# The IBM 9900 Special Index Analyzer

by R. W. Murphy

International Business Machines Corporation
Poughkeepsie, New York

November 17, 1958

# THE IBM 9900 SPECIAL INDEX ANALYZER

by

R. W. Murphy
International Business Machines Corporation
Poughkeepsie, New York

The IBM 9900 Special Index Analyzer is IBM's version of the concept of Continuous Multiple Access Collating developed by Documentation, Incorporated, Washington, D. C., under a research contract sponsored by the Air Force Office of Scientific Research, (ARDC) Directorate of Research Communications.

November 17, 1958

# THE IBM 9900 SPECIAL INDEX ANALYZER

The IBM Special Index Analyzer is a machine designed to facilitate reference to catalogued information. It may be applied to such activities as library research, or the searching of equipment design specifications. These activities share the essential problem that, in order to make use of information which has been stored, most of it must be prevented from having to be considered by the user. If the user can specify attributes of the information in which he is interested, the Special Index Analyzer will select out of the files only those references to items of information which possess that particular association of attributes. For example, a library searching problem might be to determine all the material dealing, in the same article, with reliability, transistors, and digital computers.

An information retrieval system employing the Special Index Analyzer is set up on the basis that a document, or other item of information, can be categorized by a set of terms. The terms may be the names of topics, subjects, or attributes or they may be key-words actually used in documents of the collection. When items or documents are entered into the system, each is analyzed to find which terms are pertinent to it and records are made associating the item with significant items. The terms are drawn from a pre-established glossary which is used uniformly throughout the analysis of items in the collection. These records are then rearranged into a "where found" index; that is, an index consisting of subdivisions called term files, each of which includes all the references to which an individual term is pertinent. In this arrangement the term files are ready for use by the researcher, employing the IBM 9900 Special Index Analyzer to select out significant references automatically and accurately.

The operations performed by the Special Index Analyzer are of the type found in the theory of sets. These are primarily the intersection operation in which the result is the set of item references containing only those references common to the two input sets being operated upon, and the union operation which produces a resultant set containing references from either of the two operand sets. In addition, the Special Index Analyzer can perform the intersection with the various complements of the operand sets.

In its application to information retrieval the Special Index Analyzer employs, as operand sets, term files selected from the index files by the researcher. In effect, the intersection operations provide the researcher with the means of narrowing down of the scope of his search in accordance with the degree of specificity with which he can select term files. The union operation allows him to apply the narrowing down to as many terms as he feels are significant in his search.

# METHODS OF INFORMATION RETRIEVAL AND THE APPLICATION OF THE SPECIAL INDEX ANALYZER

All information retrieval systems share the requirement that the content of an item or document to be included in the system must be determined by a trained analyzer and recorded. Once this has been accomplished, however, the various retrieval systems differ in the manner in which the records of information content are maintained and used. In order to characterize the methods of filing information relating terms and items, a matrix representation is most useful. In this, the set of terms constituting the glossary is arranged as one axis of the matrix, while the items of the collection are represented along the second axis. For each determination that a term relates to an item, the appropriate position of the matrix is posted with the fact of relationship. This posting of a relationship will be referred to as an "entry" and may be written either as a binary mark in the matrix or as the juxtaposition of a term code with an item code (the unit record form).

## THE MATRIX REPRESENTATION OF RETRIEVAL FILES

— an ITEM is a physical object . . . book, document, map, record, patent . . . which is the object of the search

— a TERM is the name of a topic or attribute of the item

|  |  | TERMS IN THE GLOSSARY |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
|  |  | 004 | 005 | 007 | 009 | 015 | 020 |
| ITEMS IN THE LIBRARY | 016 | X |  | X |  |  |  |
|  | 041 |  | X | X |  |  | X |
|  | 042 |  |  |  | X |  |  |
|  | 050 |  |  | X |  | X |  |
|  | 089 |  | X |  |  |  | X |
|  | 093 |  |  |  | X | X |  |
|  | 101 | X |  | X |  |  |  |
|  | 103 | X |  |  |  | X |  |

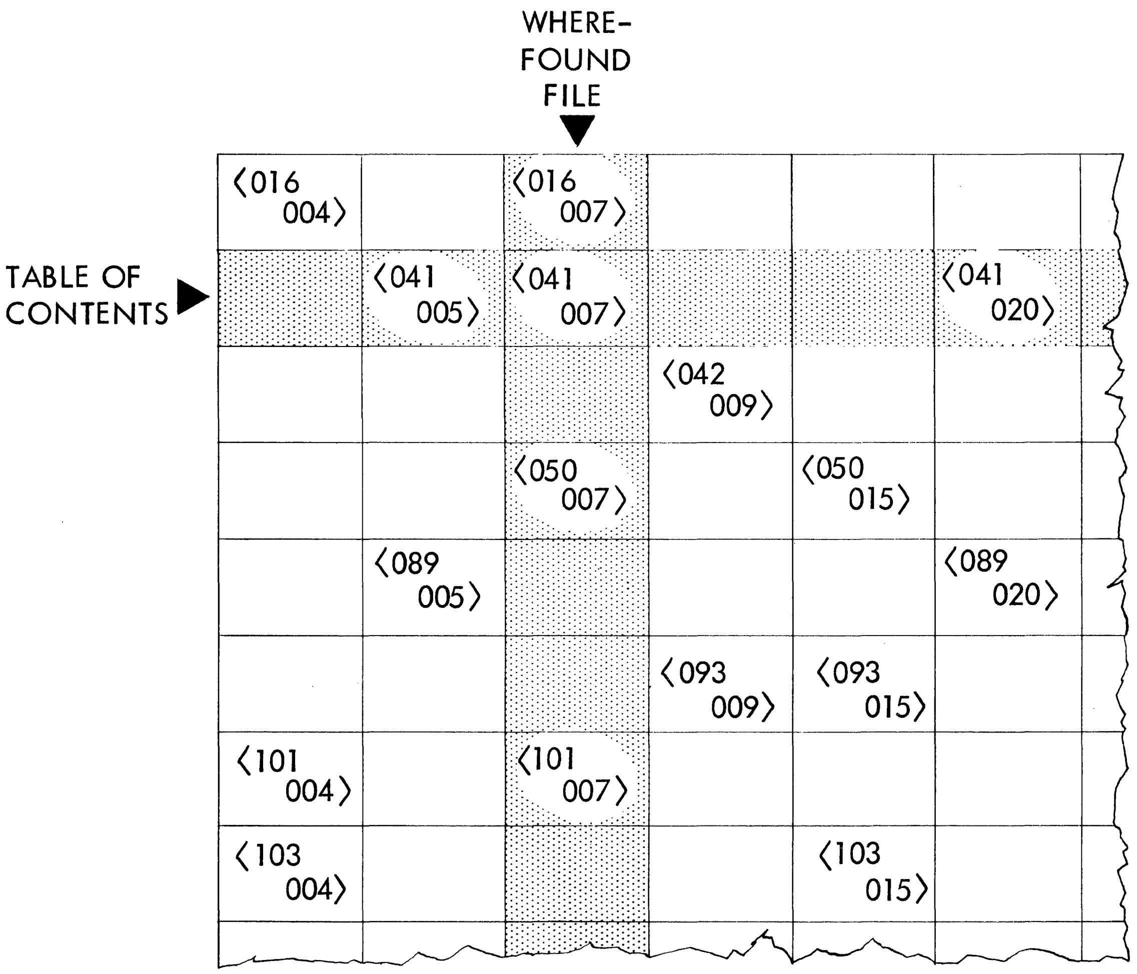
In this example, as in many practical cases, numerical codes designate both the terms of the glossary and the items of the collection. Codes which stand for terms and codes which stand for items will therefore look alike, and must be distinguished either by context or by adding a supplementary symbol.

The matrix representation is intended as a conceptual device. Most practical media for the storage of data require that the information contained in the matrix be converted to a linear sequence of unit records in the process of filing. The two usual ways of linearizing the matrix are by taking successive rows, or else by taking successive columns. Each row of the matrix corresponds to a table of contents of one of the items in the library, while each column corresponds to a 'where-found' listing.

## LINEARIZING THE MATRIX FOR DATA STORAGE

All the unit records made of entries to the matrix constitute a grand set representative of the entire library. The process of linearizing the matrix arrangement of the grand set results in the distinguishing of subsets which may be of either one type or another, depending on how the matrix is linearized. If the subset is taken from a row of the matrix, it contains unit records of entries all referring to the same item and designating all the terms to which the item pertains. This kind of subset is therefore a table of contents or item file, and contains term codes as the essential elements of information. The complete collection of item files is the grand set, but it is usable as the catalog of the library.

The alternate way of separating out subsets of the grand set is by taking them from columns of the matrix. Within a subset, all of the unit records will contain the same term code, but differ in the item codes. This kind of subset, or term file, tells of all the items where a particular topic, or term, is treated. The complete collection of term files constitutes an index to the library.



LINEARIZING BY ROW

$\{\langle 016,004\rangle \langle 016,007\rangle \ldots\}, \{\langle 041,005\rangle \langle 041,007\rangle \langle 041,020\rangle \ldots\}\ldots$

CATALOG = SET OF ALL TABLES OF CONTENTS

$= \{I_j\}$

AN ITEM FILE, $I_j = \{\langle i_j, t_k\rangle ;\ j \text{ is fixed}\}$

LINEARIZING BY COLUMN

$\{\langle 016,004\rangle, \langle 101,004\rangle, \langle 103,004\rangle \ldots\}, \{\langle 041,005\rangle, \langle 089,005\rangle \ldots\}\ldots$

INDEX = SET OF ALL WHERE-FOUND FILES

$= \{T_k\}$

A TERM FILE, $T_k = \{\langle i_j, t_k\rangle ;\ k \text{ is fixed}\}$

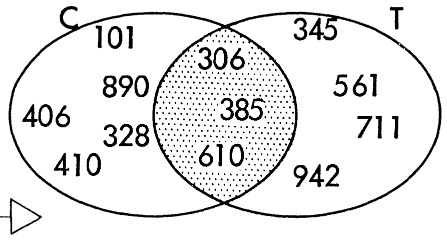
For information retrieval purposes, the IBM Special Index Analyzer is intended to be used with the index containing the "where-found", or term, files. Its essential function is to combine a pair of term files to produce a new term file, usually containing many fewer term codes than appeared in the original files. By means of his selection of the input term files and through his choice of the combining operations, the researcher can program the Special Index Analyzer to reduce a number of voluminous term files to just one set of item references, which meet his specifications, and at the same time, are few enough in number to allow the researcher to refer to the items directly.

The combining operations performed by the IBM Special Index Analyzer are operations on sets of item codes. There are only two basic operations which can be performed on two operand (input) sets, the intersection ($T \cap C$) which finds the elements common to T and C, and the union ($T \cup C$) which places the elements occurring in either (or both) of T and C in a resultant set containing no duplications. In addition, set theory deals with the complement of a set ($\bar{T}$), that is, with the set containing all of the elements which are not in T. However, since any machine can only develop new sets from sets which are specifically introduced, the complement is used in conjunction with intersection in the IBM Special Index Analyzer to provide three additional operations which complete the range of operations performable on two operand sets.

## OPERATIONS ON SETS IN THE SPECIAL INDEX ANALYZER

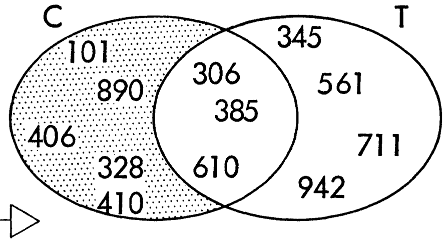
A. INTERSECTION

$T \cap C$ THE ITEM CODES COMMON TO T AND C



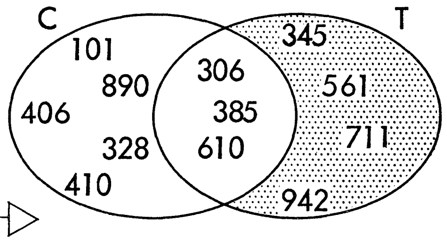
B. INTERSECTION WITH COMPLEMENT

$\bar{T} \cap C$ THE ITEM CODES IN C, BUT NOT IN T



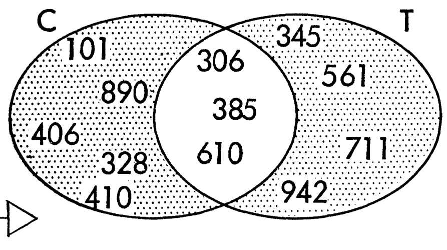
C. INTERSECTION WITH COMPLEMENT

$T \cap \bar{C}$ THE ITEM CODES IN T, BUT NOT IN C



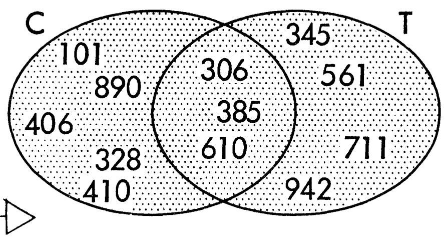
D. UNION OF COMPLEMENT INTERSECTIONS

$(\bar{T} \cap C) \cup (T \cap \bar{C})$ THE ITEM CODES IN T OR C, BUT NOT IN BOTH



E. UNION

$T \cup C$ THE ITEM CODES IN EITHER T OR C



IN THE IBM SPECIAL INDEX ANALYZER

T IS THE TERM FILE ON TAPE

C IS THE FILE BROUGHT IN FROM CARDS

Most library search operations will involve more than two term files, to be combined by means of various set-theoretic operations arranged into a program by the researcher. The program together with the term files serving as operands in the program, is equivalent to a set-theoretic expression of several variables, and may be rearranged, as the set-theoretic expression is re-arranged, in order to obtain a simpler or more efficient program. Set theory provides the relations by which the expression can be reduced to the form which provides the most efficient program.

$$T_1 \cap T_1 = T_1 \qquad T_1 \cap T_2 = T_2 \cap T_1$$

$$T_1 \cup T_1 = T_1 \qquad T_1 \cup T_2 = T_2 \cup T_1$$

$$T_1 \cap (T_2 \cap T_3) = (T_1 \cap T_2) \cap T_3$$

$$T_1 \cup (T_2 \cup T_3) = (T_1 \cup T_2) \cup T_3$$

The result of taking the intersection of any number of sets depends only on what sets are involved and not on the order in which sets are combined, nor on the number of times a set is repeated. The same is true in taking the union of several sets.

$$(T_1 \cap T_2) \cup (T_1 \cap T_3) = T_1 \cap (T_2 \cup T_3)$$

$$(T_1 \cup T_2) \cap (T_1 \cup T_3) = T_1 \cup (T_2 \cap T_3)$$

$$T_1 \cap (T_1 \cup T_2) = T_1$$

$$T_1 \cup (T_1 \cap T_2) = T_1$$

$$T_1 \cap (\overline{T}_2 \cup \overline{T}_3) = T_1 \cap \overline{(T_2 \cap T_3)}$$

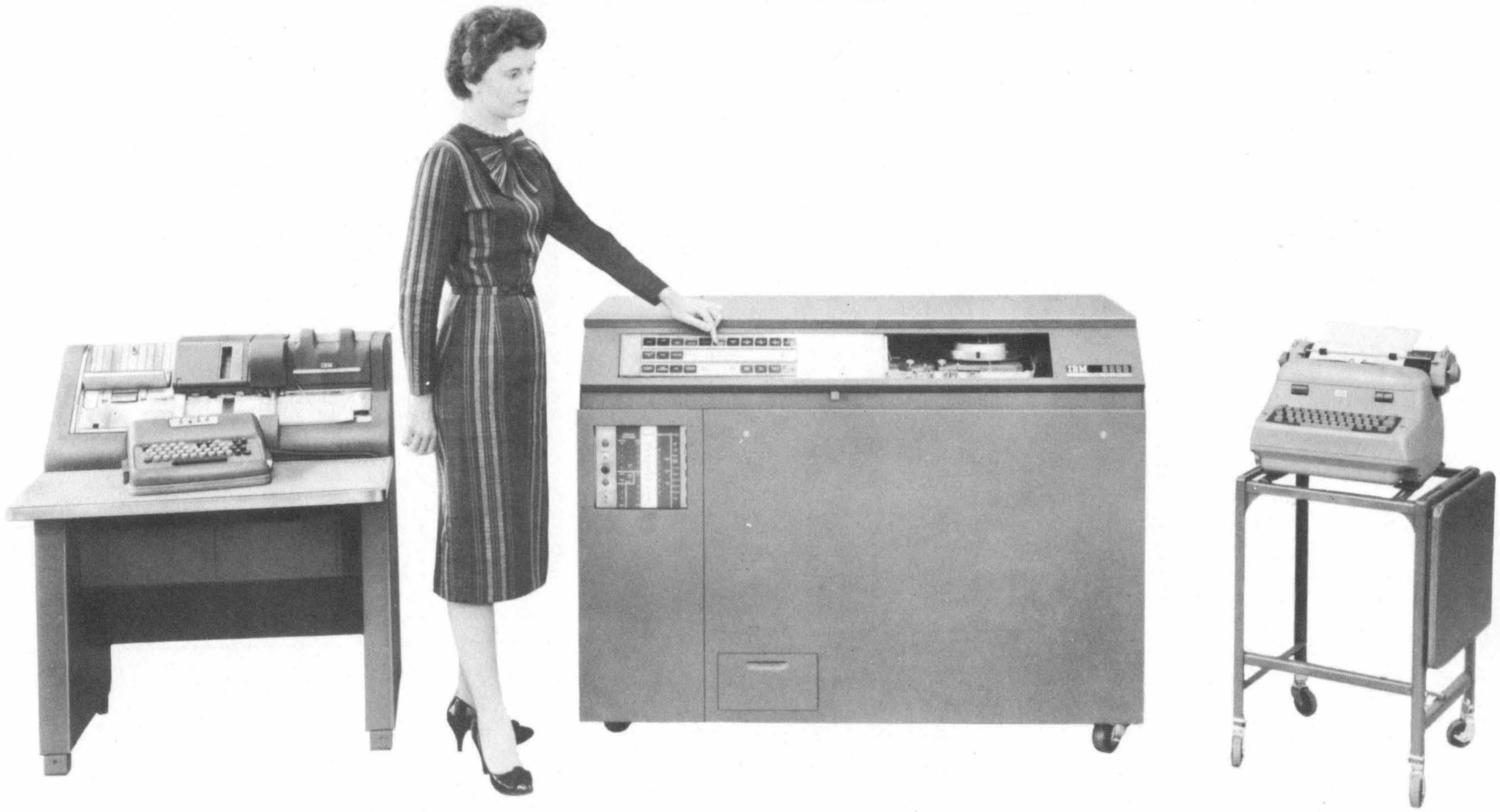
$$T_1 \cap (\overline{T}_2 \cap \overline{T}_3) = T_1 \cap \overline{(T_2 \cup T_3)}$$

The complement of any expression can be obtained by taking the complement of each term (letter or parenthetical term) and interchanging each cup for a cap and vice-versa.

In planning a search, the researcher will usually work out a statement in words of the course which the search is to follow. The verbal statement can then be written as a set-theoretic expression, using such symbols as $T_1$, $T_2$, to stand for the terms stated. Then, if necessary, the set-theorectic relations are used to reduce the complexity of the expression. The final step is to select the required term files out of the index and incorporate them with the program to obtain the machinable equivalent of the original statement.

| | |
|---|---|
| STATEMENT | Retrieve the items dealing with transistors and computers but not with production, as well as the items dealing with transistors, computers, and reliability. |
| EXPRESSION | $(T_1 \cap T_2 \cap \overline{T_3}) \cup (T_1 \cap T_2 \cap T_4)$<br>where: $T_1$ = transistor<br>$T_2$ = computer<br>$T_3$ = production<br>$T_4$ = reliability |
| SIMPLIFICATION | $(T_1 \cap T_2) \cap (\overline{T_3} \cup T_4)$<br>$T_1 \cap T_2 \cap \overline{(T_3 \cap T_4)}$ |
| PROGRAM | 1. Run in $T_4$ ("reliability" term file)<br>2. Intersection type B with $T_3$ ("production" term file)<br>3. Intersection type B with $T_2$ ("computer" term file)<br>4. Intersection type A with $T_1$ ("transistor" term file)<br>5. Print out result |

IBM 9900 SPECIAL INDEX ANALYZER



## FUNCTIONAL CHARACTERISTICS OF THE IBM SPECIAL INDEX ANALYZER

The IBM Special Index Analyzer is composed of three units, the first unit is a modified IBM 26 Card Punch which is used primarily for reading cards when operated with the system. It may also be employed as a standard card punch when the Special Index Analyzer is not in operation. The second unit is the logical and intermediate storage unit and contains both the control equipment and a paper tape punch and reader for retaining the intermediate results of operations. The final unit is a typewriter which is used for automatically printing the results of the search.

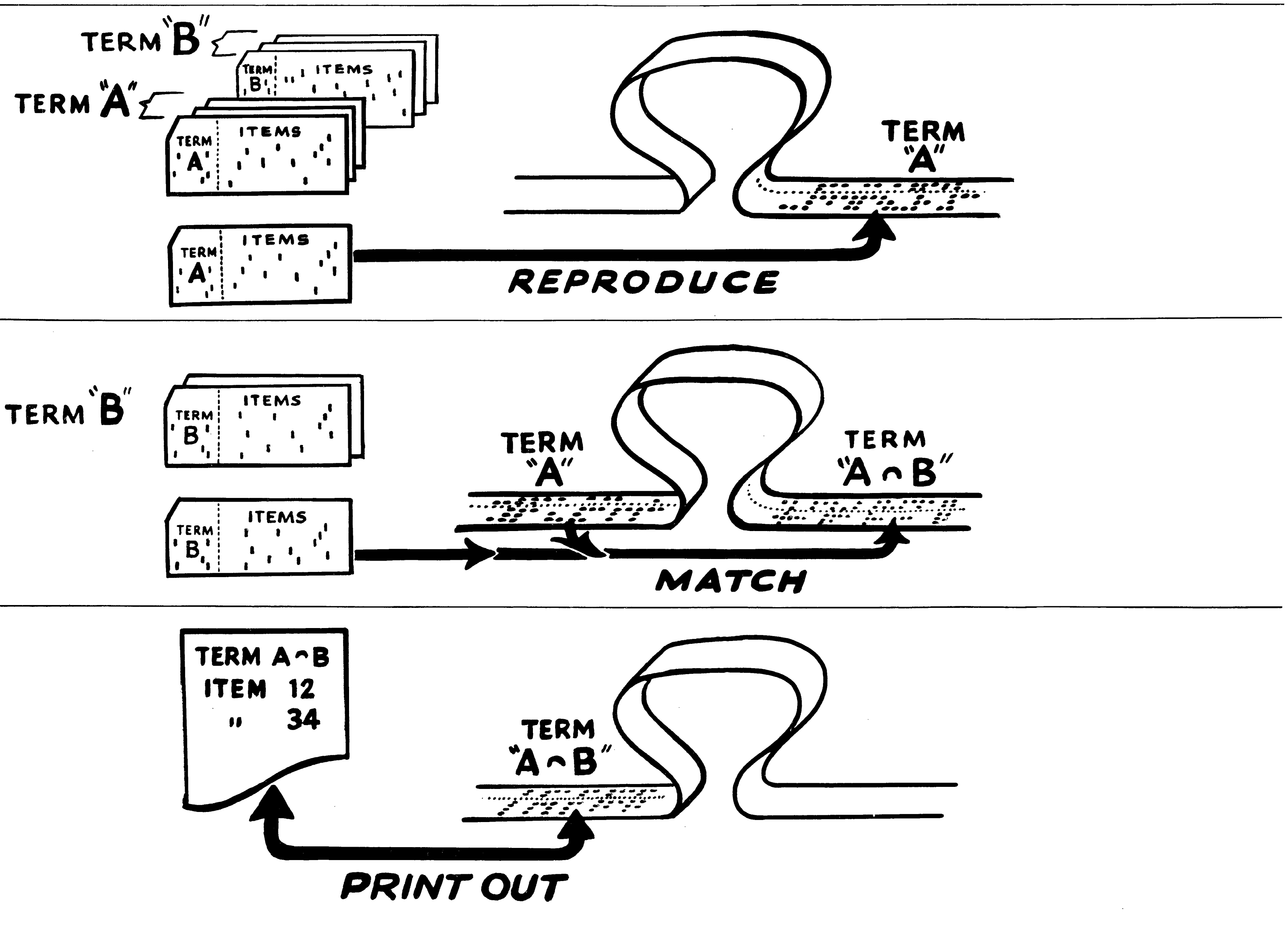
The Special Index Analyzer functions as a collator working with six-digit codes, rather than with complete card records. Codes within a term file are always maintained in numerical sequence allowing the Special Index Analyzer to operate upon term files containing hundreds or thousands of item codes. Item codes are read one by one from either of two inputs, one of which is the card reader and the other, the paper tape reader. After being read, a comparison is made between the two, and depending on how they compare and what operation is being performed, one or neither of the codes may be punched in paper tape and a new code brought in for the next cycle.

The five operations for combining sets all make use of both the card reader and the paper tape reader, and punch the resultant set into paper tape. In addition, there are certain "housekeeping" operations used for initiating a program and for reproducing the final result in a convenient form. In starting a new program, the first term file must be reproduced on paper tape before it can be combined with the second term file. After all of the term files called for by the program have been combined, the result will appear in the paper tape, from which it is usually printed out on a form.

## DATA FLOW IN THE SPECIAL INDEX ANALYZER

The index, or master file for information retrieval, is maintained on standard IBM punched cards. It will be composed of individual decks of cards each deck conveying the item codes corresponding to one term code. In turn, each deck or term file will consist of one or more cards, depending on how many items are associated with that one term. Since it is necessary in setting up a program to select manually the appropriate term files from the index, punched card decks form a convenient and inexpensive storage medium. In addition, as new material is added to the library, the individual term decks can be extended to include the additional retrieval data. For this purpose, the Special Index Analyzer provides an alternative mode of operation which reproduces data from paper tape into new term cards.

# IBM SPECIAL INDEX ANALYZER



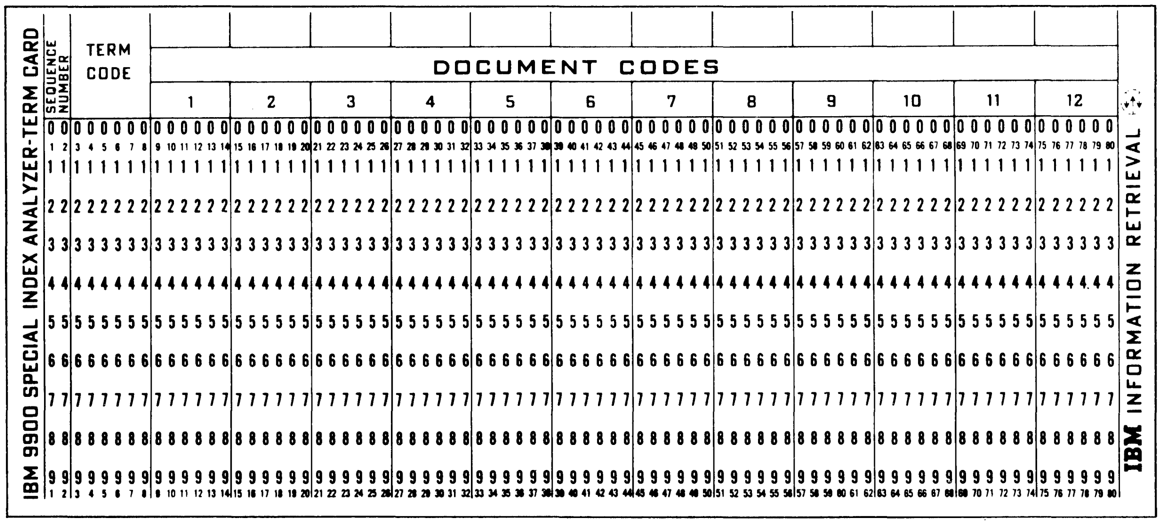
A single term card has space for thirteen six-digit codes, plus two additional digits. Of the thirteen code positions, one is reserved for the term code which identifies the term deck to which the card belongs. The two additional digit positions will customarily be used for a sequence number to locate the card within the term deck. The remaining twelve code spaces are available for item codes. These are punched in sequence from left to right across the card, with any excess positions left blank. The Special Index Analyzer recognizes the start of a new term deck by means of an X-punch in column 1 of the first card of the deck, whether the deck contains one or more cards. The X-punch does not interfere with the use of the first column for numeric data, but alphabetic punching should not be used in this column.

CARD FORM

If it is desired to punch additional data into the card, either alphabetic or numeric, successive six-digit fields may be used. The inclusion of additional information will require that the term code and the item codes be shifted to the right, and will reduce the number of item codes that can be fitted on the card. This additional data is not processed by the Special Index Analyzer. The format employed for the term cards is stored in the machine by means of a specially punched card, retained on the alternate program drum of the IBM 26 Card Punch component.

The output of the Special Index Analyzer is printed by means of a typewriter onto a form designed for convenient use by the researcher. It is important to retain a trail of the search, along with the item codes produced by the search. The Special Index Analyzer accomplishes this by first printing across the top of the page the term codes entering into the search, connected by letter symbols standing for the operations used to combine each term file with the result of previous operations. The term codes thus appear in the order in which they were used. The item codes resulting from the sequence of operations are listed in a vertical column down the left-hand edge of the page. This format provides ample room for the researcher to add further notes alongside each item code.

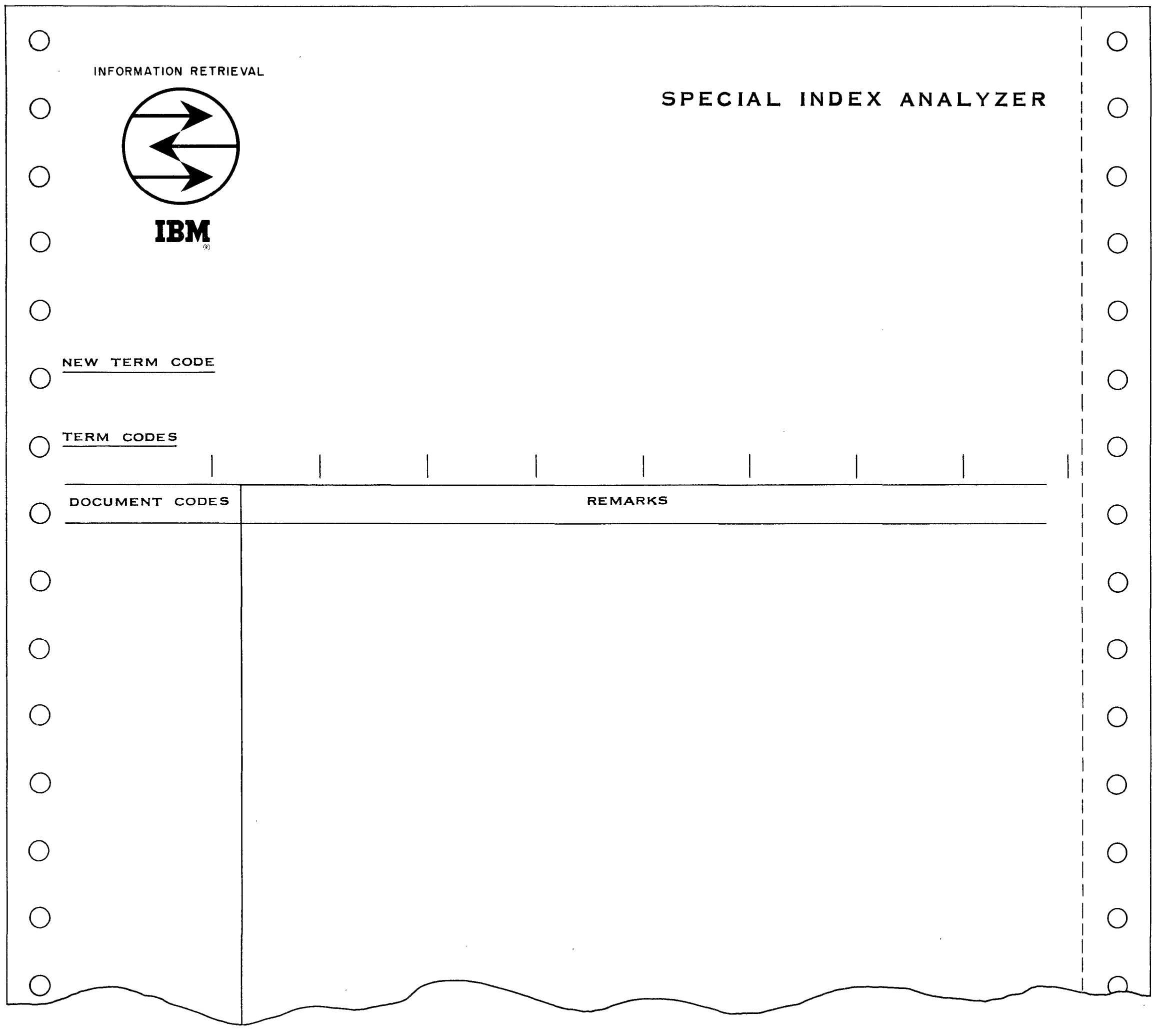
RESULT FORM



IBM 9900 SPECIAL INDEX ANALYZER-TERM CARD

| SEQUENCE NUMBER | TERM CODE | DOCUMENT CODES 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 | 0 0 0 0 0 0 |
| 1 2 | 3 4 5 6 7 8 | 9 10 11 12 13 14 | 15 16 17 18 19 20 | 21 22 23 24 25 26 | 27 28 29 30 31 32 | 33 34 35 36 37 38 | 39 40 41 42 43 44 | 45 46 47 48 49 50 | 51 52 53 54 55 56 | 57 58 59 60 61 62 | 63 64 65 66 67 68 | 69 70 71 72 73 74 | 75 76 77 78 79 80 |
| 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 | 1 1 1 1 1 1 |
| 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 | 2 2 2 2 2 2 |
| 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 | 3 3 3 3 3 3 |
| 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 | 4 4 4 4 4 4 |
| 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 | 5 5 5 5 5 5 |
| 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 | 6 6 6 6 6 6 |
| 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 | 7 7 7 7 7 7 |
| 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 | 8 8 8 8 8 8 |
| 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 | 9 9 9 9 9 9 |
| 1 2 | 3 4 5 6 7 8 | 9 10 11 12 13 14 | 15 16 17 18 19 20 | 21 22 23 24 25 26 | 27 28 29 30 31 32 | 33 34 35 36 37 38 | 39 40 41 42 43 44 | 45 46 47 48 49 50 | 51 52 53 54 55 56 | 57 58 59 60 61 62 | 63 64 65 66 67 68 | 69 70 71 72 73 74 | 75 76 77 78 79 80 |

IBM INFORMATION RETRIEVAL



INFORMATION RETRIEVAL

IBM

SPECIAL INDEX ANALYZER

NEW TERM CODE

TERM CODES

| DOCUMENT CODES | REMARKS |
|---|---|
| | |

## FILE MAINTENANCE

As new items are added to the library, the index file must be extended to include retrieval data for the additions. The first step in the procedure is to assign the item code next in sequence. Then it must be determined which terms of the glossary apply to the new item. A series of cards are punched all containing the code for the new item, and each containing the code for one of the terms determined to be applicable. The standard card form is used for this purpose, but it will normally be of a contrasting color to distinguish it from the regular cards in the file. The term code is punched in the regular field, and the item code occupies the first item field. The other positions of the card are left blank.

At this stage, each addition card must be brought together with the term deck bearing the same term code. It may be decided to bring the term file up-to-date immediately, in which case the term deck is removed from the file and the updating performed. Alternatively, the addition card may be filed as such with the term deck, and the updating postponed until a suitable number of additions have been collected or until the term file is to be used in a search.

The Special Index Analyzer is used to bring the new item codes into the regular term deck. This may be accomplished with the regular union operations applied to the old term file and to each of the addition cards, which carry an X punch in Column 1 as though they were new term decks. The result is punched into blank cards using the appropriate operation. Since non-item data normally are not punched in the punch-out operation, the resulting new term deck is then gang-punched to include the term code, and punched for the sequence number.

Two variations on this procedure may be utilized if desired. To eliminate the external gang-punching of term code, the input deck may be preceded by a command card calling for the "Read in New Term Number" operation, and punched with the term code in the regular field. This term number will then be reproduced into the output deck. The second variation is intended to conserve time where very long term files are to be reproduced. By pressing a special start key, the lack of an X punch in the first term card can be ignored, so that instead of the complete old term deck, only the final card need be brought in.

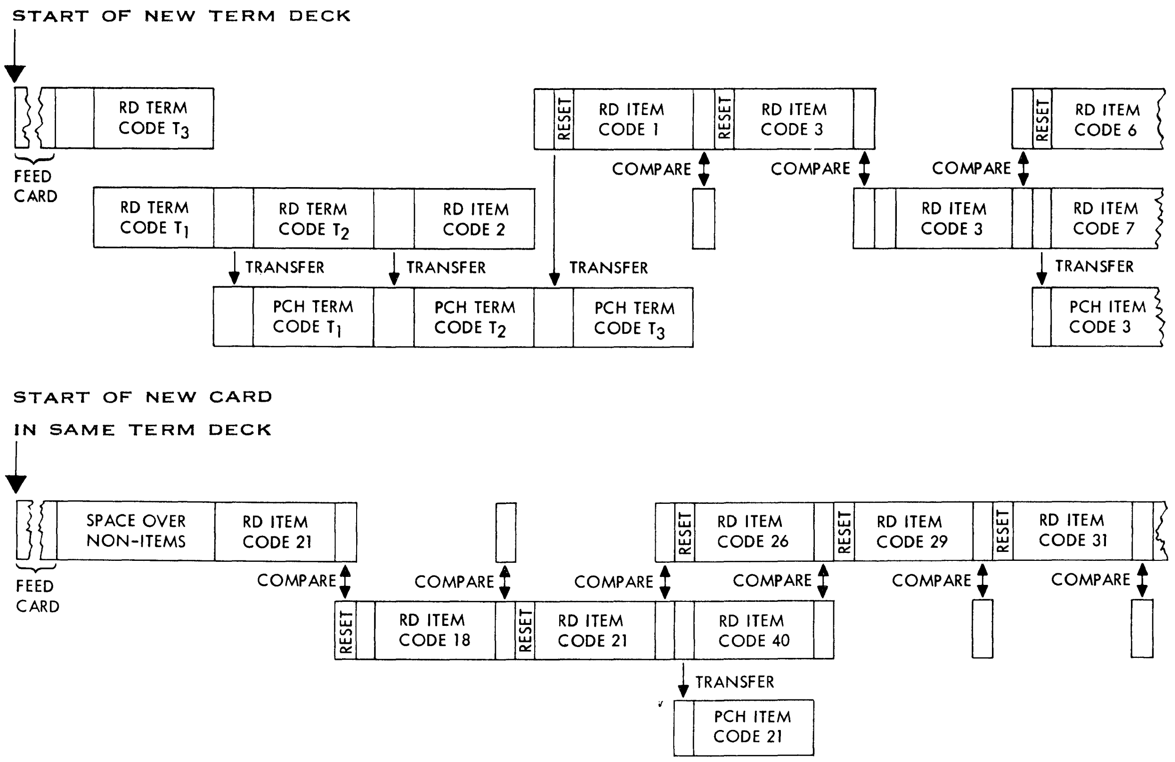
New terms may also be added to the glossary and the appropriate term files built up. If a term is completely new, and appearing for the first time in new items, the previously prepared term files will be unaffected. If however the term was not previously applied to items already in the library, then all of the items where it might be significant must be reanalyzed. The most probable situation will occur where a term file has become too large and the term subdivided into finer categories. In this case, only items in the term file in question must be reinvestigated. It may also be desirable, in the event that certain combinations of term files are repeatedly employed, to add the combination to the index under a new term code standing for the combination. This action adds no new information to the retrieval index, but may add to the convenience or speed of searching.

## SPEED OF OPERATION

In its usual operation, the Special Index Analyzer will be working with term files of various lengths, ranging from tens of items to thousands. The intersection of two term files may also contain a wide range of items, from a small fraction of one of the original files to the entirety of the smaller original file. Thus the operating times experienced in practice will follow a statistical distribution around average values which are typical for the installation.

The basic speed of the Special Index Analyzer is 18 cycles per second. A cycle may consist of reading or punching a character in paper tape or card, typing a character, or performing certain internal operations. In general, each item code of six characters requires 8 cycles (0.444 sec), including 6 cycles for reading into a register, one cycle for comparison and one for re-setting or for transferring the contents of the register. Each of the two inputs and the output channel makes use of a separate register for storing the term or item code and all three can operate concurrently. However, unless the two input registers contain equal codes, only one of the inputs will be actuated prior to processing the next code. The major portion of the time required to perform one pass will be due to the processing of item codes. However, there will also be contributions due to the reproduction of term codes at the start of the pass, and due to the feeding up to the item code portion of each succeeding card in the term deck.

## TIMING OF AN INTERSECTION OPERATION



## TIME FOR ONE INTERSECTION OPERATION

$$\cong 0.25 + \frac{8}{18}i + \frac{l_i}{12} + \left(l_i + l_{ni-1} - l_{ni}\right)\frac{8}{18} \text{ seconds}$$

FEED 1ST CARD

REPRODUCE TERM CODES

FEED + SPACE OVER NON-ITEM DATA IN SUCCEEDING CARDS

"HOUSEKEEPING" STEPS COMMON TO ALL COMBINING OPERATIONS

WHERE:

$i$ = NUMBER OF THE COMBINED TERMS

$l_i$ = NO. OF ITEM CODES IN THE $i^{th}$ TERM DECK

$l_{ni-1}$ = NO. OF ITEM CODES ALREADY ON TAPE

$l_{ni}$ = NO. OF ITEM CODES IN RESULT TAPE

For determining the time required to perform an entire program of several combining operations, it is easiest to determine the total number of distinct (non-overlapping) readings or punchings of item codes. Once this total has been obtained it is multiplied by 0.444 seconds and added to the time required by the housekeeping steps. In all complete programs, there will be in addition to the combining of term files, an initial stage of reproducing the first term deck on paper tape, as well as a final stage of printing out the results.

A program of intersections only is probably the most common type and is therefore most indicative of machine performance. For this type of program, the number of distinct input-output steps is obtained by writing the number involved in each stage:

| | |
|---|---|
| 1st. stage - reproduce $T_1$ on tape | $\ell_1$ |
| 2nd. stage - form $T_1 \cap T_2$ | $\ell_1 + \ell_2 - \ell_{\cap 2}$ |
| 3rd. stage - form $T_1 \cap T_2 \cap T_3$ | $\ell_3 + \ell_{\cap 2} - \ell_{\cap 3}$ |
| $r$ th. stage - form $T_1 \cap \cdots \cap T_r$ | $\ell_r + \ell_{\cap r-1} - \ell_{\cap r}$ |
| $(r+1)$ st. stage - print $T_1 \cap \cdots \cap T_r$ | $\ell_{\cap r}$ |
| | $\ell_1 + \sum_{i=1}^{i=r} \ell_i$ |

Thus, for a program of intersections only, the time required depends upon the total number of item codes involved and upon which term file is chosen to be first, and is independent of the numbers of item codes in the intermediate and final intersections.

This expression presupposes that the reading of paper tape is being performed sufficiently in advance of punching to allow the minimum length of four inches of tape between the stations. If the data to be punched in tape, including both terms and items should be less than six codes, then blanks will be inserted to keep the length of tape at the minimum. These blanks will require additional time to be read on the next pass, however the total time added will, in general, be very small.

The total time required to perform a program of intersections of r term files thus becomes:

$$0.22\,r^2 + 0.47\,r + 0.44\,\ell_1 + 0.52 \sum_{i=1}^{i=r} \ell_i$$

The time required to perform any one of the other set-theoretic operations is exactly the same as for performing the intersection operation. This is because the two input term files are read separately, except when item codes are equal, or in other words, except for the item codes belonging to the intersection. Punching of the result into paper tape always overlaps a reading of one or both inputs, and does not contribute to the time. However, the number of item codes in the intersection of the two operand sets is not generally known beforehand, and therefore the size of the result set from these other operations cannot be predetermined. Unlike the case of the intersection operation performed successively in a program, the size factor does not cancel, but remains significant in programs utilizing these other operations.

An upper bound can be determined for these more complex programs, if it is assumed that there are no item codes common to the intermediate intersections of term files. This assumption is equivalent to assuming that the size of the intermediate term file is the sum of the sizes of the two component term files for the two operations, 'union' and 'union of complement intersections', or that it is the same size as the non-complemented term file for the two 'intersection with complement' operations.

In the case of a program consisting of all union operations for combining term files, the upper bound of the number of distinct input-output operations is:

| | |
|---|---|
| 1st. stage - reproduce $T_1$ | $\ell_1$ |
| 2nd. stage - form $T_1 \cup T_2$ | $\ell_1 + \ell_2$ |
| 3rd. stage - form $T_1 \cup T_2 \cup T_3$ | $\ell_1 + \ell_2 + \ell_3$ |
| $r$th. stage - form $T_1 \cup \cdots \cup T_r$ | $\ell_1 + \cdots\cdots + \ell_r$ |
| $(r+1)$ st. stage - print out $T_1 \cup \cdots \cup T_r$ | $\ell_1 + \cdots\cdots + \ell_r$ |
| | $\sum_{i=1}^{i=r} (r+2-i)\ell_i$ |

## PROGRAMMING THE IBM SPECIAL INDEX ANALYZER

Programming the Special Index Analyzer consists of arranging a sequence of term decks to serve as the input data, and then, at the start of reading each term deck, calling for a particular operation. Two modes of programming the machine are provided, the automatic mode and the manual mode. The automatic mode makes use of command cards in addition to the term cards, and allows the entire search procedure to continue without operator attention, except possibly for reloading the feed hopper and emptying the stacker.

In general, the manual mode will cause the Special Index Analyzer to stop after reading each term deck. The operator causes the machine to continue with the next operation by pressing the appropriate operation button. If the operation is one of the combining types, the Special Index Analyzer will only continue if there is a term deck in the field hopper. The type-out can be called for whether or not there are cards in the hopper. However, if it should be desired to type out an intermediate result and then continue with a combining operation, the paper tape must be repositioned by the operator, so that the last punched term file can be read again in the continuation of the program.

The automatic mode of operation permits as extensive a search as desired to be performed entirely automatically after the term decks have been assembled, placed in the hopper, and the "Start" button pushed. The automatic mode may be used either with or without command cards. A command card is used preceding each term deck for which a combining operation different from the previous is wanted. If no command card is used, the next term deck is combined with no change of operation type. Type-out, however, is not done automatically unless there is a type-out card in the hopper. Thus, if hopper capacity is insufficient for a set of term decks, the machine will stop and await further action by the operator.

| CODE | OPERATION | RESULT SET | TYPE-OUT SYMBOL | DESCRIPTION |
|---|---|---|---|---|
| 0 | TYPE AND PUNCH | | | Type out term and item codes from tape, and punch item codes into term codes. Punch new term code if previously programmed. |
| 1 | READ IN NEW TERM CODE | | | Read in new term code from command card in preparation for final punch-out operation (codes 0 or 7) |
| 2 | INTERSECTION | $T \cap C$ | A | Put item codes common to term file from tape and to term file from cards into tape. |
| 3 | INTERSECTION WITH COMPLEMENT FROM TAPE | $\bar{T} \cap C$ | B | Put into tape item codes from cards, provided they are not read from tape previously prepared. |
| 4 | INTERSECTION WITH COMPLEMENT FROM CARDS | $T \cap \bar{C}$ | C | Put into tape item codes not on cards but in tape. |
| 5 | UNION OF COMPLEMENT INTERSECTIONS | $(\bar{T} \cap C) \cup (T \cap \bar{C})$ | D | Put into tape item codes not common to tape and cards. |
| 6 | TYPE OUT | | | Type out last-punched paper tape in standard format. |
| 7 | PUNCH OUT | | | Punch out last-punched item codes from tape in term card format. Punch new term code if previously programmed. |
| 8 | UNION | $T \cup C$ | E | Put into tape item codes appearing in either or both tape and cards. |

## RULE FOR TIMING A TYPICAL SEARCH

The total time $T$ required to perform a program of intersections of $r$ term decks is:

Total time, $T = 0.22r^2 + 0.47r + 0.44\ell_1 + 0.53 \sum_{i=1}^{i=r} \ell_i$

Where:

| | |
|---|---|
| $r$ | = number of term decks |
| $\ell_1$ | = number of items in the ith term deck |
| $i$ | = 1, 2, 3, etc. depending upon whether it is the lst, 2nd, or 3rd, etc. term decks |
| $\ell_i$ | = number of items in the ith term deck |

Following is an example of the calculation of time required for the intersection of three term decks.

| Terms | Deck | Cards per Deck | | Items per Deck |
|---|---|---|---|---|
| Reliability | second | 25 | (25· 12) = | 300 |
| Transistors | first | 20 | (20· 12) = | 240 |
| Digital Computers | third | 30 | (30· 12) = | 360 |
| | | 75 Cards | | 900 Items |

Result of typical intersection:

T = (0. 22· 9) + (0. 47· 3) + (0.44· 240) + (0. 53· 900)
1. 98 + 1. 41 + 105. 60 + 477. 00 = 585. 99 seconds

$\frac{585.99 \text{ seconds}}{60 \text{ seconds}}$ = 9. 77, or 10, minutes required machine time for the intersection of three term decks.

IBM®

International Business Machines Corporation

590 Madison Avenue, New York 22, New York

PRINTED IN U.S.A.