifications and their corresponding addresses, the machine address of an item can be determined. The identifications may or may not be duplicated in both the table and item record. There may exist a single entry table or multiple entry tables, and single level or multiple level tables.

An appraisal of the table look-up method in item retrieval is given as follows:

A. Retrieval Time

Time required for item retrieval consists of time for the table searching and time for the actual retrieval. The latter is at an attainable minimum, i.e., one seek per item retrieval. The former depends on the table organization and equipment configuration.

B. Storage Utilization

Storage of tables inherently involves a space penalty. The penalty in terms of percentage of total storage capacity varies widely, depending upon table organization. Alternatively, a separate storage device may be used to house the table.

C. File Maintenance

A growth in the number of items in the file increases the size of related tables and the search time. Additions and deletions require maintenance of the tables.

D. External Manipulation

External manipulation is needed in setting up the tables.

E. Item Length

If the item length is greater than the machine record length, the table function should probably contain the number of records for the item in addition to its machine address, or the program will specify the number of machine records used to accommodate each item. In case more than one item is stored in one machine record, an indicator may be included in the table to locate the required item within the machine record.

F. Identification Structure and Address Distribution

If identifications are not duplicated in the table and item records, their length does not affect storage utilization. If they are duplicated, the greater their length, the more the loss in storage utilization.

G. Multiple Item Files

Multiple item files require as many different tables. For searching a specific table will be selected by the program in the processor according to the type of input transaction.

H. Item Activities

Table(s) may be organized according to activity, but the task may be laborious.

I. Secondary Identifications

Multiple entry table or separate tables allow item retrieval by any one identification.

J. Link Reference

Link reference is possible by means of specifying the identification and/or the pertinent table.

K. Output Sorting

Output sorting on the basis of existing tables can be accommodated. However, output sorting in a sequence different from the available tables requires conventional sorting, file searching or specialized table(s).

L. Batch Processing

Batch processing in table sequence is facilitated. The table look-up portion of the retrieval process is sequential, while the record retrieval by the storage device is direct.

APPENDIX 4

PROPOSED INTERIM TABLE LOOK-UP TECHNIQUES

4.1 Multiple Level Table Look-up

4.1.1 Description

A linear search of the items stored in a file offers a retrieval technique in which any portion of the item may be used as an identifier but requires the examination of all items in the file until the appropriate item is found. To reduce the time required for this brute force search, a table, "A", with an entry (identifier and address) for each item can be set up and only this "A" table searched. This requires extra storage capacity for the "A" table and searching the entire "A" table until the item entry is found.

By organizing this "A" table in a sequential manner, another table, "B", can be developed, much smaller than "A", to direct the entrance to a portion of the "A" table so that only a portion of the "A" table is examined to find the appropriate item's entry. This multiple level table requires more storage for the "B" table but reduces the time required to find the appropriate entry.

Consider a two-level table used for 100 items with a three character identifier ranging from 000 to ZZZ, Fig. 4.1.1. The "A" table would consist of 100 entries in identification sequence in 10 sections of 10 entries each. Each "A" table entry consists of the item's identifier and the machine file address of the item. The "B" table contains 10 entries corresponding to the last (highest) identifier in each of the 10 "A" table sections, as well as the location of the first entry in the appropriate "A" table section. The particular item would be retrieved through comparing the identifier with the various table entries in a sequential fashion. If the item "CZZ" is desired, this identifier is compared with the entries in the "B" table to determine the section of the "A" table to search.

In this case, the identifier is greater than the zero "B" table entry but not greater than the first "B" table entry. Therefore, the entry for "CZZ" is in the section of the "A" table indicated by the address contained in the first "B" table entry, table "A" section 1. The identifier is then compared with the entries in section 1 of the "A" table until an equal is found at entry number 5. This entry supplies the address in the machine file where the item "CZZ" is located (address A15).

		ENTRY NUMBER										
Table	Table Section		0	1	2	3	4	5	6	7	8	9
B		Idntfr.	AD6	D10	G34	IJ5	KLM	NOO	ORI	RAT	SIT	ZZZ
		Addr.	Tbl A-0	Tbl A-1	Tbl A-2	Tbl A-3	Tbl A-4	Tbl A-5	Tbl A-6	Tbl A-7	Tbl A-8	Tbl A-9
A	0	Idntfr.	000	123	456	A01	A02	A03	A99	ABC	ABD	AD6
		Addr.	A00	A01	A02	A03	A04	A05	A06	A07	A08	A09
	1	Idntfr.	AD7	AZ9	B01	BCD	BZA	CZZ	DO1	DOA	DOZ	D10
		Addr.	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19
	2	Idntfr.	E66	E77	E88	F30	F31	F32	F99	GOO	G33	G34
		Addr.	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29
	3	Idntfr.	G35	H00	H98	HZA	HZQ	I00	I10	I11	I12	I55
		Addr.	030	A31	A32	A33	A34	A35	A36	A37	A38	A39
	4	Idntfr.	IWX	IYY	IZZ	JOO	JPQ	JXY	JZZ	KOO	KAA	KLM
		Addr.	A40	A41	A42	A43	A44	A45	A46	A47	A48	A49
	5	Idntfr.	KNQ	KPQ	LMN	LOP	LQR	MO4	M99	MZO	MZ4	NOO
		Addr.	A50	A51	A52	A53	A54	A55	A56	A57	A58	A59
	6	Idntfr.	N01	N02	N03	N04	N05	N06	OAB	OCD	OEF	ORI
		Addr.	A60	A61	A62	A63	A64	A65	A66	A67	A68	A69
	7	Idntfr.	OSW	OTX	OUY	OVZ	POD	POQ	QAB	QCD	QER	RAT
		Addr.	A70	A71	A72	A73	A74	A75	A76	A77	A78	A79
	8	Idntfr.	KAZ	RBC	RDE	RFG	RH9	SOO	SOA	S11	S1A	SIT
		Addr.	A80	A81	A82	A83	A84	A85	A86	A87	A88	A89
	9	Idntft.	SZI	TAP	URT	VIP	WAR	XIP	YOU	YYY	ZAP	ZZZ
		Addr.	A90	A91	A92	A93	A94	A95	A96	A97	A98	A99

Two Level Table Fig. 4.1.1 Scan Entries in This Direction →

The average search required with this two-level table is 10 entries compared with 50 for a linear table search. It should be noted that the entries within a section of the "A" table can be in any sequence provided they are within the range specified by the "B" table (e.g., the entries in section 1 of the "A" table can be in any sequence provided the lowest identifier is greater than AD6 and the highest is not greater than D10).

The number of table levels can vary and in practice will be determined by the number and length of the items stored, the bucket capacity, and the machine file geometry. Additional table levels are indicated here as "C" for the third level and "D" for the fourth level table.

This technique requires the scanning of a section of each table level and may also require an access for each table level. The number of access operations may be reduced through repeating the first table levels in various sections of the file.

The technique proposed here is to eliminate the storage requirements for the "A" table entries through considering the items themselves as the "A" table. For example, in Fig. 4.1.1, the "A" table would not contain the identifier-address entry but the item (identifier and data). When the comparison is complete at the "A" level, the item has been found rather than the location of the item. An "A" table for the Advanced File with twenty 100-character records/track and 30,000 tracks would require 600,000 entries of from 20 to 30 characters each - therefore, a total of 12 to 18 million characters. With the proposed approach, this 12 to 18 million characters is not required for an "A" table but is available for storing items. "B" and "C" level tables for the Advanced File would require less than a million characters of storage. (Less than 2% of the file's capacity.)

This technique can be used with the Advanced File and VLCM (for functional description see Appendix 6) through programming, but would be quite time consuming for the file as well as the processor. Additional file functions, such as "independent bucket search," could make the technique quite efficient as well as automatic.

The comb type access mechanism provides facilities to use this technique with the Advanced File and retrieve an item with but one access operation and three table section scans. This can be accomplished by:

A. Repeating the "C" table at each comb access position (250 tracks).

B. Recording at each comb access position (track) the section of the "B" table associated with that track. The section of the "B" table would indicate the identifier range of items stored on each surface in this comb access position (track).

C. The "A" table section consists of the items stored on each track surface or bucket.

Whenever an item is to be retrieved, a reference is made to the "C" table (available at each comb access position) which indicates the comb position, track number, where the item is located. The comb access is moved to this position and the "B" table section at this position indicates the surface with this track number upon which the item is located. This surface is searched for the item.

This approach is not as attractive as indicated above because:

A. The track format (number of records and their lengths) may vary from one track to the next, making the "C" and "B" table section searches complicated.

B. The "C" table requires 250 entries and the "B" level table requires a maximum of 120 entries. Each entry may require from 20 to 30 characters for the identifier and address. This means 3 or 4 surfaces are required for the "C" table and at least two surfaces are required for the "B" table.

For these reasons, less complicated multiple level table techniques may be developed, using additional hardware. One technique uses a section of one Advanced File module to store the "C" and "B" tables and requires a second comb access covering the particular module. Another method uses an extra surface of each module, with a fixed format, for the "B" table, and a fixed comb access covering a "C" table on a track of one module.

Two access operations are required if this technique is used with the VLCM. A four-level table can be used. The "D" and "C" tables require the capacity of two tape strips to indicate the tape strip where the item is located. This table can be repeated in each bin to reduce access time to this table. The "B" table section associated with each tape strip is stored on the tape strip and indicates which channel of the strip the item is located upon. The "A" table, a tape strip channel, contains the items.

For either the Advanced File or VLCM, this multi-level table look-up technique can be used conveniently and automatically with an additional storage device, such as a magnetic drum, with appropriate storage capacity, speed, and logic.

4.1.2 Evaluation of the Proposed "Multi-Level Bucket Table Look-up Method"

A. Retrieval Time

The processor is only required to transfer the identification to the addressing device while the only machine file time required is to seek the item. The addressing device

searches for the bucket address independently of the processor and machine file, thus all three operations can take place simultaneously, minimizing the average retrieval time. To handle multiple access devices as well as minimize the total series time (processor, addressing device, machine file) for retrieval, the addressing device should find the bucket address faster than the machine file's effective seek time (average seek time divided by number of access mechanisms).

B. Storage Utilization

Utilization of storage can be 100% but in practice will be less due to item file stability.

C. File Maintenance

The tables may be initially set up with file stability in mind; therefore, table alterations will occur only at the bucket level (when items are added or deleted). An overflow area can be provided for a number of buckets in case additions create bucket overflows. Periodically, completely dependent upon the file's stability, the items should be rearranged to allow better packing and space distribution for additions. This item rearrangement would also require reorganization of the multi-level tables.

D. External Manipulation

Initial set-up should be almost automatic provided the item files are in primary identifier sequence in a machine readable form. Some set-up time may be required to determine the space requirements for additions and file growth. Manual look-up operations are required only for input or inquiry operations referring to secondary identifiers.

E. Item Length

Direct retrieval will be limited to one record (record length may vary from bucket to bucket). Link referencing may be used to combine portions of items placed in more than one record.

F. Identification Structure

The table size is directly related to both the identifier and address size. A general purpose addressing device would probably limit the combined identifier and address length to 20 or 30 characters. The distribution of identifiers has no effect upon the retrieval technique.

G. Multiple Item Files

Multiple item files can be handled through the use of multiple tables (the processor selects the appropriate table for the individual item).

H. Item Activity

Look-up time is independent of the item's activity but items within a bucket and its overflow area can be periodically arranged according to activity.

I. Secondary Identifications

Multiple identifications cannot be handled by the proposed bucket table technique unless the bucket contains a complete secondary identifier to primary identifier conversion table. This would require a second look-up operation on the primary identifier and storage for secondary identifier conversion table. Items with common secondary identifiers can be referred to through use of link reference or bucket search techniques.

J. Link Reference

Link referencing can be accomplished through use of either the chaining, searching, or separate table techniques.

K. Output Sorting

Special tables, links,, bucket searches or standard sorting techniques can be used for output sorting.

L. Batch Processing

A batch process on the basis of the primary identifier is facilitated through placement of the items within an identifier range in a bucket even though the items in this bucket are not necessarily in sequence.

4.2 Modified Binary Search of a Sorted Table

4.2.1 Considerations

The particular table look-up method is proposed in view of the following considerations:

A. Binary search of a sorted table takes less time than linear search (maximum $\log_2 N$ vs. N comparisons where N is the number of items in the table). For example, if there are 512 items in the table, binary search requires a maximum of 9 comparisons while linear search a maximum of 512 comparisons.

B. To reduce maintenance effort of a sorted table (and file) table arguments should be "ranges" for item identifications. Ranges may be arbitrarily determined (not necessarily equal), to counter-act the effect of gaps in the identification numbering system, and to provide rooms for file growth.

C. Items with identifications within a certain range are to be stored in the same bucket in the machine file so that additions and deletions will not affect the table. Searching for equal within the bucket retrieves the item with the given identification. The address of the overflow area of a physical bucket can be stored in that bucket to facilitate chaining.

D. Binary search of a sorted table can automatically generate a binary bucket address to select the bucket. For the same number of buckets, a binary address requires less bits than a decimal address (e.g., 32,768 buckets can be addressed with 15 bits, while decimally with at least 20 bits, 5 binary coded decimal digits).

4.2.2 Proposed Method

The proposed table look-up method involves a modified binary search of a sorted table that automatically generates a bucket address, and a linear search of item identifications within the bucket to locate the particular item. The table contains $2^N - 1$ groups of 2^M identifications (limits) each (N and M are positive integers and $2^M(2^N - 1)$ the total number of buckets assigned to the item file).

A binary search of the $2^N - 1$ groups selects the proper group in N or less comparisons and generates the first N bits of the bucket address. A linear search of the selected group determines the range into which the identification falls and generates the next M bits of the bucket address by counting the unsuccessful comparisons. Another linear search within the bucket retrieves the required item.

To set up the table, the identifications are sorted in ascending sequence and divided into $2^M(2^N - 1)$ groups for storing in the file buckets. The number of items stored in each bucket will not be uniform so as to accommodate expected additions or deletions. The ranges of these groups, therefore, will not necessarily be equal and the range limits selected may not be identifications in existence.

Assuming a drum is to serve as a storage for the table of range limits, these limits are stored on the drum, 2^M limits per track (depending upon the capacity of the track and length of the identification), with the lower limit (same as last limit on the preceding track) and upper limit (same as last limit on the present track) stored at

the beginning of the track. There will be $2^N - 1$ tracks on the drum for the given table. The tracks will start from different physical locations on the drum (in a manner to be described in detail below).

The table searching will proceed through the following steps:

A. The identification of the desired item is available for comparison with table arguments.

B. Start search at the track in the center (i.e., 2^{N-1} th track).

C. Compare the given identification with the lower limit recorded on that track. If the identification is less than the lower limit, make the first bit of bucket address 0, and switch to 2^{N-2} th track, continuing the process (repeat steps C, D, and E). If the identification is greater than the lower limit, generate a "1" as the first bit of bucket address and compare it with the upper limit.

D. If the identification is greater than the upper limit, switch to $(2^{N-1} \nearrow 2^{N-2})$ th track, repeating steps C, D, and E.

E. If the identification is less than (or equal to) the upper limit, it lies within the limits on this track, the remainder of the first part of the bucket address (N bits) will be zero and comparison with each limit on the track proceeds. Each "greater than" counts as 1 (accumulated) and the process stops when the identification is less than or equal to a certain limit. This generates the second part of the bucket address (M bits) and ends the table search.

F. While the bucket address is being generated, the access mechanism can be actuated to select the particular bucket thus overlapping part of the access time with the table search time.

G. The identifications of items stored in the bucket will be recorded in a particular location (e.g., track, channel) in the bucket. They are successively compared with the given identification and a binary count is kept. When equality is found, the count gives the address of the required item in the bucket, and the read/write mechanism is switched to locate the item. In case of inequality, search the overflow bucket. If there is no overflow bucket, then the required item is not in storage.

4.2.3 An Example

As mentioned earlier, the table tracks are arranged in a staggered manner so that the search can be accomplished in less than two drum revolutions. The layout of the table may be best illustrated

by an example. Suppose there are seven tracks on the drums and eight limits per track. Then the table on the drum will be arranged as shown in Fig. 4.2.1, assuming that switching from one track to another can be done in one-two word (sector) time.

Search starts with Track 4. After reading the marker (*) indicating the beginning of a track, the given identification is compared with the lower limit in Track 4. If it is less than the lower limit, an "0" is taken as the first bit of bucket address and switching to Track 2 is effected. In the same drum revolution, the search process is continued through Track 2, and, if necessary, switched to Track 1 or 3. This is made possible by the fact that the beginning (home position) of Track 2 is located three words (sectors) apart from the marker of Track 4 (to the right in the diagram), and that of Tracks 1 and 3 similarly located with respect to Track 2. Therefore, less than two drum revolutions are required to generate the bucket address.

If the given identification is greater than the lower limit in Track 4, a "1" is generated as the first bit of the bucket address, and the identification is compared with the upper limit. If it is less than or equal to the upper limit, it will be compared with the limits on this track and the remainder of the first part of the bucket address will be zeros. If it is greater than the upper limit, a switch to Track 6 will be effected and the search continues in the manner as previously described.

When the given identification is compared with the limits in a track, a binary count is accumulated for each "greater than" comparison and the process ends with a "less than" or "equal to" comparison. The count is used as the second part of the bucket address. For example, when the given identification is compared with limits in Track 5, and is found to be less than limit #35, the count is 010. Therefore, the bucket address for this item is 101 010 (where 101 is generated through comparisons with lower limits in the tracks).

(The last limit on each track may not be necessary, since it is the same as the upper limit. The comparison ends when the marker is detected or a count of seven is reached.)

It is recognized that bucket addresses with first N bits equal to zero are not used in this addressing scheme. In a large capacity storage, these buckets will be a small fraction of the total storage. It may be used as an overflow bucket or as storage for programs, working space, etc.

4.2.4 Applied to Devices in Development

The table searching and bucket address generating should be automatic, independent of the processor. It is desirable that multiple access mechanisms will be provided so that it will be possible to seek and read the following item while the current item is being processed.

Track	Part of Bucket Add.	← Drum Revolves Toward the Left									
1	001	Limit #2	Limit #3	Limit #4	Limit #5	Limit #6	Limit #7	*	Lower Limit =000...0	Upper Limit = Limit #8	Limit #1
2	010	Limit #13	Limit #14	Limit #15	*	Lower Limit = Limit #8	Upper Limit = Limit #16	Limit #9	Limit #10	Limit #11	Limit #12
3	011	Limit #17	Limit #18	Limit #19	Limit #20	Limit #21	Limit #22	Limit #23	*	Lower Limit = Limit #16	Upper Limit = Limit #24
4	100	*	Lower Limit = Limit #24	Upper Limit = Limit #32	Limit #25	Limit #26	Limit #27	Limit #28	Limit #29	Limit #30	Limit #31
5	101	Limit #33	Limit #34	Limit #35	Limit #36	Limit #37	Limit #38	Limit #39	*	Lower Limit = Limit #32	Upper Limit = Limit #40
6	110	Limit #44	Limit #45	Limit #46	Limit #47	*	Lower Limit = Limit #40	Upper Limit = Limit #48	Limit #41	Limit #42	Limit #43
7	111	Upper Limit = ZZ..Z	Limit #49	Limit #50	Limit #51	Limit #52	Limit #53	Limit #54	Limit #55	*	Lower Limit = Limit #48

Fig. 4.2.1 A Sample 7 X 8 Table Layout

This scheme can be applied to both VLCM and Advanced File, provided addresses in these units are organized in a binary fashion and a drum and necessary circuitry are attached to these units for automatic addressing.

For VLCM type storage with 128 strips per cell, 16 cells per bin, and 16 bins per unit, the 32,768 strips can be addressed with 15 bits. On each tape, there may be one or two channels for storing identifications of items in the bucket, and eight channels for storing the items. These eight channels may be staggered in a similar manner as the drum tracks so that less than two channels should be scanned to read the item. A drum for the table look-up may contain 511 tracks, each track storing 64 limits (plus lower limit and marker). Assuming the limits of 20 characters in length, the drum capacity will be 674,520 characters.

For an advanced disk storage with comb type access arms, it may consist of four modules, each with 32 surfaces. There will be 256 tracks per surface. Thus track address will have ten bits. Within each track, five bits address the surfaces and five or less bits the records in the track on a surface. Depending upon the capacity of the drum the same tracks on adjacent surfaces may be grouped into a bucket. The auxiliary table and records in a bucket will be so arranged that a record can be read out in less than two disk revolutions.

To facilitate binary addressing, it seems that there should be 2^n($n = 0, 1, 2 \ldots$) records within a channel or a track, and record length will be a multiple (in powers of 2) of 80 characters. A 10,240 character channel on a strip or 2,560 character track on a disk surface will allow records of 80, 160, 320, 640, 1,280, 2,560, . . . characters in length.

4.2.5 Evaluation of the Proposed Method

A. Retrieval Time

The processor needs only to transfer the identification to the addressing device. Association time and read retrieval time will overlap. With the drum search, access, and "scan and read" time in balance, and the provision of multiple access mechanism, time required of the storage to retrieve an item will be minimized.

B. Storage Utilization

Utilization of storage will be high, with the usual provision for additions. In case of multiple tables (and tables for secondary identifications), a part of the storage may be assigned for storing them. Although certain buckets will not be reached by table look-up, they can be used as overflow buckets, program and table storage, etc.

C. File Maintenance

Since the table is set up in terms of ranges, a careful derivation of initial values should eliminate frequent maintenance of the table. Normally, deletions will be tagged and additions placed in the bucket, following the existing records. Periodically, the storage will be purged of the deletions and rearranged for the additions (according to activity, if desired).

D. External Manipulation

Initial set-up will not take much time since it involves only the sorting of existing identifications in any ascending sequence and selecting of appropriate ranges. In daily processing, no manual operation in addressing is required.

E. Item Length

Item retrieval will be limited to one record, which will be of definite length within a bucket, but variable for different sections of the storage. Items of indeterminate length may be obtained by chaining the records.

F. Identification Structure and Address Distribution

Identification length cannot be greater than a predetermined limit (say 20 characters). Since the binary address is generated by table look-up, not stored in the table, it has no effect on retrieval efficiency, except on table organization.

G. Multiple Item Files

Multiple files require multiple tables. Search of a particular table is controlled by the processor program.

H. Item Activities

Programs can be written to rearrange items in a bucket according to their past activities. This may be done as a part of file maintenance.

I. Secondary Identifications

Multiple identifications are handled through additional tables for secondary identifications. Two table look-ups are required - one converts secondary to primary identification and the other primary identification to bucket address. For multiple items with the same identification, addressing is possible through table look-up or chaining.

J. Link Reference

Link reference will be automatic (by special registers) or program controlled.

K. Output Sorting

Output sorting will require the use of a table (or list). The drum may be used to sort items physically.

L. Batch Processing

Since there will be no ordering within the bucket, batching in primary identification sequence will not realize great advantage (through programming, access time may be reduced).

4.2.6 Miscellaneous Possibilities

For record retrieval by a secondary identification, it is necessary (if done automatically) to have a table of secondary identifications similarly organized. Search of the table gives the bucket where individual secondary identifications and their corresponding primary identifications are stored. From the latter, the machine address is derived by a second table look-up.

It may be necessary to transfer blocks of information between the drum and file storage, and from one section of the file storage to another. Data on the drum may be altered by the processor (to update table if necessary). If several tables are stored on one drum, search of a particular table will be under program control.

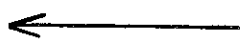
It should be possible to retrieve the item with their machine address. The contents of address register(s) of the storage can be manipulated by programming. A set of instructions may contain search-read (write), read (write) with absolute address, search bucket defined by identification or machine address.

The binary address of a bucket generated through the table look-up method may be automatically translated into decimal form, if desired. However, since item storage and retrieval will not require manual manipulation under the proposed method, it seems less likely that the user should be concerned with the number system of internal machine addresses. Should he ever want to address a bucket directly, it is possible for him to use a decimal address which will be converted by a utility program into binary form.

To apply the binary search principle to a decimal-type table requires complicated logic in the searching mechanism (switching of track search no longer proceeds in a symmetrical manner). The bucket address may be generated by arithmetic (adding or subtracting a different integer as track search is switched) or by code conversion (converting the binary address generated into decimal address). As the number of tracks is increased (say from 10 to 100) the process must be completely revised.

For "decimal" search, it seems a multi-level table has certain advantages. A search of the first level determines the particular second level to be searched, and the process may be repeated for

*	Limit #10	Limit #20	Limit #30	Limit #40	Limit #50	Limit #60	Limit #70	Limit #80	Limit #90
Limit #8	Limit #9	*	Limit #1	Limit #2	Limit #3	Limit #4	Limit #5	Limit #6	Limit #7
Limit #17	Limit #18	Limit #19	*	Limit #11	Limit #12	Limit #13	Limit #14	Limit #15	Limit #16
Limit #26	Limit #27	Limit #28	Limit #29	*	Limit #21	Limit #22	Limit #23	Limit #24	Limit #25
Limit #35	Limit #36	Limit #37	Limit #38	Limit #39	*	Limit #31	Limit #32	Limit #33	Limit #34
Limit #44	Limit #45	Limit #46	Limit #47	Limit #48	Limit #49	*	Limit #41	Limit #42	Limit #43
Limit #53	Limit #54	Limit #55	Limit #56	Limit #57	Limit #58	Limit #59	*	Limit #51	Limit #52
Limit #62	Limit #63	Limit #64	Limit #65	Limit #66	Limit #67	Limit #68	Limit #69	*	Limit #61
Limit #71	Limit #72	Limit #73	Limit #74	Limit #75	Limit #76	Limit #77	Limit #78	Limit #79	*
*	Limit #81	Limit #82	Limit #83	Limit #84	Limit #85	Limit #86	Limit #87	Limit #88	Limit #89
*	Limit #91	Limit #92	Limit #93	Limit #94	Limit #95	Limit #96	Limit #97	Limit #98	Limit #99



Drum Revolves Toward the Left

Fig. 4.2.2

A Two-Level Table of 99 Entries

several levels. For an N level table with staggered tracks, a maximum of N drum revolutions is required. The track switching and address generating, however, will be accomplished in a straight-forward manner. A two-level table to address 100 buckets is depicted in Fig. 4.2.2.

4.3 Indexing Scheme Using an Unsorted Table

4.3.1 Description

The addressing technique functionally described here has the following characteristics:

A. It is predicated on table look-up; i.e., indexing.

B. It uses a separate storage device to house the table.

C. It is general purpose in applicability.

D. It reasonably satisfies the four basic evaluation factors of retrieval time, storage utilization, file maintenance and initial set-up.

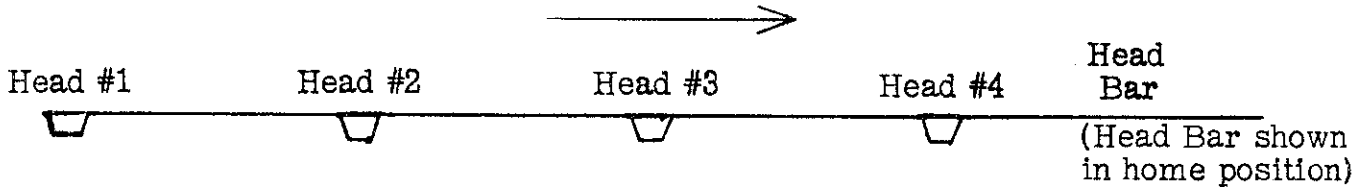
E. It requires a low level of table organization; table elements are grouped but not sorted.

The technique envisages use of a table recorded in the storage device physically separate from the main storage. The information must be disposed along closed loops or tracks, either radially parallel (e.g., disks) or axially parallel (e.g., drums). Associated with these tracks is a set of ganged read/write heads mounted on a bar which may be physically shifted in track increments. The unit of information is an index which is defined here as consisting of an argument (the coarse item identification) and the function (the main storage address).

For illustration, consider Fig. 4.3.1, a simplified example of a 12-track configuration with a capacity of 600 indices.

The head bar carries four heads. On every search the bar starts at home position but is capable of shifting two increments to the right, a track pitch at a time. At the conclusion of a search, it returns to home position.

Each track contains 51 addressable sectors, one R index (Fig. 4.3.2), followed by 50 r indices (Fig. 4.3.3). R represents a "Track-High Index" (THI) and contains simply the high-limit for arguments in that track. An r represents a "bracket index" (BI), and contains as co-arguments, the high limit and low limit of a range and their function, the associated main storage address.



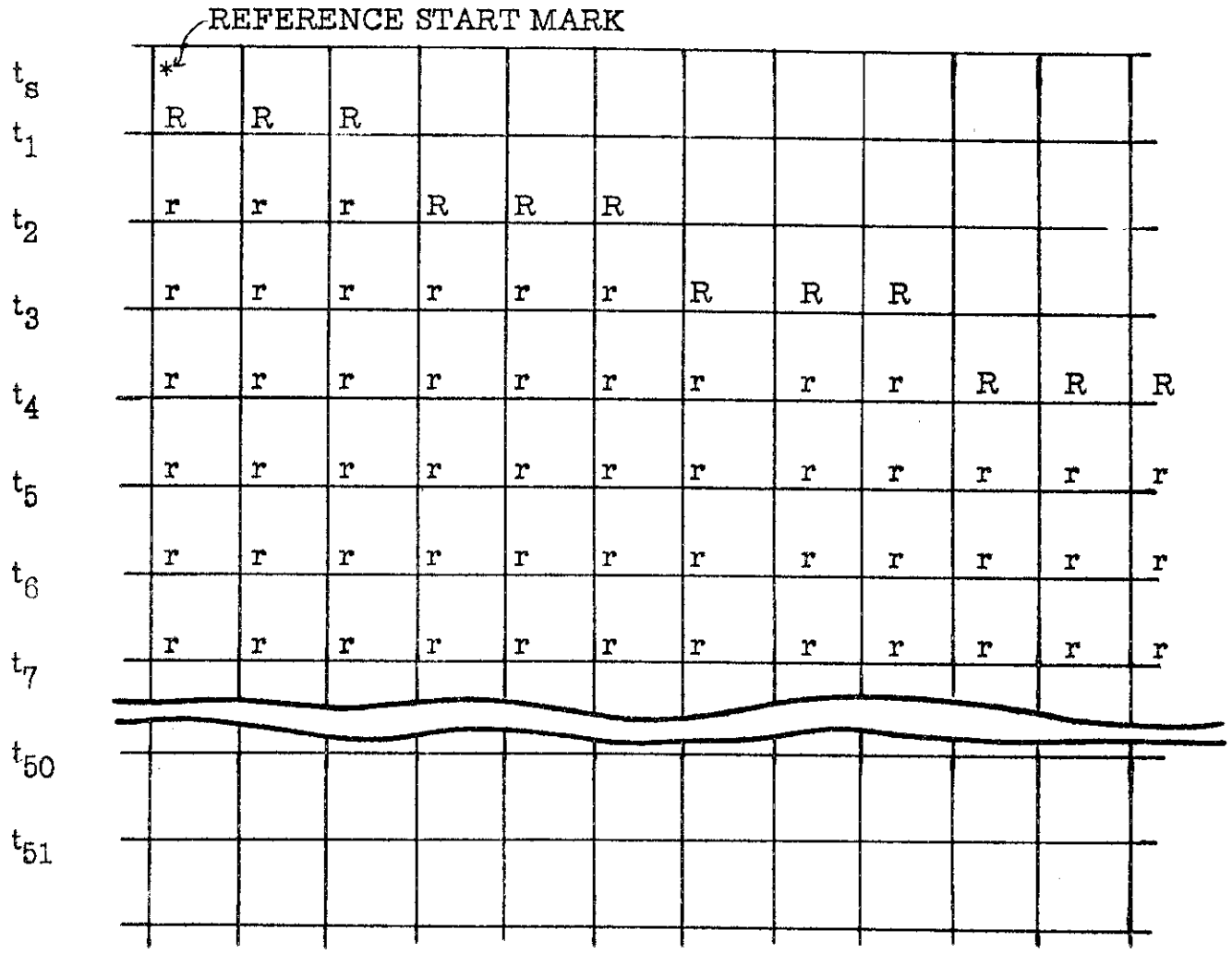


Fig. 4.3.1

Example Track Layout

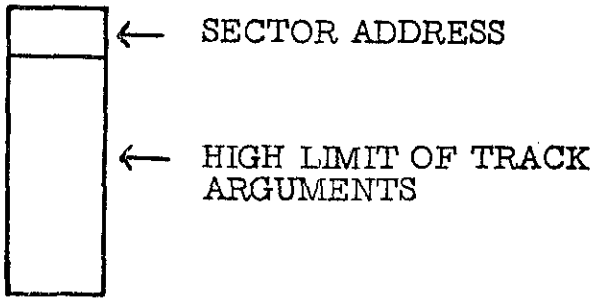


Fig. 4.3.2
Track-High Index

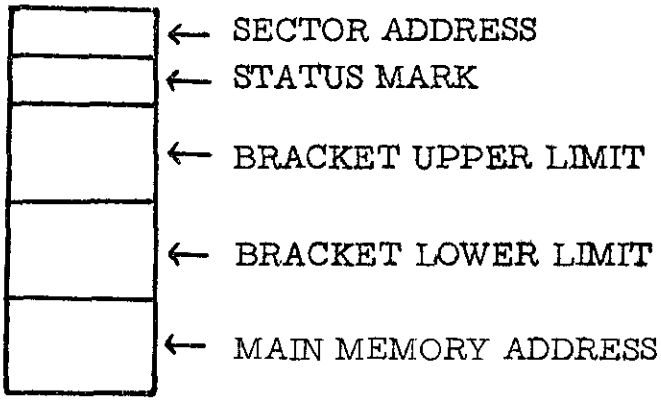


Fig. 4.3.3
Bracket Index

The mapping arrangement disposes the R's in a helical fashion around the recorded surfaces to time-optimize the coarse (i.e., THI) search. There are as many banks of R's as there are heads. Rotation-wise, all searches are initiated at the Reference Start Mark.

A typical search is as follows: an item identification is entered in the search register. Detection of the Reference Start Mark initiates the search. Two stages follow: the coarse search to converge on the track and the fine search to pinpoint the bracket. Thus, the first stage would compare the input against the R indices until a low-compare is indicated. The second stage would execute an inclusion-search within the track selected until an inclusion-compare is encountered. At such time, the associated function, the main storage address, is read out to the main storage address register.

For example, if the desired bracket index is in track 2.2, the search would have head #1 first examine R in track 1.1, switch to head #2 to examine R in track 1.2, then head #3 in track 1.3, then head #4 in track 1.4. The head bar is now shifted one track increment to the right. Head #1 tests R in track 2.1. Head #2 next finds a low-compare with R in track 2.2. Head #2 remains selected. An inclusion search finds the proper bracket index.

4.3.2 Evaluation

The technique described may be scrutinized in the light of the four basic evaluation factors:

A. Retrieval Time

An extra revolution time for head bar shifting may or may not be required, depending on several factors. If the time interval between the last bank of R's and the next Reference Start Mark is safely in excess of the head bar shift time, then the latter may be buried. Otherwise, an extra revolution time is necessary. This is, of course, related to the ratio of the number of heads to the number of indices per track, the head bar shift time, and the revolution time of the device.

The average and maximum address transformation time, T, may be expressed analytically, using the notation of: t for one revolution time in milliseconds; h for number of heads; n for number of tracks. (n is always an integral multiple of h.) Case A refers to the situation where head shifting incurs one extra revolution; Case B, where shift time is buried.

Maximum Time

Case A $T = 2t \neq 2t \left(\frac{n}{h} - 1\right)$

Case B $T = 2t \neq t \left(\frac{n}{h} - 1\right)$

Average Time

Case A $T = t \neq t \left(\frac{n}{h} - 1\right)$

Case B $T = t \neq \frac{t}{2} \left(\frac{n}{h} - 1\right)$

To bring the addressing time into practical focus, some current memory developments may be discussed. Suppose it is contemplated to use the San Jose High Performance Million Character File (HPMCF) as the indexing device for the San Jose Very Large Capacity Memory (VLCM). The problem is to associate an item identification with one of 10,000 VLCM bucket addresses.

The HPMCF has 50 heads on its head bar which shifts to one of 10 positions to cover 500 tracks. Revolution time is 7.5 ms. Head bar shift time is 3 ms. track-to-track. The addressing time would be:

Maximum

Case A 150 ms.

Case B 82.5 ms

Average

Case A 75.0 ms

Case B 41.2 ms

These average time values are several times superior to what is now experienced by mathematical manipulation on the 305. Additionally, the subject technique does not tie up any processor time. As an independent device equipped with low-level logic, its short address transformation time may generally be simple to bury under process or main storage seek time.

B. Storage Utilization

The existence of the table index in a separate storage device frees the main memory from space considerations for the addressing problem. Coupled with the fact that machine addresses may be arbitrarily assigned, the potential usage of bulk memory is 100%.

C. File Maintenance

The philosophy on file maintenance in the subject technique is to disturb only those items which are directly affected in a maintenance activity. To realize this, random placement of items is made within a track.

This is in contrast to the conventional approach to file maintenance. In the latter case, items in an area are placed in perfect collating sequence. A new entry would call for finding the proper position for insertion and displacing, by one position, all old items to one side of this position. Similarly, a deletion would require a mass closing of ranks.

To illustrate the random philosophy, file maintenance may be discussed under the headings of addition, deletion, change, bracket explosion and track explosion.

Deletion is handled by inserting the argument of the deletion item in the search register and searching for an equal-compare. A counter or equivalent device remembers the matching sector address. The latter is fed back into the address register and on the next revolution converts the status indication (see Fig. 4.3.3) of the relevant sector from Full to Empty. The index record proper is left undisturbed.

Addition on a track is also done in two revolutions after the track is found. On encountering the first empty status indicator, operation is switched from read to write (for the new record). On the second revolution, the status indication of the sector is changed from Empty to Full and read-check follows.

Change of an index record is in effect a modified combination of deletion and addition.

Bracket explosion is the process of splitting an existing bracket index into two. This usually comes about when abnormal growth in localized areas of the bulk memory call for extension of a logical bucket beyond a reasonable manageable size. The remedy is to regroup the items into two narrower range brackets. Maintenance is executed in two stages. First, the argument of the old bracket index is placed in the search register. An equal-compare develops the sector address.

The latter is fed back to the address register and the first of the new bracket indices is written in the old location. The second new bracket index may now be entered as a normal addition item.

To minimize the expansion-type of maintenance, spare areas should be planned at the lower end of the tracks rather than as spare tracks at the far end of the track group. By confining maintenance to single tracks at a time, unaffected indices are not disturbed at all. Where unexpected developments cause a track to overflow, however, track explosion procedure is necessary. If spare room exists in the adjacent track, it is preferred that the disturbance be confined to two tracks only. Contents of both tracks would have to be read out, regrouped and reloaded according to newly assigned Track-High Indices.

D. Initial Setup

Before loading the indexing device for the first time, a sequence ordering of original items would have to be made. Manually, or by a suitable program in a data processing machine, bracket ranges may be associated with bulk memory addresses. The assignments take into consideration the gaps and clusters (and the varying density within these clusters) present in the population of items.

For loading, all indices pertaining to a track would be grouped together on a continuous medium, such as tape. First would be the Track-High Index, followed by the bracket indices. A Track-End symbol would separate track groups and would be the signal to cause the track address register to advance. Within each track, the THI is written in a specified sector to conform with the helical pattern desired and the bracket indices in the succeeding sectors.

APPENDIX 5

PROPOSED MATHEMATICAL TECHNIQUES

There are four basic analytical manipulations that give promise of transforming an identification into a machine address with a low percentage of overflows. For definitions of the mathematical terms used in this Appendix, refer to Glen James and Robert C. James, MATHEMATICS DICTIONARY, NY, D. Van Nostrand, (1944).

5.1 Arithmetic Operations Based on Statistical Distribution

Various arithmetic operations are already in use which convert an identification into a pseudo-random number. The present procedure is to try several systems and then choose the one which gives the least number of overflows. It is desirable to develop a shorter system of analyzing the identification number system to determine which technique is best without trying all of them. To illustrate a present system of selecting a randomizing technique, five different systems which are commonly tried are listed below:

A. Squaring the identification number and selecting a group of digits in the middle of the resultant number.

B. Splitting the identification number into two parts and adding to obtain address.

C. Multiplying the identification number by a prime number of two or more digits and selecting an appropriate group of the resultant digits as the machine address.

D. Division of the identification number by a selected segment of the number and selecting a group of the resultant digits for the machine address.

E. Combination operation upon the identification number X, such as making the address R as follows:

$$R = X^2 + aX$$

where a is a constant.

An advance over the present procedure would be the establishment of some criteria on the probability distribution of the usage of possible identification numbers which would indicate which formula would be better, without trying each one.

An over-simplified example - a single cluster identification distribution - is shown in Fig. 5.1. Here 980 identification numbers approach a Gaussian distribution in the range of 8000 to 10,000. The plot on arithmetic probability paper shows how the identification can be transformed to stay within a range of 1,000 machine addresses by use of the Gaussian probability distribution formula (shown graphically).

In Fig. 5.1, the histogram, Curve H, shows the percentage usage of the blocks of 100 identification numbers. For example, the block 8,600 - 8,699 has 50% usage. Curve P shows the corresponding probability distribution curve (step function where detailed distribution within blocks is omitted). For example, the probability that the identification number over a long run of trials is greater than 9,200 is 26% (upper scale). The step-function can be approximated by a straight line, curve S. An identification number distribution of curve H consisting of 990 identifiers is converted to a straight line distribution by projection onto curve S to the "Address" scale. For example, identification 8,620 is mapped onto address 174 and is shown by point D. This would still have 50 overflows. Increasing the address scale by 10% would reduce the overflows to 32. The overflows are cross-hatched for addressing and the overflows for 1,100 addresses are the whole areas on curve H which are enclosed by dashed lines. The numbers on the left and right of the overflow blocks are the overflows respectively for 1,000 and 1,100 address.

A numerical representation of the utilization of the possible identification numbers by means of a conditional probability tree is shown in Fig. 5.2. As the identification tree is carried to include more digits, it approaches a complete listing of all the identification numbers.

With i representing the first $h - 1$ digits of the identifier I, the conditional probability of the h - th digit being the symbol j is $P_{hj}^{(i)}$.

The information contained in the histogram of Fig. 5.1 can be put in the form of a utilization tree as in Fig. 5.3. In this case, the tree only has $h = 1$ and $h = 2$ columns, because the histogram is in blocks of numbers.

Another possible way to analyze identification numbers is to prepare a matrix of the probability of occurrence of different digits in different positions, as in Fig. 5.4.

The matrix representation loses the conditional probability of the tree-type representation so it would be limited in usefulness to determining which rearrangement of digits would give a better distribution of identifications. A computer program could be developed to determine the closeness of fit of different probability density functions to the different columns of the matrix as a short-cut to determining which arithmetic manipulation is the most useful.

The following references are available on the basic statistics and probability theory:

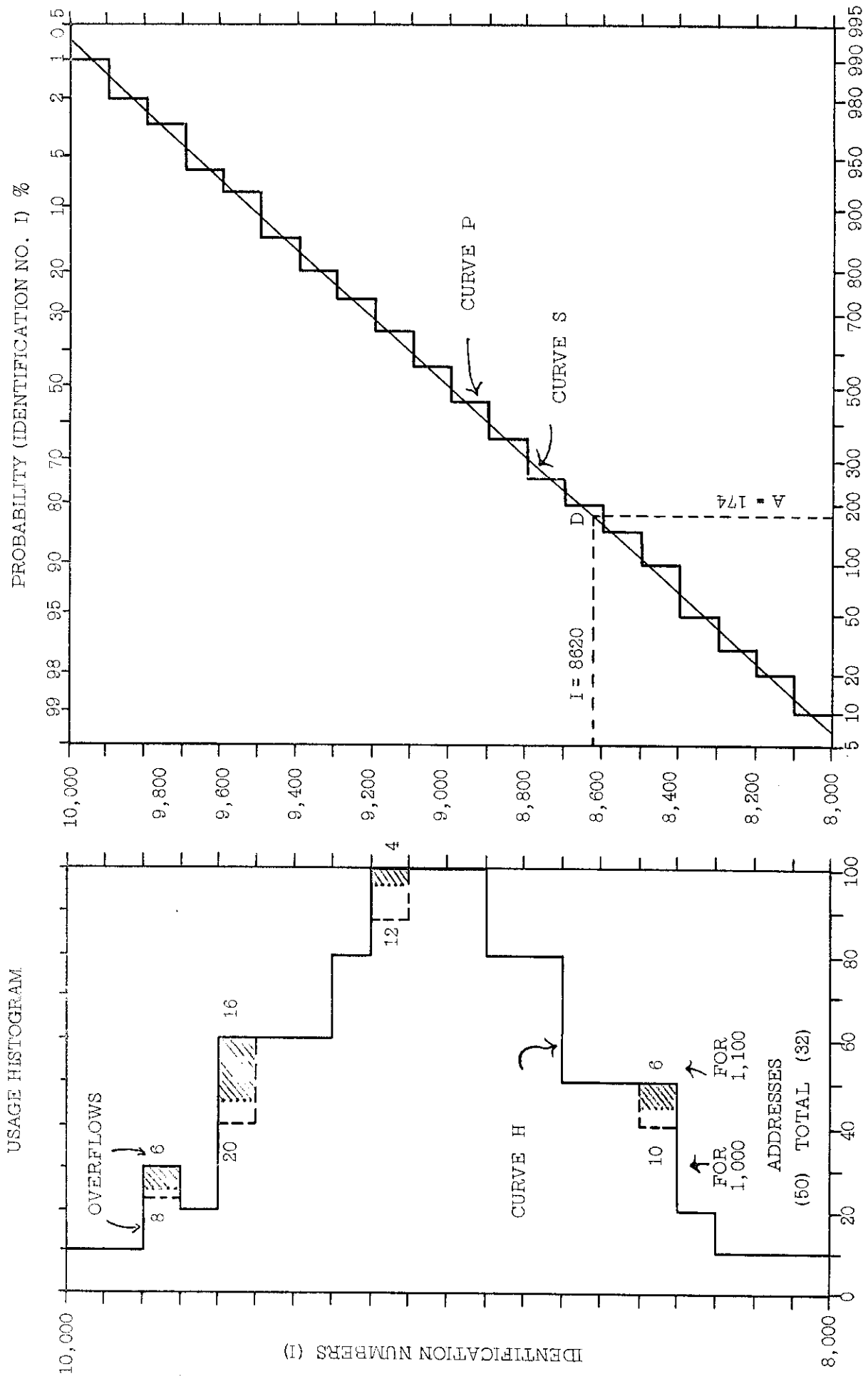


Fig. 5.1 - HISTOGRAM AND PROBABILITY DISTRIBUTION FOR A SINGLE CLUSTER OF IDENTIFICATION NUMBERS

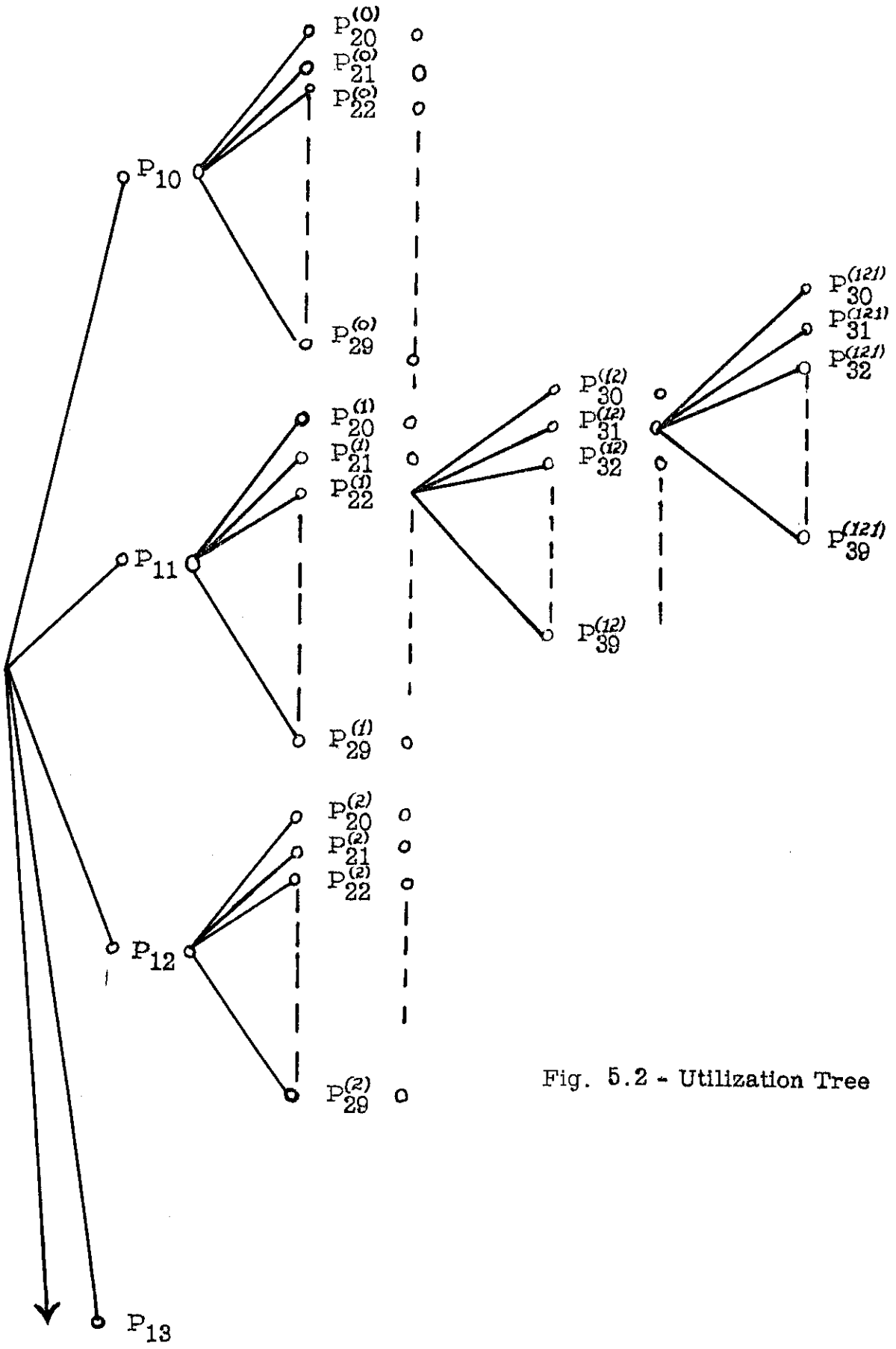


Fig. 5.2 - Utilization Tree

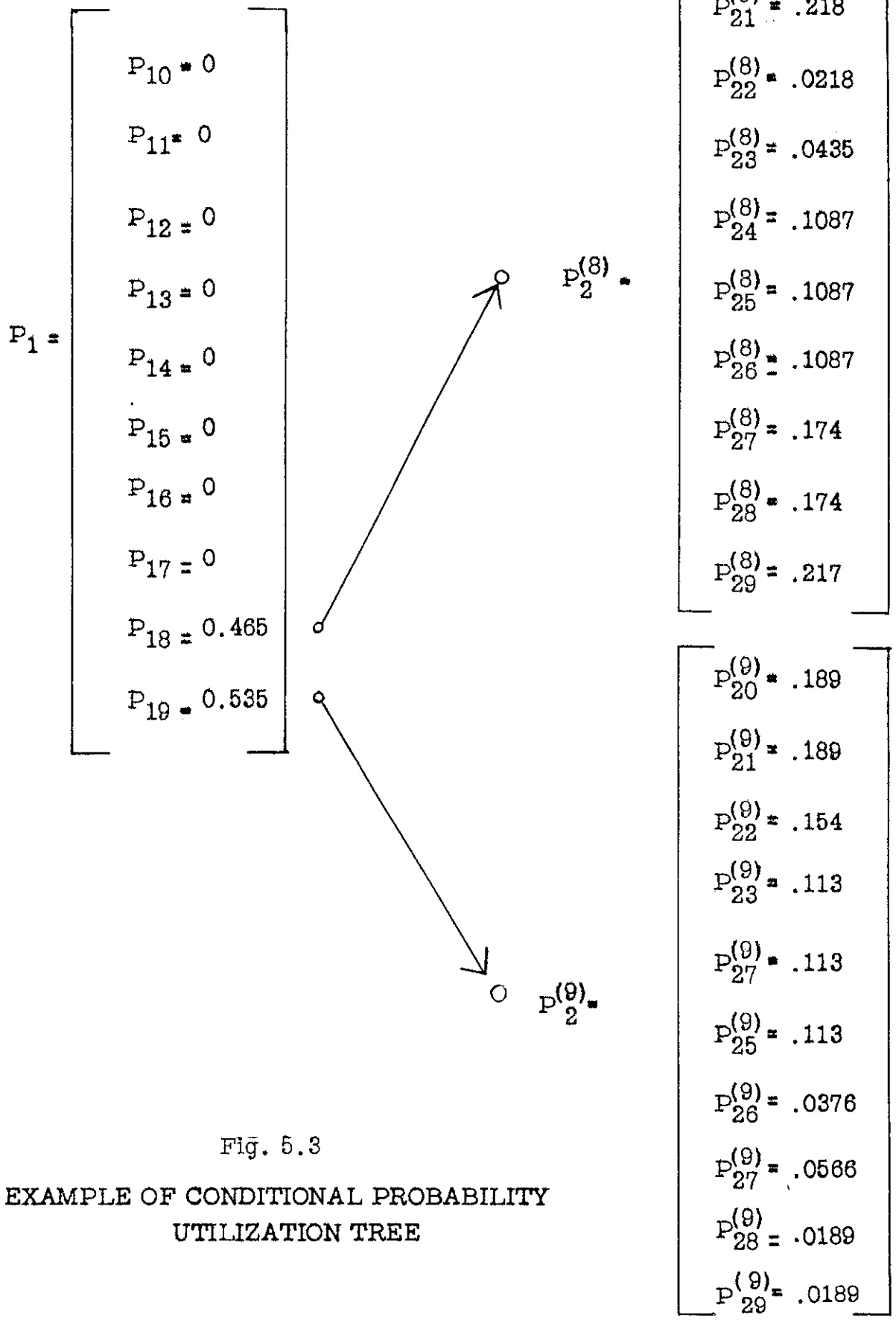
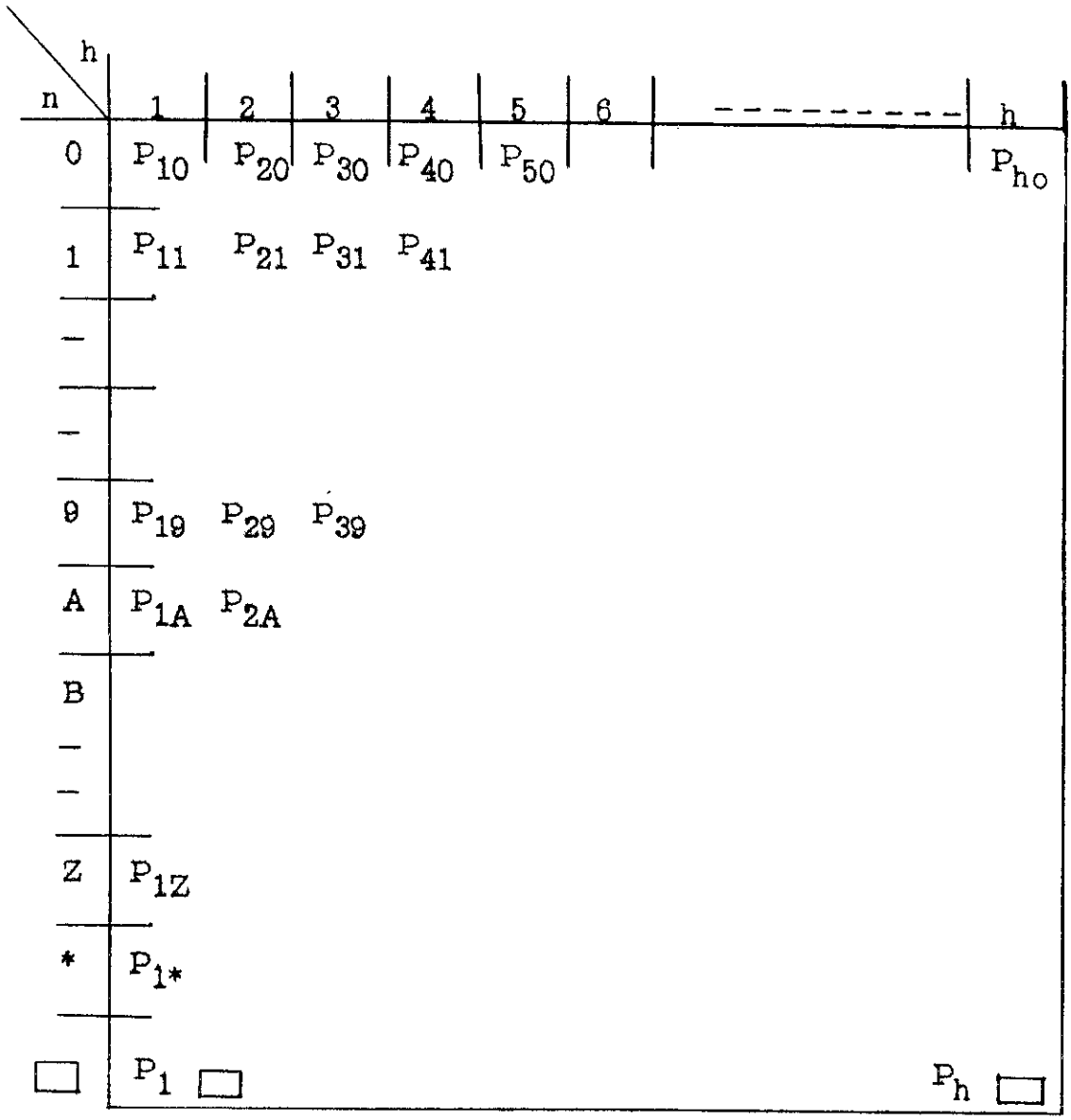


Fig. 5.3

EXAMPLE OF CONDITIONAL PROBABILITY UTILIZATION TREE



n - symbols
h - digits in identification number

Fig. 5.4 — PROBABILITY MATRIX

N. Abramson - DECISION THEORY AND ITS APPLICATION TO COMMUNICATION, IBM RJ-MR-8, Jan. 1958

D. Blackwell and M. A. Girschick - THEORY OF GAMES AND STATISTICAL DECISIONS, NY, Wiley (1954)

W. G. Cochran - SAMPLING TECHNIQUES, NY, Wiley (1953)

W. G. Cochran and G. M. Cox - EXPERIMENTAL DESIGN, NY, Wiley (1957)

W. E. Deming - SOME THEORY OF SAMPLING, NY, Wiley (1950)

M. G. Kendall - ADVANCED THEORY OF STATISTICS, NY, Hafner, Vol. 1, 5th ed. (1952), Vol. 2, 3rd ed. (1951)

M. B. Mann - ANALYSIS AND DESIGN OF EXPERIMENTS, NY, Dover

Thrall, Coombs, and Davis - DECISION PROCESSES, NY, Wiley (1954)

A. Wald - SEQUENTIAL ANALYSIS, NY, Wiley (1947)

5.2 Formal Logic

A set of classes could be defined such as numerical or alphanumeric, odd or even, less than or greater than various reference numbers in the identification system. The development of a proper syntax language for the machine logic to decipher and to then assign addresses would involve a considerable expansion of the command vocabulary. To develop such a calculus of propositions would require a most vigorous effort and a protracted period of contemplation with little direct hope of success. As a means of stating with some final precision the ultimate scope of all addressing problems, this method might be useful.

A basic set of references in formal logic is listed below:

D. Hilbert and W. Ackerman - PRINCIPLES OF MATHEMATICAL LOGIC, Chelsea

Susanne K. Langer - INTRODUCTION TO SYMBOLIC LOGIC, 2nd rev. ed. NY, Dover

P. C. Rosenbloom - ELEMENTS OF MATHEMATICAL LOGIC, NY, Dover

A field related to formal logic is the specialty of cryptography. Two references to cryptography are listed below, since they may yield some useful clues as to potential addressing schemes:

Helen F. Gains - CRYPTANALYSIS, A STUDY OF CIPHERS AND THEIR SOLUTIONS, NY, Dover, (1955)

L. D. Smith, CRYPTOGRAPHY: THE SCIENCE OF SECRET WRITING, NY, Dover, (1955)

Another special field related to formal logic is the development of error-correcting codes. A simple error-correcting code allowing two symbols: 101 and 010 are illustrated in Fig. 5.5. Any single error can be corrected. A minimum distance between points of three sides of the cube in Fig. 5.5 is required to permit single error correction. This suggests the possibility of transforming an identification system in which each identification number is transformed to an intermediate point which is within one side of only one point on an n-dimensional cube. Then error-correcting logic could be applied, followed by dropping the redundant bits to transform an irregular distribution of points to an evenly distributed sequence. This system would not be practical unless a system is developed to transform the identification numbers to the intermediate points. The use of the error-correcting code technique relaxes the requirements on the intermediate transformation.

Two references are listed below:

R. W. Hamming - ERROR-DETECTING AND ERROR-CORRECTING CODES Bell System Tech. Journal, Vol. 29, (1950), p 147.

Colin Cherry - ON HUMAN COMMUNICATION, NY, Wiley (1957) p 185-187

An analysis in terms of n-dimensional space with amplitudes 1 and 0 allowed may involve the use of algebraic number fields.

5.3 Number Theory

The study of integers and the relations between integers is known as number theory. The customer's identification system may be an irregular distribution of integers. If the identification numbers $I = km + a$ is an uneven distribution, it may be possible to find integers k and m which will transform the identifications I to a set of residues a which are more evenly distributed. Considerable research in number theory would require the exploration of possibilities of this approach. If congruences of a discrete nature could be defined or some unique method of treating the residue classes could be developed, a suitable method of addressing would be a natural consequence. A basic reference list for number theory is as follows:

Garret Birkhoff and Saunders MacLane - A SURVEY OF MODERN ALGEBRA, NY, McMillan (1947)

Leonard E. Dickson, INTRODUCTION TO THE THEORY OF NUMBERS, Univ. of Chicago, (1929), reprint NY Dover (1957)

John G. Kemeny, J. Laurie Snell, and Gerald L. Thompson, INTRODUCTION TO FINITE MATHEMATICS, Englewood Cliffs, NY, Prentice-Hall, (1957)

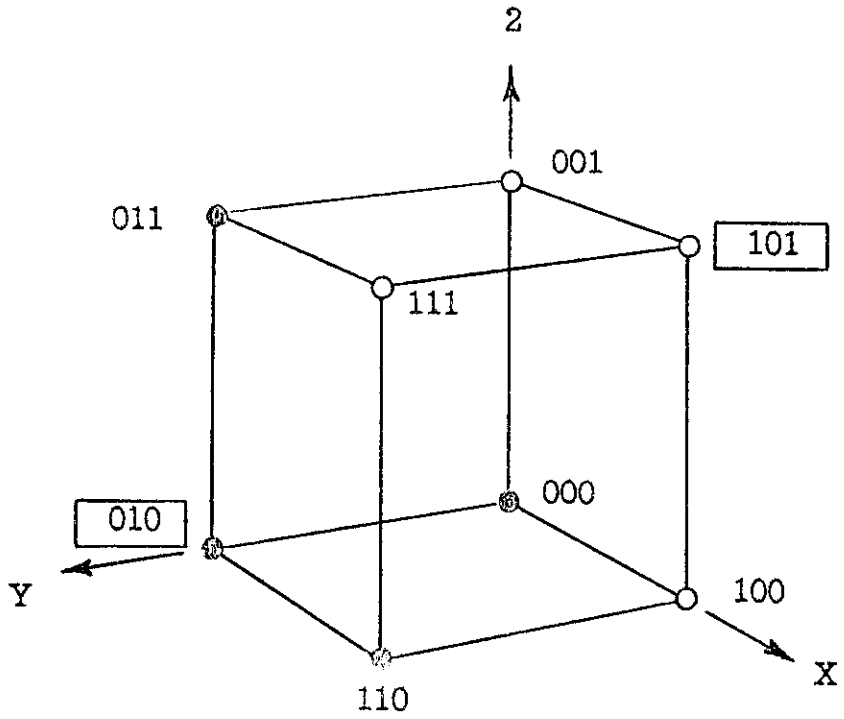


Fig. 5.5 - SIMPLE ERROR-CORRECTING CODE SPACE

I. M. Vinogradov, INTRODUCTION TO THE THEORY OF NUMBERS, London, Pergamon (1955)

I. M. Vinogradov, ELEMENTS OF NUMBER THEORY, 5th ed., NY, Dover

5.4 Geometry

Consider the set of all possible identification numbers as a set of evenly distributed points on the surface of a cylinder generated by a straight line. Consider a second cylinder generated by a curved line representing the density of usage of the potential numbers. It may be possible to develop a geometrical or topological transformation which would transform the usage cylinderical surface to a third geometrical surface which has a direct relationship to the memory. Two references on topology are listed below:

E. M. Patterson, TOPOLOGY, NY Interscience (1956)

Harold E. Tinnappel, ON THE TOPOLOGICAL INDEX, Pub. 24,516, Ph.D. Thesis, Ohio State, 1952, Dissertation Abstracts (1958) Vol. 18, No. 3, p 1057-1061

APPENDIX 6

TENTATIVE FUNCTIONAL DESCRIPTION OF THE ADVANCED FILE AND VLCM

These descriptions are included here primarily for reference purposes for the proposed table look-up techniques are predicated on their improved capabilities.

6.1 Functional Description of the Advanced File

The Advanced File is a random access magnetic disk storage device with capacity ranging from 25 to 75 million alphanumeric characters. The capacity range is obtained through providing from one to three modular disk arrays within one cabinet. Each of the modular disk arrays will have the following characteristics:

A. 20 disks with 40 surfaces, each with 250 tracks.

B. One to four comb type access mechanisms. Each comb will have 40 read/write heads (one for each surface) capable of moving across the 250 tracks. Average access time will be about 80 ms.

C. Track oriented addressing. The addresses will start at track zero, surface zero, and continue down through surface 39 then move to track one, on down, etc., to the highest module address at track 249, surface 39. Addresses will then continue to the next module in the same fashion.

D. Flexible length record facilities. The number of records per track and therefore the number of characters for each of the records, all the same length, can be controlled through a program set-up operation. All records in a module with the same track number (e.g., track 00, surfaces 00 through 39) will be of the same length.

   Due to the variation in the number of records possible per track, the address of records within the file may not be continuous but contains gaps between the address of the last record on a surface and the first record of the next surface. An indelible address will be available before each record in the file to insure operation upon the proper record.

   Total capacity of the module is dependent upon the number of records per track and the required gaps between these records. The maximum record length, with one record per track is 2,500 characters. Therefore, the maximum capacity of a module is 25 million characters.

E. Data within the file will be transferred serial-by-bit, serial-by-character at a rate of 600,000 bits per second.

F. A disk speed of 1,800 RPM provides a rotation time of 33 ms.

6.2 Functional Description of the VLCM

The Very Large Capacity Memory (VLCM) is a rectangular bin array with a capacity of one billion alphanumeric characters.

The basic physical element is the short magnetic strip of which there are 10,000. The strips are grouped 10 to a subcell, with 10 subcells to a cell, 10 cells in a bin and 10 bins per VLCM.

Each strip contains 10 character channels with 10,000 characters per channel. For read-write, the strip is completely extracted from its home cell and wrapped around a hollow drum so that the character channels can be scanned sequentially. Scan time for one character channel will be 50 ms. The data is transferred parallel-by-bit, serial-by-character at 200,000 characters per second.

The bins move in a direction at right angles to that of the access mechanisms. The latter carry the heads and the hollow drum. During seek, the bins and access mechanism move simultaneously.

Both bins and cells will be interchangeable and removable for reliability and expandability. The effect is an open-ended file.

There will be two independent access mechanisms in the basic machine. Since access time will be of the order of 300 ms. and the scan time 500 ms., (for all 10 channels in a strip or "bucket"), working the access mechanisms in alternation would allow access time to be completely overlapped. To realize this, it is planned to reserve an area near the beginning of Channel 0 of each bucket to house an overflow address. On scanning the current strip, the overflow address is automatically transmitted to the address register of the alternate access mechanism, setting the latter into motion. In this (automatic chaining)manner, physical buckets may be linked together to form logical buckets of varying lengths.

The VLCM may operate in one of several modes. In the bucket search mode, a 4-digit bucket address is given. This permits an automatic scan of all channels of the "home" bucket, plus any overflow buckets to which it is linked. In the channel search mode, a 5-digit address will give scanning over a particular channel. In the sector retrieval mode, an additional 2 digits in the address will pinpoint the desired sector.

The record length will be adjustable-fixed. Tentatively, calibration will permit the choice of 100, 200 or 500-character records for a given machine.

Adjustable-length field compare will be available for retrieval on the basis of any identifiers in the item.

APPENDIX 7

IDENTIFICATION CHARACTERISTICS

Many of the problems associated with addressing a large file are brought about by the restrictions placed on a system by the item identifiers. As was previously mentioned, the identification codes are established from system considerations and, therefore, in a general addressing method cannot be altered to facilitate addressing the items.

Some general examples of identification origination are listed below to illustrate some of the problems associated with identification codes.

A. The identifiers may originally form a complete sequential set. But with the passing of time, some members may be removed from the set and new ones added. Many accounting files fall into this category. For example, in the insurance industry, policy number might be the identifier for the items in a file, but policy cancellations would cause some numbers to drop out of the set.

B. The entire set of identifiers may be a complete sequential set, but a given customer might deal with only a subset of the entire set. An identification of employees by social security number would be an example of this.

C. Identifiers are often built up from several independent numbers. For example, a part number could uniquely identify a part, and additionally, the number could represent department, material type, location classification, and serial number within these classes.

D. The identification numbers may start out as one of the first three classes, but is later broken into smaller categories by a prefix or suffix to the original identifier.

E. An identifier may be a name, descriptive word or phrase. In this case, the identifier will contain alphanumeric characters and may be of variable length.

F. There may be multiple identifiers in an item. The one normally used for retrieval is called the "primary identifier"; the others are classified as secondary identifiers. The primary identifier will normally be a unique description of an item such as payroll number. The secondary identifiers may or may not be unique. Social security number or name might be unique secondary identifiers, while department or job classification would not be unique.

APPENDIX 8

BIBLIOGRAPHY

8.1 Introduction

This bibliography is divided into three parts: General, History of Indexing, and Physical Techniques. References on basic mathematics are indicated in the text of Appendix 5.

The specific indexing needs of the application to business and government of the next few stages in the development of larger memory files is not thoroughly documented. The available literature is weighted in the history of what has been accomplished and in the vision of the utopias of the future such as storing all human knowledge. Within each section the references are arranged chronologically by year and alphabetically by author within the year.

This list was started by an information retrieval search run by Mr. F. E. Firth, on a file of 5,000 references stored in a 305A RAMAC, using the following key terms, the numbers in parentheses refer to the number of references found in the search:

A. Addressing (7)

B. Addressing Automatic/Direct (1)

C. Record Retrieval (None)

D. Retrieval (13)

E. Scanning (15)

F. Search (35)

G. Table Look-Up (2)

H. Randomizing (27)

I. Indexing (14)

The items retrieved above have been supplemented by references from the personal files of members of the Committee.

The procedure used is described in the following reference:

F. E. Firth - AN EXPERIMENT IN MECHANICAL SEARCHING OF RESEARCH LITERATURE WITH "RAMAC", May, 1958 (Western Joint Computer Conference)

8.2 General

These references are directed to the literature retrieval problem, and are included because they indicate future trends in larger memory requirements or because of specific details applicable to the more immediate indexing problem.

1953

M. Taube - STUDIES IN COORDINATE INDEXING, Vol. 1, Documentation Incorporated, Washington, DC (1953), p 73, "The Logical Structure of Coordinate Indexing" (Formal Logic)

1954

M. Taube, OPT. CIT., Vol. 2, p 15, "Searching and Collating," p 72, "The Problem of Machine Association of Ideas", p 81, "The Actual and Potential Association of Ideas in Information System," p 85, "An extension of the Algebra of Classes for the Association of Ideas."

1955

V. P. Cherenin, CERTAIN PROBLEMS OF DOCUMENTATION AND MECHANIZATION OF INFORMATION SEARCH, Moscow, 1955, Lib. of Congress Trans. RT-4586 (San Jose Lib.-Pam 01676) The section on coding has application to catalog numbering systems having a logical structure.

1956

M. Taube, OP. CIT., Vol. 3, p 7-17, "Communication Theory and Storage and Retrieval Systems"

1957

H. P. Luhn, STATISTICAL APPROACH TO MECHANIZED LITERATURE SEARCH, Jan. 1957, SJ-Pam 01453

H. P. Luhn, MECHANIZED ENCODING AND SEARCHING OF LITERARY INFORMATION, Oct. 1957, p309, IBM Journal

1957 (Cont'd.)

J. H. Shera, A. Kent, and J. W. Perry, INFORMATION SYSTEMS IN DOCUMENTATION, NY Interscience (1957), p 535 - Of particular interest in the future machine envisioned by L. I. Gutenmahker which would contain the entire contents of all scientific works.

1958

Anon. INFORMATION HEADACHES, Chem. & Engin. News, Feb. 1958, p 104, proposals for national information center.

J. J. Nolan, INFORMATION STORAGE AND RETRIEVAL USING A LARGE SCALE RANDOM ACCESS MEMORY, SJ-1, Jan. 1958

8.3 History of Memory Indexing

1951

W. Buchholz, SINGLE COUNTER DRUM ADDRESSING, Mar. 1951, 003.040.223

E. F. Codd and P. W. Knapland, TABLE SEARCHING ON SINGLE DRUM WITH PILOT ARGUMENTS, Apr. 1951, 001.010.260

N. Rochester and J. H. Holland, TABLE LOOK-UP PROCEDURES FOR FREQUENTLY USED TABLES, Mar. 1951, 011.010.225

1952

M. E. Maron, INFORMATION SEARCHING SYSTEM, Sept. 1952, 201.000.012, Research Library Searching Using Random Numbers.

1953

J. W. Haanstra, D. W. Kean, and M. E. Maron, AUTOMATIC ADDRESSING FOR RANDOM ACCESS STORAGE UNITS, July, 1953, 201.004.031

H. P. Luhn, ADDRESSING SYSTEM FOR RANDOM ACCESS STORAGE, Mar. 1953, 001.011.450

1955

W. W. Peterson and L. H. Haibt, PROCEEDINGS OF THE CONFERENCE ON INDEX NUMBERS, June, 1955, 001.011.575

1956

E. J. Issac and R. C. Singleton, SORTING BY ADDRESS CALCULATION, Apr. 1956, Journal Assoc, for Computing Machines, vol. 3, No. 3, July, 1956, p 169-174, SJ-Pam 01231

F. B. Wood, ELECTRONIC AUTOMATIC ADDRESSING, IBM Code: 211.011.101, Apr, 1956

A. I. Dumey, INDEXING FOR RAPID RANDOM ACCESS MEMORY, Computers and Automation - Vol. 5, No. 12, Dec. 1956, p 6 - 9

1957

W. P. Heising, SOME MATHEMATICAL NOTES ON RANDOM ADDRESSING, Nov. 1957, (Unpublished Notes)

IBM 650 Bulletins, BULLETIN 11, 650 RAMAC ADDRESSING, Oct. 1957, #32-7935

W. W. Peterson, ADDRESS FOR RANDOM ACCESS STORAGE, IBM Journal Apr. 1957, p 130

The Service Bureau Corp. PROGRAMMED ADDRESS CONVERSION OF THE IBM 305 RAMAC, Bull. 28-9022

1958

J. E. Applequist, PROPOSAL FOR AN AUTOMATIC TRANSLATING DEVICE INCORPORATING THE ADVANCE FILE, SJ Prod. Dev. Tech. Memo, Apr. 1958

IBM Data Processor, Vol. 3, No. 2, Mar. 1958, p 4-5, CHAINING A USEFUL METHOD IN RAMAC FILE ORGANIZATION, Vol. 3 No. 3, Apr. 1958, p 8. CHAINING (Published two revised tables for the above article.)

IBM 650 Applied Programming Group, New York (Mimeographed paper, 21 pages) FILE ORGANIZATION FOR IBM DISK FILE UNITS
A METHOD OF FILE ORGANIZATION FOR THE 650 RAMAC
A PROGRAM FOR TESTING CONVERSION FORMULAE - VALADCON (Evaluation of Address Conversion)

W. H. Marlow, DATA PROCESSING FOR LOGISTICS, Feb. 1958, (Paper at Dec. 1957 DP Symp. Spons. by Navy Math. Computers Advisory Panel) p 9 - Scalar Product Addressing; p 9 - Direct Addressing Using Digits of Input Data to Build Address; p 10 - Table Look-up by Classes; p 14 - Auxiliary Drum for Addressing.

H. W. Renner, A LINK ADDRESS TECHNIQUE FOR PROVIDING MULTIPLE SEQUENTIAL PATHS THROUGH RECORDS COMPRISING A RANDOMLY ACCESSIBLE STORE, DP Prod. Planning, Endicott, Feb. 1958

D. Teichroew, Unpublished Memorandum to E. A. Quade (San Jose Research Laboratory) Identification and Requirements for a Future Memory System (May, 1958)

8.4 Physical Techniques

The following techniques offer possibilities for new physical systems which would solve the indexing problem:

A. Electronic Switching (Cores)

A potentially low cost core system has been reported in which holes are formed in sheets of ferrite with a part of the wiring prepared by printed circuit techniques.

(J. A. Rajchman, FERRITE APERTURED PLATE FOR RANDOM ACCESS MEMORY, Proc. IRE, Mar. 1957, p 325-334, SJ-Pam 01903)

B. Delay Line Logic

Delay line logic offers an opportunity to provide addressing operations at a higher speed than the basic machine logic. For information on delay lines refer to:

F. C. Chiang, Research Laboratory, San Jose, California

C. High Speed Auxiliary Drum

A potential table look-up system is a high-speed auxiliary drum, such as a small air bearing type drum.

R. B. Edwards and J. J. Lynott, "Random Access Magnetic Air Drum," RJ-RR-125, May, 1958

D. Random Net Logic (learning)

The ultimate in addressing techniques would be to have a unit which could perform associative addressing similar to the functioning of the human brain. Such a type of logic has been explored theoretically at Cornell Aeronautical Laboratories. A series of logic elements would be connected randomly in parallel to a group of elements in the next stage. Inhibiting connections would be included so that an identification pattern could be learned. This would be dependent upon the development of a logic element whose threshold changes with usage. The mathematical analysis of simulation on a computer has been accomplished at Cornell Aeronautical Laboratories.

Frank Rosenblatt, THE PERCEPTRON, A THEORY OF STATISTICAL SEPARABILITY ON COGNITIVE SYSTEMS, Buffalo, Cornell Aeronautical Labs., Jan. 1958, Report VG-1196-G-1

J. S. Bruner, Jacqueline J. Goodnow, and George A. Austin, A STUDY OF THINKING, a publication of the Harvard Cognition Project, NY, Wiley (1956)