SOME FLEXIBLE INFORMATION RETRIEVAL SYSTEMS
USING STRUCTURE MATCHING PROCEDURES*

Gerald Salton and Edward H. Sussenguth, Jr.
Computation Laboratory, Harvard University
Cambridge, Massachusetts

INTRODUCTION

The comparison between stored information identifications and requests for information is one of the principal tasks to be performed in automatic information retrieval. In so-called descriptor systems, where information is represented by sets of independent key words, this operation is relatively simple, since it consists of a comparison between the respective "vectors" of key words. In many retrieval systems it has been found necessary or expedient to use more complicated constructs for the identification of information. Notably "role" indicators are often added to identify various types of key words, and "links" specify a variety of relations between key words. A complete identification for a document or an item of information is often represented by a graph, consisting of nodes and branches between nodes, to identify respectively the key words and relations between key words. The matching of such information graphs with graphs representing requests for information is a relatively complicated and time consuming operation, particularly since the request structure can be made to match the information structure only partially and incompletely.

The graph matching problem arises also in document retrieval systems where certain significant portions of text are extracted and compared with the search requests. In such cases, it is possible to represent the syntactic structure of the text excerpts by abstract trees, and a tree matching procedure becomes necessary to compare the extracted information with the requests. As before, an exact matching procedure would not be very helpful, since many different ways can be found to express the same ideas or requests. What is needed instead is a procedure which permits inclusion of partly unspecified information, and which provides for the possible relaxation of the various conditions that render a complete match impossible at any given time. Graph matching techniques are, of course, also applicable to the comparison of items of information which exhibit inherently a multi-dimensional structure, such as electrical or pipeline networks, street or geographical maps, chemical molecular structures, and so on.

Extensive experience has been gained in the past with structure matching programs which operate on a "node-by-node" or a "piece-by-piece" basis. In the node-by-node approach, the nodes of the two structures are compared one at a time, until either the complete structures match, or else an incompatibility arises; in the latter case it becomes necessary to backtrack to a point where there is agreement and try again with different elements. In the piece-by-piece approach, a dictionary of basic substructures is used to break a given structure into pieces which are then matched as a whole.

* This study was supported in part by the National Science Foundation under Grant GN–82.
Neither of the two techniques works well for any but the simplest structures. The node-by-node method usually requires extensive backtracking, involving the comparison of many hundreds of nodes for even very simple structures. The piece-by-piece approach, on the other hand, suffers from the fact that no standard, well-defined method exists for breaking a given structure into substructures.

A topological structure-matching procedure has been programmed for the 7090 computer which does not depend on a specific ordering of the nodes, nor on the presence or absence of certain specified substructures. Little or no backtracking is required, and the method can be used to detect complete as well as partial matches. The basic idea is to determine certain simple properties of the nodes of the two structures to be matched, and to equate those subsets of the nodes in the two structures which exhibit equivalent properties. A standard procedure is then used to form new matching subsets, and to break down already existing subsets into sets with fewer members. The procedure is completely determinate except in cases where it is necessary to resolve certain symmetries in the connection pattern of the nodes; in that case a guess (assignment) is made as to the correct solution; such a guess may later prove to have been right or wrong, and if wrong, may require some backtracking. In most practical problems, however, little backtracking seems to be needed.\footnote{Computer experiments indicate that the topological procedure is much more efficient than either the “node-by-node” or the “piece-by-piece” approach. Operating automatic retrieval systems based on the use of relatively complex structures (as opposed to sets of unconnected key words) seem therefore to become a practical possibility instead of merely a theoretically desirable goal.}

An example is given first to illustrate the partial matching procedure, as well as the methods which may be used to alter one or both of the structures to be compared in order to make a match between them more likely. The procedure is then applied to the matching of document graphs with request graphs, and to a retrieval system based on the comparison of syntactically analyzed document excerpts with a stored phrase dictionary.

**THE STRUCTURE MATCHING PROCEDURE**

Consider first the problem of determining whether the graph of Fig. 1(b) is contained in the graph of Fig. 1(a), that is whether Fig. 1(b) is a subgraph \$\$ of Fig. 1(a). The nodes in the two structures are labelled arbitrarily from (1) to (6) and from (9) to (6) respectively, and the connection pattern of the nodes is represented by the binary connection matrices \$\$ shown in Figs. 2(a) and 2(b). Since no additional information is furnished about either the nodes or the branches of the two graphs under consideration, all relevant properties of these graphs are in fact derivable from the con-

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig1.png}
\caption{Directed Graphs.}
\end{figure}

\footnote{Theoretical foundations and proofs of convergence of the method are given in detail in Reference 10. The theory as well as applications to chemistry are also more fully treated in References 11 and 12.}

\footnote{\textit{A graph} \(G\) consists of a set \(X\) (the nodes) and a set of relations between certain pairs of nodes (the branches). A matrix \(C\) such that \(C_{ij} = 1\) whenever there is a branch from node \(x_i\) to node \(x_j\), and is 0 otherwise, is called the \textit{connection matrix} of graph \(G\). A \textit{subgraph} \(H\) of \(G\) is obtained by removing from \(G\) certain nodes as well as all branches adjacent to the removed nodes. A \textit{partial graph} \(J\) of \(G\) is obtained by removing from \(G\) some of its branches. A \textit{partial subgraph} \(K\) of \(G\) is a subgraph of a partial graph of \(G\). A \textit{completed partial subgraph} \(L\) of \(G\) is a partial subgraph to which branches are added so as to preserve all original paths between the nodes included in \(L\); specifically, if node \(z\) is removed from \(G\) in forming \(L\) and if there exist paths from \(x\) to \(z\) and from \(z\) to \(y\) in \(G\), then a path exists from \(x\) to \(y\) in \(L\) for all \(x, y\) included in \(L\).}
I 2 3 4 5 6 7 8 9
1 1
2 1 1 1
3 1 1
4 1 1
5
6 1
7
8
9 1 1

a) DICTIONARY STRUCTURE
b) QUERY STRUCTURE

Figure 2. Connection Matrices for Graphs of Figure 1.

Some flexible information retrieval systems are therefore based on the manipulation of binary matrices of the type shown in Fig. 2.

The computer program is based on the manipulation of binary matrices. The heart of the algorithm consists in using various properties of the nodes and/or branches of the graphs in order to generate pairs of sets which must match if the two graphs are eventually to match. The following properties are particularly useful for this purpose:

a. the kth order outward (or inward) degree of the nodes, that is the number of nodes reachable from a given node by outgoing (or incoming) paths of length k;

b. labels or identifiers which may be associated with nodes or branches;

c. the connectivity patterns of sets of nodes, that is, the nodes reachable from a given set of nodes by paths of length k.

The procedure for the graphs of Fig. 1 is outlined in Fig. 3. Only connections of length 1 have been used to simplify the exposition.

The initial set correspondences are shown in Fig. 3(a), lines I to IV. Set II, for example, is constructed by noting that the set of all nodes of outward degree 2 in the query structure, must correspond to the set of all nodes having at least outward degree 2 in the dictionary structure. The only node of outward degree 2 in Fig. 1(b) is \( \textcircled{5} \); there are four nodes in Fig. 1(a) that have outward degree 2 or greater; \( \textcircled{5} \) must therefore correspond to either nodes \( \textcircled{2}, \textcircled{3}, \textcircled{4} \) or \( \textcircled{6} \).

At this point it is necessary to generate smaller sets from the ones shown on lines I to IV of Fig. 3(a). This is done by noting, for example, that the set of query nodes contained in both sets I and III of Fig. 3(a) can correspond only to dictionary nodes which are also contained in sets I and III (plus possibly in other sets). The only query node contained in both sets I and III is \( \textcircled{5} \); in the dictionary structure, nodes \( \textcircled{5}, \textcircled{6}, \textcircled{7}, \textcircled{8} \) are both in sets I and III, so that set \( \textcircled{5} \) must be contained in sets \( \{2, 4, 6, 8\} \). The set of possible correspondents of node \( \textcircled{5} \) has then been reduced from the seven nodes of set I to the four nodes of set VI. This “set partitioning” procedure is performed by the computer by comparing the columns of the binary matrices exhibited in
Fig. 4. Node δ, for example, has column vector 1010; the only nodes of the dictionary structure including the pattern 1010 are nodes δ, ζ, ι, and δ with vectors 1111, 1110, 1010, and 1111 respectively.

The partitioning process yields three new sets labelled V, VI and VII. New sets (VIII to XI) are added using outward and inward connections of length 1 from the sets V, VI, and VII. (The outward connection of the set {c} is empty, and therefore it is not included in the table.) The partitioning process is repeated, yielding sets XII, XIII, and XIV. Since the possible correspondents of node δ have changed, it is not redundant to test the connectivity again. When this is done and another partition performed, sets XVI, XVII, and XVIII result. These pairs of sets are identical to those produced by the previous partitioning, and no simple properties of the nodes can be used at this point to generate new sets which would in turn result in a reduced partition.

An “assignment” is therefore made by postulating the correspondent for node δ (set XIX). This assignment represents a guess which must later be verified for correctness. Partitioning of the sets XVI to XX yields the sets XXI to XXIII. New assignments (not shown in Fig. 3 (a)) of δ first to ζ and then to ι then produce two one-to-one correspondences between the graphs of Fig. 1:

If it is desired to obtain other possible correspondences, it is now necessary to go back to the sets XVI to XVIII and attempt other assignments for node δ. Sets XIX' to XXIII' of Fig. 3(b) illustrate the assignment δ ↔ ι. This assignment yields a partition which is seen to be improper since node δ cannot be included in the empty set. The assignment δ ↔ ι is therefore not useful since it leads to an incompatibility. The other two possible assignments for node δ, do, however, furnish acceptable one-to-one correspondences as follows:

\[
\begin{align*}
\delta & \leftrightarrow \iota \\
\iota & \leftrightarrow \zeta \\
\zeta & \leftrightarrow \iota \\
\iota & \leftrightarrow \zeta
\end{align*}
\]

The four mappings obtained are easily verified by comparing Figs. 1 (a) and 1 (b).

A flowchart of the complete procedure is shown in Fig. 5. The procedure is seen to be iterative since the generation of corresponding sets is followed by a partitioning process, followed again by the generation of new sets, and so on. Alternate applications of partitioning followed by formation of new sets will result in one of three situations:

1. the membership of each set is reduced to one, thus exhibiting the complete match between the given structures;
2. an incompatibility arises between pairs of corresponding sets, that is, a cardinality
violation is found to exist between pairs of corresponding sets; in that case no match exists in general;

3. no incompatibility arises but repeated application of the partitioning procedure will not result in the formation of new sets; in that case, more than one match is generally possible, and it is necessary to perform an arbitrary assignment of correspondents for one of the nodes.

If a cardinality violation is detected, that is, if, for example, a set A is found to be included in a set B which has fewer members, as happened in the example for sets XXIII', then the two structures being compared obviously cannot match. The comparison process can therefore be stopped immediately, unless the incompatibility resulted from a previous assignment; in the latter case, only that particular assignment can be discarded, and other possible assignments must be tried before deciding that the two structures do or do not match. The procedure to be followed in case of cardinality violation is shown by broken lines in Fig. 5. In practice, cardinality violations normally arise early for graphs which do not match, so that the procedure is very rapid in such cases.

THE ADAPTIVE MATCHING PROCESS

In the example described in the preceding section, four isomorphisms were detected between the graph of Fig. 1(b) and that of Fig. 1(a). Clearly, it is possible to increase or decrease the number of matches (or, alternatively, to increase or decrease the probability of a match between any two given structures) by suitably relaxing or tightening the conditions which affect the matching process. If, for example, the unilateral connections (directed branches) in Fig. 1 are replaced by bilateral connections, and therefore the non-symmetric connection matrices of Fig. 2 are changed into the symmetric ones of Fig. 6, then eight additional isomorphisms will be found between the two graphs. In fact, three different isomorphisms will then exist between the “triangle” \{a, b, c\} and each of the triangles \{2, 6, 7\}, \{4, 2, 8\} and \{9, 4, 8\} included in the dictionary structure.

Another possible way of relaxing the conditions which are operative during the matching process is to permit the introduction between any two nodes in the query structure of a variable number of intermediate nodes. This process replaces the query structure of Fig. 1(b) by the new structures of Figs. 7(b) and 7(d). (The broken lines indicate indirect connections.) Since each of the intermediate nodes may or may not match a given node in the dictionary structure, it is now necessary to test whether the query structure is a completed partial subgraph (rather than a subgraph) of the dictionary structure. A comparison of Fig. 7(b) with the partial subgraph of Fig. 1(a) represented as Fig. 7(a), and a comparison of Fig. 7(d) with Fig. 7(c) reveals at least two additional completed partial subgraph matches that could be obtained in addition to the four subgraph matches already exhibited in the preceding section. Further completed partial subgraph matches not shown in Fig. 7 are also possible.

![Figure 6. Symmetric Connection Matrices Derived from Graphs of Figure 1.](image)

![Figure 7. Matching Partial Subgraphs.](image)

Non-matching graphs of fifty nodes required an average of less than one-half millisecond on the 7090 computer during a test run.
To determine whether a graph is a completed partial subgraph of another graph, it is no longer sufficient to know whether two nodes are directly connected or not, but it is also necessary to know whether a (possibly indirect) path exists between any pair of nodes. Thus the connection matrices of Fig. 1 must be replaced by the "path matrices" shown in Fig. 8 in which the i-jth element is 1 whenever a path exists from node i and to node j. Each 1 in the matrices of Fig. 8 thus indicates either a direct or an indirect connection between the corresponding nodes, and use of the algorithm of Section 2 with the path matrices of Fig. 8 (instead of the connection matrices of Fig. 2) will generate the isomorphisms derived in Figs. 3 as well as a number of additional completed partial subgraph matches including those exhibited in Fig. 7.

Consider now, on the other hand, methods which will tighten the requirements to be met for a satisfactory match. Instead of specifying less information than for the directed subgraph comparison, it is now necessary to add restrictions to the graph of Fig. 1. A possible method consists in adding labels to the unlabelled branches of the graph to simulate, for example, various types of relations between the nodes. Another possibility is the addition of labels to the nodes of the graph so as to restrict the correspondents of a given labelled query node to only those nodes in the dictionary structure which carry the same label.

Consider first the two graphs shown in Fig.

** The path matrix may be generated automatically as a sum of powers of the corresponding connection matrix.18
and, obviously, given a pair of corresponding sets not only must the nodes match as before, but the branch labels must match as well. The matching procedure is illustrated in Fig. 11.

The branch labels make it possible to generate a large number of sets at the outset. The set partitioning procedure illustrated by the set inclusion matrix of Fig. 11(b) then results in the formation of the three small sets reproduced in the figure. Assignment of (3) to either node 2 or node 5 finally produces two one-to-one mappings as follows:

\[
\begin{align*}
(3) & \leftrightarrow (6) \\
(6) & \leftrightarrow (5) \\
\end{align*}
\]

and

\[
\begin{align*}
(6) & \leftrightarrow (7) \\
(7) & \leftrightarrow (5) \\
\end{align*}
\]

A comparison of the graphs of Fig. 9 can be used to verify that these two mappings are the only ones which obey the branch labelling restrictions.

As a last extension, consider now the two Syntol graphsΩ of Fig. 12. These graphs correspond to an actual document abstract and to a search request, respectively, as encoded under the Syntol system, and may be seen to be identical with the structures of Fig. 9 except for the added node labels. In order fully to represent the system, it is now necessary to add node label matrices to the connection matrices and to the branch label matrices. The node label matrices may be represented either as a table including all the node names together with the (possibly vacuous) corresponding labels, or alternatively as a full matrix whose i-j<sup>th</sup> element is 1, whenever label j is attached to node i. The node labels serve the same purpose as the branch labels, since they restrict the number of possible correspondents of a given node to only those nodes which either carry the same label, or else carry no label, thus indicating that they can match any node whatsoever that satisfies the remaining restrictions.

The procedure used to determine whether the query graph of Fig. 12 is a subgraph of the document graph is outlined in Fig. 13. Since all nodes are labelled, an immediate correspondence is established between the nodes of the two graphs under consideration (sets I, II and III of Fig. 13). It remains to determine whether the connections and branch labels are preserved. Sets IV of Fig. 13 reveal an incompatibility, since node (3) has two outgoing branches with a branch label (3), whereas the corresponding node 3 has only one such outgoing branch. Since a set containing two elements cannot be contained in a set containing only one element,

<table>
<thead>
<tr>
<th>Set Number</th>
<th>Criterion for Set Generation</th>
<th>Corresponding Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Node label “cortex”</td>
<td>(a) (\subseteq) (3)</td>
</tr>
<tr>
<td>II</td>
<td>Node label “nerf optique”</td>
<td>(b) (\subseteq) (8)</td>
</tr>
<tr>
<td>III</td>
<td>Node label “chronaxie”</td>
<td>(c) (\subseteq) (7)</td>
</tr>
<tr>
<td>IV</td>
<td>Direct connections from set 1 with branch label (3)</td>
<td>(d) (\subseteq) (7)</td>
</tr>
</tbody>
</table>

Figure 13. Set Correspondences for Graphs of Figure 11 Using Direct Connections.
the subgraph test fails, and there is no need to proceed further.

It is therefore necessary to relax the matching conditions by taking into account indirect connections and intermediate nodes. The path matrix of Fig. 8 is now used to verify that complete paths (rather than direct connections) and path labels are preserved by the correspondence in the node labels. The set correspondences, shown in Fig. 14, demonstrate that to each outgoing and incoming labelled path in the query structure there corresponds a path with similar properties in the document structure. The query graph therefore matches the document graph when indirect connections are taken into account.

An adjustable procedure for the comparison of query and dictionary structures can now be outlined. The process uses the same matching algorithm throughout, and is modified only by altering the matrices which represent the connection patterns and the branch or node labels. The exact strategy used in the progressive alteration of the matrices may be made to depend on the type of document collection being processed, and on preliminary retrieval tests. Clearly, the weaker the restrictions which affect the matching process, the more matches are likely to be obtained, and the larger therefore the collection of answers to a given search request.

In general, elimination of the branch labels from the query and document graphs reduces a variety of possible relations between terms to a single one (represented by an unlabelled branch). Replacement of directed by non-directed branches further reduces the ability to discriminate between a variety of relations, since a relation from A to B is now equivalent to one from B to A. Finally, removal of node labels simplifies both the search requests and the document identifications, since it eliminates from consideration some of the terms used as identifiers.

A possible strategy for the gradual broadening of matching criteria is as follows:

a. Use unmodified query structure Q and dictionary structure D and test whether Q is a subgraph of D;

b. If the preceding test is negative use path matrix including indirect connections to determine whether Q is a completed partial graph of D;

c. If the preceding test is again negative, selectively remove branch labels by altering branch label matrix and test again using first only direct connections (subgraph test), then indirect connections;

d. If matching conditions must be further relaxed, replace unilateral by bilateral connections and use symmetric connection matrices first with direct and then with indirect connections;

e. Finally, selectively remove node labels and test again for subgraph and then for incomplete partial graph.

A retrieval system using graph matching procedures in conjunction with natural language data is outlined in the next section.

A SENTENCE MATCHING PROCEDURE FOR DOCUMENT RETRIEVAL

A simplified automatic document retrieval system is shown in Fig. 15. This system makes use of the standard statistical procedures, including the computation of word frequency counts, word associations based on co-occurrence in the same sentences or texts, document associations based on co-occurrence of words, and document relevance coefficients. In addition, a dictionary or thesaurus may be used if available to normalize the vocabulary.
The quantitative procedures may be supplemented by choosing a set of significant sentences, as determined by the statistical process, and using them to perform a structural analysis. Specifically, each word is furnished with one or more thesaurus category numbers (the semantic labels) as a result of the dictionary look-up procedure. If no dictionary is available, each word can of course function as a semantic label by itself. A syntactic analysis is then performed which determines a dependency structure for the words of a sentence, and also generates a syntactic label for each word. A typical dependency tree, resulting from an automatic syntactic analysis, is shown in Fig. 16. A syntactically analyzed sentence can of course be represented as before by direct and indirect connection matrices, as well as syntactic and semantic label matrices; moreover, these matrices can be generated automatically from the output furnished by the syntactic analysis program. 17

It is now possible to compare the set of analyzed sentences or search requests with a set of "criterion phrases" included in a phrase dictionary. Each criterion phrase is representative of one or more subject categories, and if a match is obtained between a criterion phrase and an analyzed sentence or search request, the corresponding subject categories can be attached to the matching sentences or requests. To retrieve a set of documents in answer to a given search request, it is then sufficient to compare the subject categories attached to the requests, with the subject identifiers attached to the documents as outlined in Fig. 15.

A typical criterion phrase is shown in Fig. 17. Each criterion phrase is represented, as before, by an identification number and control

![Figure 15. Simplified System Using Structural Matching.](From the collection of the Computer History Museum (www.computerhistory.org))

![Figure 16. Typical Syntactic Dependency Tree.](From the collection of the Computer History Museum (www.computerhistory.org))
information, the direct and indirect connection matrices, the syntactic and semantic node label matrices, and the category indicators which identify the subject classes for the given phrase. The semantic node labels attached to the sample criterion phrase are decoded in Fig. 17.

The matching process between a given criterion phrase and a sample sentence or search request is identical with that used in the preceding section for document graphs. That is, two principal criteria must be satisfied:

1. Given a specified node of the criterion phrase, all those sentence nodes are selected which have matching syntactic and semantic labels;

2. From among those sentence nodes which obey the restriction of part 1, some subset must be chosen whose direct (or indirect) connection pattern is identical with the connection pattern of the corresponding nodes in the criterion phrase.

Consider, as an example, the criterion phrase of Fig. 17 and the sentence of Fig. 16. Clearly, both syntactic and semantic labels of nodes ② and ⑤, and of nodes ③ and ⑦ will match properly. However, there exists a path from node ① to node ⑦ in the criterion phrase, while no such path exists from node ③ to node ⑦. Therefore, the structure matching procedure will not be successful for the given example. On the other hand, it can be easily verified that the trees of Fig. 18 will, in fact, properly match the criterion phrase of Fig. 17.

Several methods are provided in the system for adjusting the matching process. First, the matching algorithm itself is adaptable, since node and branch labels can or cannot be taken into account, and direct as well as indirect connections can be used. Second, it is possible to provide the criterion phrases with a smaller or larger number of syntactic and semantic labels,
thus restricting or enlarging the possible sentence nodes which are compatible. Finally, the thesaurus which can be used to replace text words by thesaurus categories, as well as the criterion phase dictionary can be enriched to ensure inclusion of a larger variety of possible sentence structures. The system is presently being tested in order to determine the practical effectiveness of these various measures.

REFERENCES


