VII. THE SENTENCE MATCHING PROGRAM - GRAPH

Edward H. Sussenguth, Jr.

1. Introduction

A fundamental hypothesis of automatic document classification is that the words of the document itself can be used to detect the subject matter; words of particular significance are those which occur frequently in a given text. Commonly occurring words, such as articles, conjunctions, and prepositions, are normally ignored, so that only certain high frequency words are counted. In addition, words may be normalized by removing plural, adjective, and verb suffixes, so that variations of a basic stem are grouped into the same class. Synonym dictionaries may also be used to group words with similar meanings. After the normalizing and counting processes, those word classes having the highest frequency of occurrence are considered indicative of the document content.

A refinement of this statistical procedure is to consider the words taken in pairs or triples, not just individually. By so doing, compound and refined concepts, which are not adequately described by individual words, can be detected; the procedure may then become sensitive to phrases such as "information retrieval" and "data processing." However, if only adjacent word pairs were to be used as part of the phrase-counting process, a large amount of relevant information would be lost, as may be seen by considering the following four sentences:

(1) Light radiation occurs above the critical temperature.

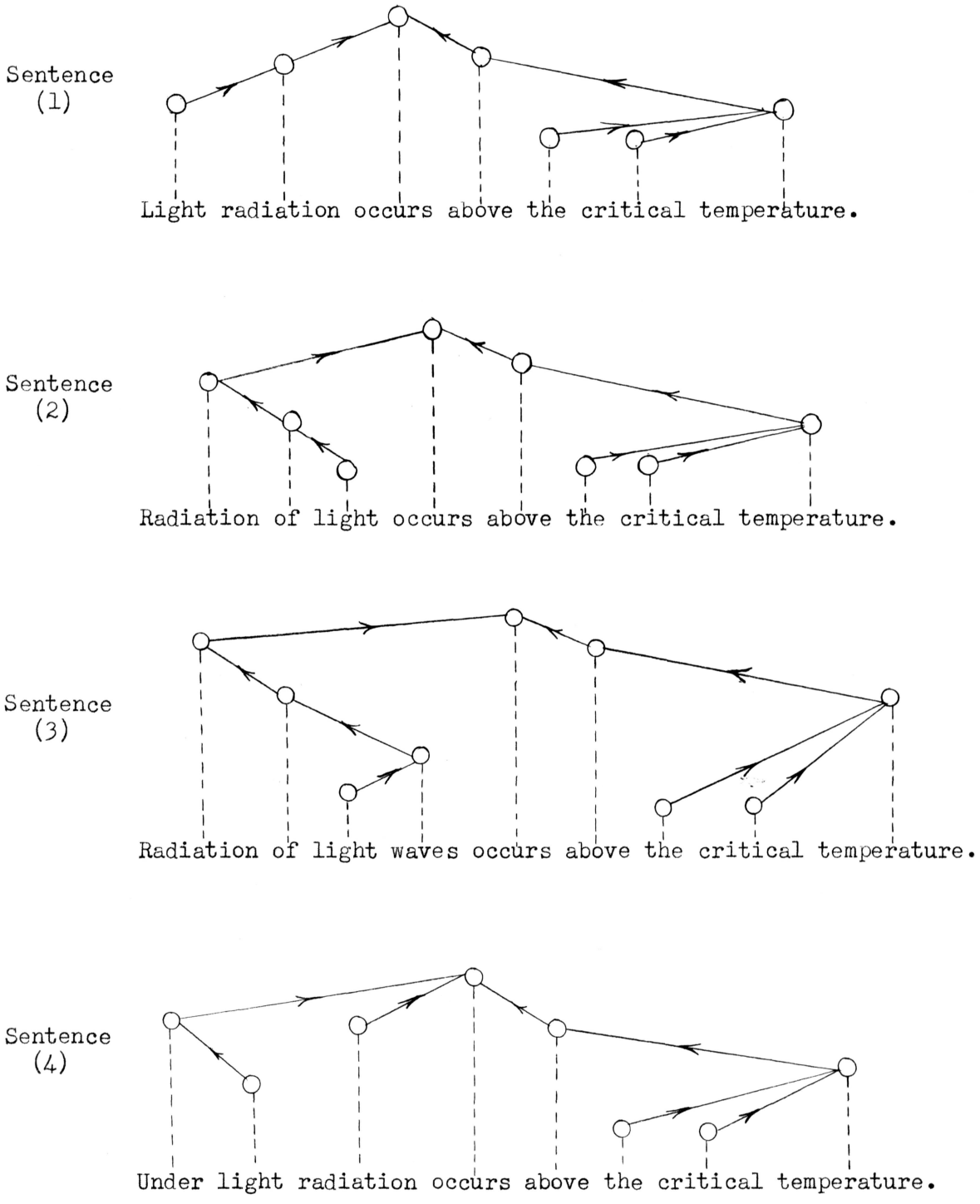
(2) Radiation of light occurs above the critical temperature.

(3) Radiation of light waves occurs above the critical temperature.

(4) Under light radiation occurs above the critical temperature.

The first three sentences all express the idea that energy in the form of light is emitted above a certain temperature. However, only in sentence (1) does the word pair "light radiation" occur explicitly. Moreover, although the pair "light radiation" occurs in sentence (4), that sentence does not indicate that the radiation is in the form of light. What is important, then, is not the occurrence of the word pair "light radiation," but the fact that the word "radiation" is modified by the word "light." To use this type of information, the sentence must be analyzed syntactically. It has been demonstrated that it is possible to perform such an analysis automatically, so that the syntactic relations between words can be used in automatic document classification systems.

A tree is a convenient representation of syntactic relations between the words of a sentence. In the simplest form of the syntactic dependency tree, the words of the sentence and the nodes of the tree are in a one-to-one correspondence, and if word x is dependent upon word y, there is a branch (x,y) in the tree. The dependency trees for sentences (1), (2), (3) and (4) are shown in Fig. 1. Although such dependency trees exhibit the direct syntactic dependence of one word upon another, they do not show

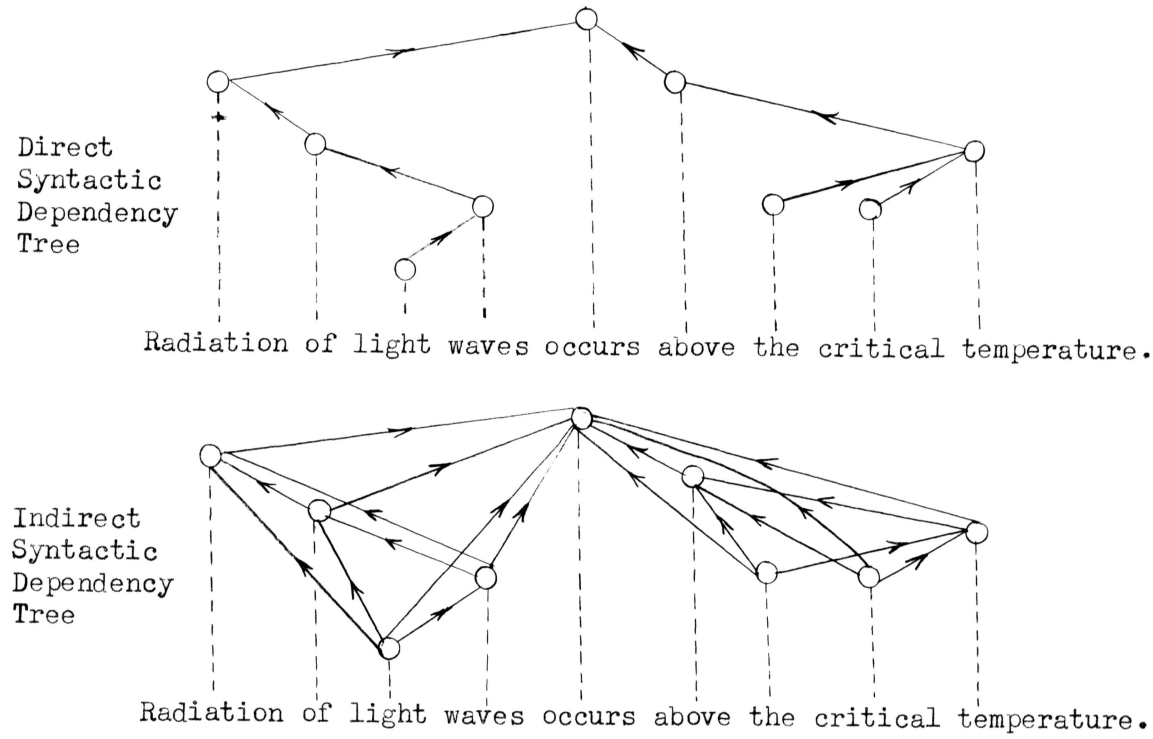


Syntactic Dependency Trees

Figure 1

indirect relations explicitly. Indirect dependencies may be exhibited in an "indirect dependency" tree, which is derived from the direct dependency tree by adding branches (u,v) between each node u and all nodes v reachable from u. An example is shown in Fig. 2. Phrases and other combinations of words and modifiers are reflected in the tree structures as subgraphs or partial subgraphs, and graph-matching techniques may be used to detect the occurrence of specified phrases within a syntactically analyzed sentence. In the previous example, the tree of the phrase "light radiation" (see Fig. 3) is a subgraph of the direct dependency tree of sentence (1), and is a subgraph of the indirect dependency trees of sentences (2) and (3), but is not a fragment of either dependency tree of sentence (4).

As explained in previous sections, the statistical (word counting), semantic (synonym dictionary), and syntactic (grammatical dependence) properties of the words of a text have all been incorporated in the SMART system.[1] The theoretical implications of these properties and the implementation of most of the properties on a digital computer have been discussed in other sections of this report. The present section contains a discussion of the implementation of the test to determine whether or not a particular phrase is contained within a specific sentence. There are three major parts to this test: (1) the syntactic analysis of the sentence, (2) the representation of the analyzed sentence in a tree form, and (3) the matching of a phrase (called a criterion tree) with the sentence tree. In the SMART system the syntactic analysis is performed by the program developed by Kuno and Oettinger,[2] the construction of the



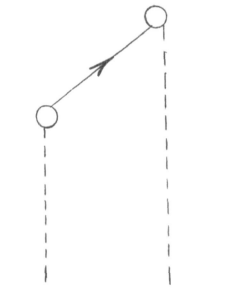
	r	o	l	w	o	a	t	c	t
Radiation				1					
of	1			1					
light	1	1		1	1				
waves	1	1			1				
occurs									
above					1				
the					1	1			1
critical					1	1			1
temperature					1	1			

Direct Connection Matrix

Indirect Connection Matrix

Direct and Indirect Syntactic Dependency Trees

Figure 2



Query Phrase
Figure 3

tree representation follows the outline given by Lemmon (see ref. 3 and Sec. VI of this report) and the matching test is a modification of the structure-matching algorithm described by Sussenguth.[4,5] Some of the details of the modified structure-matching program are presented below.

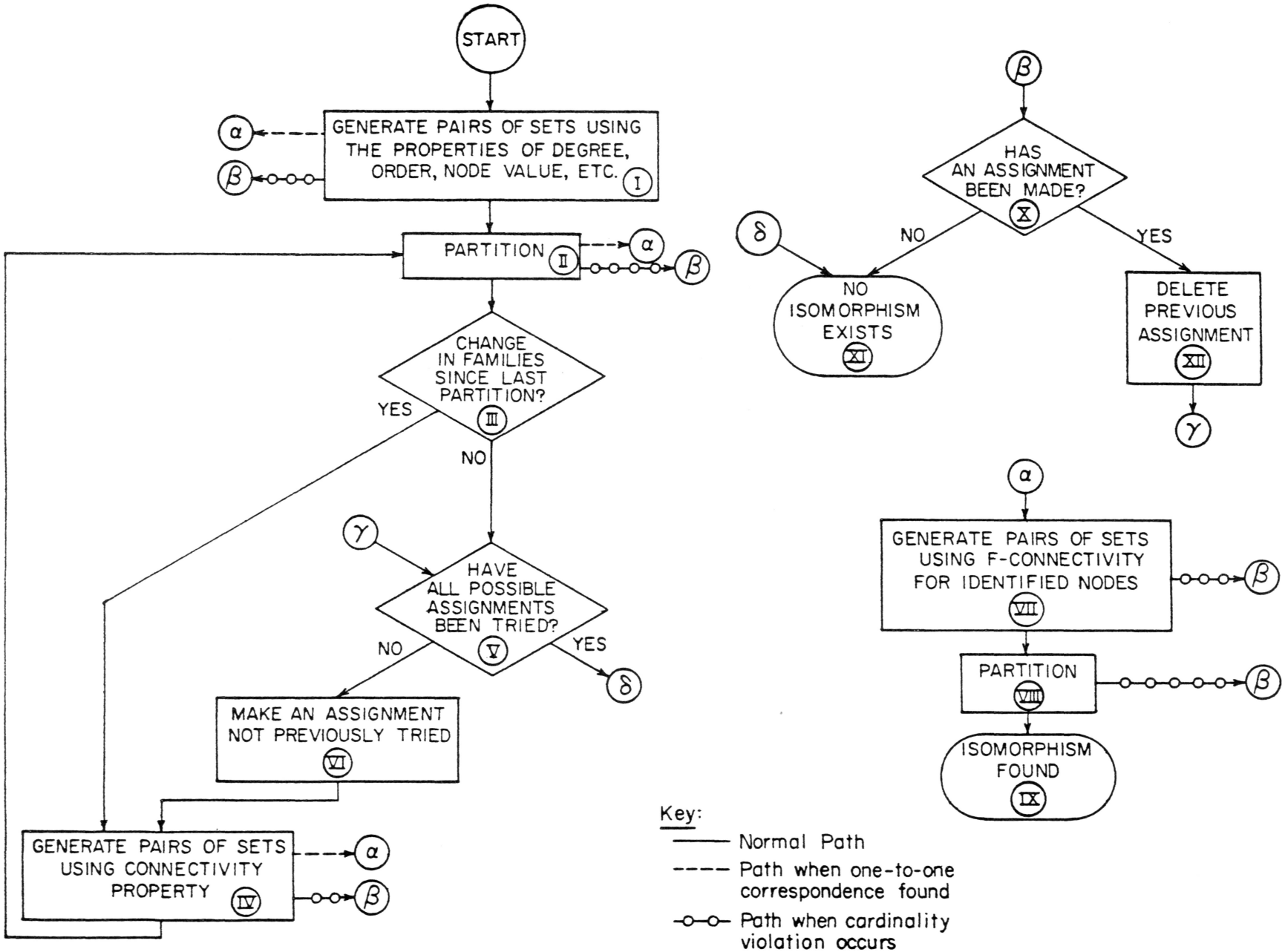
2. The GRAPH Structure-matching Program

GRAPH[†] is a program for testing if criterion graph C is a partial subgraph of sentence graph S. The program is specifically tailored for testing trees which represent syntactically analyzed English sentences. Figure 4 is a flow diagram of GRAPH.

The properties used to generate sets in box I of Fig. 4 are:

node values - synonym class types
node values - grammatical type
f-degree - direct connections

[†]Familiarity with either ref. 4 or 5 is assumed.



Flow Diagram of Graph

Figure 4

b-degree - direct connections

f-degree - indirect connections

b-degree - indirect connections.

In box IV the connectivity property is used:

f^k-connections - direct connections

b^k-connections - direct connections

f^k-connections - indirect connections

b^k-connections - indirect connections.

The partitioning procedure (box II), the assignment process (boxes V, VI, X, and XII), and the method of family formation exactly follow the descriptions given in ref. 4 (specifically, in Secs. 3.3A and 3.3C, 3.4A, and 3.3E, respectively). The graph transformations which delete discarded nodes of S and pairs of corresponding nodes from C and S from pairs of corresponding sets are included (see Sec. 3.3D of ref. 4).

The program is designed to accommodate sentence trees of up to 144 nodes, and criterion trees of up to 36 nodes. The trees are entered into the program in the form of binary connection matrices (both direct and indirect), and the values associated with the nodes are specified by a binary node value matrix. When the program terminates in either box IX (isomorphism) or box XI (no isomorphism), the correspondences which have been found between nodes are listed in a table called KNOWN. The form of this listing is given in Appendix A.

GRAPH has been programmed for and tested on the IBM 7094. The calling sequence for GRAPH is given in Appendix A of this section, and the subroutines called by GRAPH are described in Appendix B.

A number of tests have been made with GRAPH. In most cases the properties used to generate sets in box I were sufficient to delineate a one-to-one correspondence or to indicate that no isomorphism is possible. (In other words the most frequent path followed in the flow diagram is the one from box I to II to either VII or XI.) For some examples, it was necessary to use the connectivity property (box IV) once, in order to delineate a one-to-one correspondence. Only to resolve multiple isomorphisms, reflecting a parallel construction in the sentence, was it ever necessary to partition (box II) more than twice. In general the computing time required by GRAPH is of the order of tens of milli-seconds; an upper bound is one second.

APPENDIX A

PROGRAMMING DETAILS FOR GRAPH

1. Calling Sequence

```
CALL GRAPH, S, C
```

or:

```
TSX   GRAPH, 4
TSX   S
TSX   C
TSX   *
TSX   *
```

where:

S is address of data for sentence graph

C is address of data for criterion graph

2. Return

Accumulator = 1 if C is a partial subgraph of S

Accumulator = 0 if C is not a partial subgraph of S

The correspondences which are determined by the process are listed at location KNOWN.

Location	Criterion Node (Decrement)	Sentence Node (Address)
KNOWN	not used	
KNOWN + 1	a	ø(a)
KNOWN + 2	b	ø(b)
...	...	...
KNOWN + k	c	ø(c)

Notes: k is an integer (in location 77456_8) stating how many correspondences exist. If an isomorphism is found, k = n; if there is no isomorphism, $0 \leq k < n$.

The order of the listing is determined by the order in which the correspondences are found.

3. Data Format

A	number of nodes (n) in graph (in the decrement)
A+1	not used by GRAPH
A+2	number of rows (r) in value matrix (in the decrement)
A+3	value matrix
A+3+w·r	indirect connection matrix
A+3+w·(r+n)	direct connection matrix

where

A is S or C

w is ceiling (n/36).

4. Miscellaneous

Tape B1 is used as a scratch tape.

10311_8 words are used for GRAPH and its associated subroutines.

Locations 31402_8 to 77461_8 of COMMON are used.

Running time is about one second.

APPENDIX B

SUBROUTINES USED BY GRAPH

Name	Function
BKBAR	Performs the set operation $A = B \cap C \cap \bar{D}$
CANDID	Computes those nodes of S which have not been deleted
CDEG	Computes f-degrees
CEL36	Computes ceiling (n/36)
{ECVQ FY}	Partitioning routine (see ref. 4, Secs. 3.3A, 3.3C)
FORM	Family formation routine (see ref. 4, Sec. 3.3E)
FRST	Finds first 1 in a binary vector
GAMMA	Computes f-connectivity of a set
ISIV	Tests if the ith bit of a binary vector is 0 or 1
LAST	Finds last 1 in a binary vector
ORFAM	Given sets $A_1, A_2, \ldots, A_r$, the sets $B_1, B_2, \ldots, B_r$ are computed where $B_r = A_r$ and $B_i = A_i \cup B_{i+1}$
SETIF	Sets ith bit of a binary vector to 0
SETIN	Sets ith bit of a binary vector to 1
SIGV	Counts number of 1's in one machine word
SIGVW	Counts number of 1's in w machine words
TRANS	Transposes a logical matrix

REFERENCES

1. Salton, G., "A Flexible Automatic System for the Organization, Storage and Retrieval of Language Data (SMART)," Information Storage and Retrieval, Report ISR-5 to the National Science Foundation, The Computation Laboratory of Harvard University (January 1964).

2. Kuno, S., and Oettinger, A. G., "Multiple-path Syntactic Analyzer," Proc. of the IFIP Congress-62, Munich (1962).

3. Lemmon, A., "Automatic Identification of Phrases for Document Classification," Information Storage and Retrieval, Report ISR-5 to the National Science Foundation, The Computation Laboratory of Harvard University (January 1964).

4. Sussenguth, E. H., Jr., "Structure Matching in Information Processing," Information Storage and Retrieval, Report ISR-6 to the National Science Foundation, The Computation Laboratory of Harvard University (April 1964).

5. Sussenguth, E. H., Jr., "Automatic Structure-Matching Procedures," Information Storage and Retrieval, Report ISR-5 to the National Science Foundation, The Computation Laboratory of Harvard University (January 1964).