# A SELF-ORGANIZING RETRIEVAL SYSTEM FOR GRAPHS

Robert Levinson
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

## ABSTRACT*

The design of a general knowledge base for labeled graphs is presented. The design involves a partial ordering of graphs represented as subsets of nodes of a universal graph. The knowledge base's capabilities of fast retrieval and self-organization are a result of its ability to recognize common patterns among its data items. The system is being used to support a knowledge base in Organic Chemistry.

## 1. Introduction

When asked to develop a retrieval system for known chemical reactions and molecules, we chose to undertake the more fundamental task of designing **a general knowledge base for labeled graphs**. In particular, we wished to have a system that would efficiently handle the query: Given a labeled-undirected graph Q and a data base of labeled undirected graphs answer the following:

1. Is Q a member of the data base? (exact match)
2. Which members of the data base contain Q as a subgraph? (supergraphs)
3. Which members of the data base contain Q as a supergraph? (subgraphs)
4. Which members of the data base have large subgraphs in the data base in common with Q? (close matches)

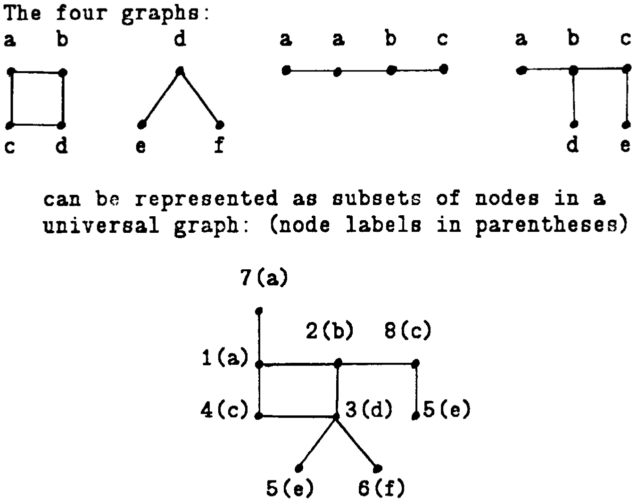
In this paper we discuss a system that meets the design objective mentioned above and also supports other features that are highly desirable in intelligent knowledge bases but are usually difficult to achieve. Most important of these features is the ability of the system to structure its own knowledge base through the recognition of common patterns (subgraphs) in its data items (graphs). In fact, the critical idea that our system demonstrates is that **the common patterns can be exploited for multiple purposes.** We call these common patterns *concepts*. They can be used to enhance retrieval efficiency, to increase the knowledge of the system (concept discovery), to characterize the relationships between its individual data items, and to provide criteria to select among partial and relaxed matches.

The system is currently being used successfully to support a knowledge base for Organic Chemistry. In further research we hope to demonstrate its utility in a variety of domains where individual data items can be represented as labeled graphs. Some of the domains being considered are VLSI designs, program trees, computer networks, structural diagrams, and semantic nets.

## 2. The data base design

The key to the data base design is the recognition that **(1) All of the graphs of the system can be viewed as subgraphs of a single "universal" graph.** and that **(2) these graphs can be represented by subsets of nodes of the universal graph.** The universal graph is constructed as new graphs are added to the system and it is used when old graphs are to be retrieved. For an example see Figure 1.

The four graphs:



can be represented as subsets of nodes in a universal graph: (node labels in parentheses)

The subsets are {1 2 3 4}, {3 5 6}, {1 2 7 8}, and {1 2 3 8 9}.

**Figure 1: An example of a universal graph**

The rest of the design involves making explicit the ordering achieved by the partial ordering relation *subgraph-of* (see Figure 2.) This is achieved by storing with each graph pointers to its immediate predecessors and to its immediate successors in the partial ordering. Each graph in the system is called a concept and describes a structure that is determined to be of interest. Initially the system has only the graphs (concepts) that represent complete facts in the problem and *primitives*. Primitives are the labels that appear on the nodes of the graphs. As concepts are added to the system they are inserted in their proper position in the partial ordering. Some of these new concepts may represent new complete objects and some may represent new primitives. But others may represent common substructures that are useful in

*This research is sponsored in part by the Robert A. Welch Foundation, a Cottrell Research Grant from Research Corporation, NSF grant MCS-8122039, and an NSF Graduate Fellowship.

analyzing complete objects. These intermediate concepts provide structure to the partial ordering, and thus aid the response rate and flexibility of the system. Another way to view the universal graph is that it is a graph that is pointed to by all graphs at the top of the partial ordering.

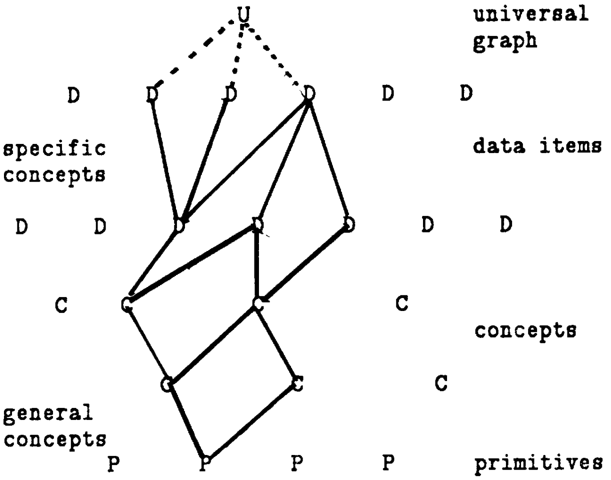


**Figure 2: The Partial Ordering of Concepts**

**(A typical chain is shown)**

## 3. The retrieval algorithm

In this section we describe an algorithm for answering parts 1-4 of the basic query given in the Introduction. The algorithm operates on the universal graph and partial ordering described in Section 1. In fact, the algorithm works on any set of data for which a partial ordering has been established. It is desirable to have an algorithm that minimizes the number of comparison operations that are necessary to answer the query for exact match, supergraphs, subgraphs and close matches. This minimization of comparison operations is particularly important in a system that uses complex objects like graphs since the complexity of these operations is usually exponential. The main way that our algorithm attempts to minimize the number of comparisons is by using the partial ordering to segment the data base so that only a small part of it need actually be considered in detail. The algorithm has the feature that it is easy to implement and that it searches nodes in a logical bottom-up order that may be useful in domains in which additional computation is desired during the retrieval process. (For example, we may wish to apply general concepts to a situation before more specific ones are found to be applicable.) On a data base of 200 concepts an average of about 25 node-by-node searches are required to answer a typical query. We are looking for other algorithms that might require an even smaller number of comparisons.

First we discuss the general algorithm for all partial orderings, and then we show how the universal graph representation can be used to further the efficiency of the algorithm for a data base of graphs.

The query can be answered by finding where the query structure should fit in the partial ordering, whether it is already in the ordering or not. Then Part 1 can be answered. Then parts 2-4 of the query can be answered by simply following pointers (chaining) in the partial ordering. We will see that most of the pointer chasing is already accomplished in the process of finding the immediate predecessors and successors of the query object in the partial ordering. We accomplish this in two phases:

Let *IP(y)* denote the set of immediate predecessors of the data element y. In **Phase 1** we determine IP(Q) where Q is the query object:

```
S := {}
While there is an unmarked element y of
the data base such that each member of IP(y) is
marked T or IP(y) = {} do
    If $y \preceq Q$ (* comparison needed *) then
        mark y as T
        S := [S - IP(y)] $\cup$ {y}
    Else
        mark y as F.
```

This process terminates with S = IP(Q).

Note that when Phase I begins, all objects at the bottom of the partial ordering are compared to Q since they have no immediate predecessors. This process can be accomplished quickly if we require that the bottom of the partial ordering contain real *primitives* (such as single nodes) for which the comparison operation is trivial.

An informal description of **Phase 2** shows what takes place: The goal of Phase 2 is to calculate *IS(Q)* - the immediate successors of Q:

```
Chain up from each member of IP(Q) in breadth
first or depth first fashion (the chaining from
the last member of IP(Q) must be breadth first)
When these upward chains meet (i.e. there is a
data item y on each of these chains) check if
$Q \preceq y$. If so, y is in IS(Q), else continue to
chain up from y.
```

Now let's go over how Phase 1 and Phase 2 help to answer parts 1-4 of the query:

1. Exact match: Q already exists in the data base, if IP(Q) = IS(Q). If so, then Q is the single element contained in these sets.
2. Subgraphs: The subgraphs are simply all nodes that were marked T in Phase 1.
3. Supergraphs: (This is the only place where additional chaining is required). The supergraphs are the union of the upward chains from each member of IS(Q)
4. Close matches: The close matches are the union of the upward chains from each member of IP(Q). In the most obvious implementation of Phase 2, a hash table is used to manage the breadth first search. It contains information about which nodes have been visited and which upward chains they are on. The desired union can be found simply by collecting elements of the hash table. (In our implementation of the graph system we use these nodes as candidates for comparison to Q, and we use an heuristic-based maximal common subgraph algorithm to extract larger close matches.)

How can the universal graph improve the efficiency of the algorithm when applied to a data base of graphs? Since we try to construct the universal graph to be as small as possible, many of its nodes will be shared by many graphs. This overlap and the fact that the graphs of the system are represented as sets means that **some exponential graph operations may be improved or they may be replaced by linear set operations.**

Where in the algorithm do these savings take place?

1. If we find a supergraph in Phase 2 we can use the location of Q in this supergraph to find a proper occurrence of graph Q in the universal graph itself. Now that Q has been reduced to a set, we can infer that all graphs that are represented by sets that are supersets of this set are supergraphs, without doing a node-by-node search. In practice, the universal graph helps to eliminate about 20% of the node-by-node searches required in phase 2.
2. If we know the placement of Q in the universal graph and we wish to determine common subgraphs often we can do this by taking intersections of Q's set with the sets representing other graphs.

### 4. The system applied to organic chemistry

In this section we show how the system is being used as a knowledge base for organic chemistry. The chemical data base represents chemical reactions reported in the chemical literature as labeled graphs. Primitives (see Section 2) are written in the form "X-Ylr" meaning that atoms X and Y are connected with bond type l on the left-hand-side of the reaction and bond r on the right-hand-side of the reaction. (The atom names represented by X and Y are in lexicographic order.) If a molecule is being represented, l will equal r. For example, C-C21 represents a double-bond between two carbon(C) atoms that is changed to a single bond. C-O02 likewise represents a newly created double bond between carbon(C) and oxygen(O). Finally, these labels are given concept numbers, since all primitives must also be concepts. Complete molecules and reactions also become concepts. In addition, intermediate concepts such as the functional groups arene and ester are added (either by hand or by the system) to provide additional structure to the partial ordering. The data base currently has about 600 concepts for complete structures, 50 intermediate concepts, and 100 primitives. Soon 500 reactions with the associated molecules will be added. Preliminary experiments confirm that the use of well-chosen intermediate concepts to structure the partial ordering does in fact significantly limit the number of graphs that must be examined to answer a query.

To be useful, the chemistry data base (or most any other data base, for that matter) must contain more than just graph structures. Other knowledge is associated with each structure. For example, each reaction concept has associated with it, pointers to the two graphs representing the left-hand-side and right-hand-side of the reaction. This association allows us to view reactions as graph-to-graph production rules. Further information about the chemical reactions such as yields, reaction conditions and literature references are stored in auxiliary files that are associated with the standard graph system. This will help the system to serve as an aid to the organic chemist who is trying to synthesize an organic molecule.

The major difference between our system and other chemical substructure search systems is our ability to organize *multi-levels* of search screens *dynamically*. See also (Adamson, 1973), (Bawden, 1983), (Dittmar, 1983), (Feldman and Hodes, 1975), (Fugmann, 1979), (O'Korn, 1977) and (Willett, 1980). The major difference between our system and other systems designed to do organic synthesis is our ability to organize and employ a large body of *real world data*. Another important difference is the explanatory power that can be gleaned from the generalization arcs in the partial ordering. See also (Gelertner, 1973), (Sridharan, 1973), (Willett, 1980) and (Wipke, 1977)

### 5. The system applied to organic chemistry

In this section we discuss features that in addition to the retrieval capacity make the graph system a useful design for an Artificial Intelligence knowledge base. Examples are taken from the Organic Chemistry application.

- **The universal graph is an efficient way to store a large number of graphs** since adjacency lists need only be stored once as part of the universal graph.
- **The universal graph and the partial ordering are excellent aids to concept discovery.** We have seen that in the Organic Chemistry domain useful concepts to the chemist can be found by finding common subgraphs among the elements in the data base. These common subgraphs may be recognized as overlaps of sets of nodes in the universal graph. For instance the set {1,2,8} in Figure 1. By finding places in the partial ordering where further differentiation among concepts is required we can see where additional graphs should be added. See Figure 3. The added concepts make the ordering more balanced. These local techniques are important on large data bases where global statistical techniques like cluster analysis are computationally infeasible. Examples of common chemical structures discovered by our system include the functional groups arene, ether, phenol, and carboxylic acid as well as some useful generalizations of real-world reactions.

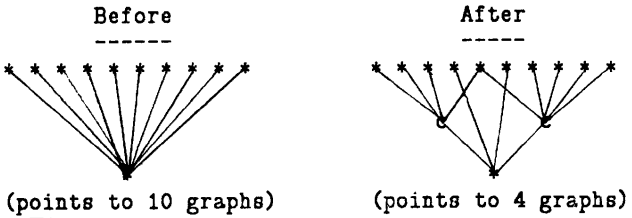


**Figure 3: Adding concepts**

- **The partial ordering represents a useful characterization of the data in the data base.** As we move down the partial ordering we move to concepts of greater and greater generality. Likewise, as we move up, we move to more and more specific concepts. A theory of such *generalization hierarchies* is given in (Sowa, 1983). An important feature of our system is its ability to derive generalizations from its reaction data base. These generalizations become important when they are applied to suggesting precursors to a molecule not yet known by the data base. Another unique feature of the Organic Chemistry domain is that generalizations can be written down simply as substructures of larger graphs. See Figure 4.

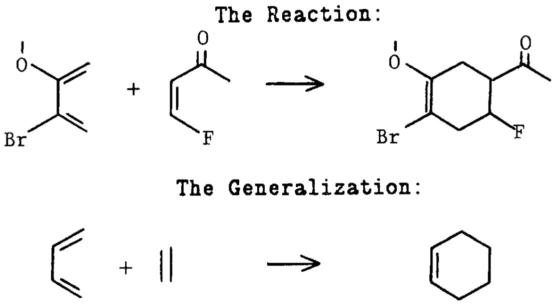


**Figure 4: A Reaction and Its Generalization**

- **Retrieval is fastest for graphs that already exist or are quite similar to stored graphs**. This allows the system, by storing query structures and finding common patterns with them, to adapt its retrieval capabilities to the needs of an individual user who may often ask queries that are similar or identical to previous queries.
- **The system has the capability for relaxed-matching.** This is made possible by allowing an individual label in the query structure to match any of a set of labels. The retrieval algorithm no longer works the same as before since the partial ordering does not contain the pointers associated with the "relaxed" structures. Subgraphs are discovered as before but often we must wait until the close match stage to determine the supergraphs. This is because IP(Q) for a "relaxed" Q usually contains more elements than IP(Q) otherwise. An example of relaxed matching in the chemistry domain is allowing the primitives C-CL11 and C-BR11 and C-F11 to be equivalent since halogens (Cl, Br, and F) often function similarly. These equivalence classes currently must be defined by the user who has the option of invoking one or more of them at query initiation time. We are exploring whether the system can discover some of these equivalence classes on its own.

## 6. Conclusion

The capabilities of our system may seem at first glance to be surprising when taken with the result that weak, syntactic methods are usually not enough to support intelligent behavior. However, there seems to be a more powerful principle at work here: *An ideal representation is one that has a form analogous to what it represents.* We exploit this principle twice:

1. We use chemical structural diagrams. These are known to be useful analogies of the real world.
2. The universal graph and partial ordering make explicit the relationships between individual data items. Graphs that have much in common are physically and logically close together.

This principle is not new. For example, Doug Lenat cites this principle as the major reason for success of his AM program and Gelertner's geometry theorem prover (Lenat and Brown, 1983) and (Gelertner, 1963). The Handbook of Artificial Intelligence (Barr, 1981) calls such *ideal* representations *direct* or *analogical representations.* Recently, (Pentland and Fischler, 1983) called these representations *isomorphic representations* .

Without a good deal of commonality between the individual data items, the power acquired from the second application of the principle would be lost. However, a data base of more or less unrelated data items probably would not be useful for complex reasoning.

## ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Elaine Rich and my collaborator Dr. Craig Wilcox (Dept. of Chemistry) for their support and many contributions to this research. I also would like to thank James Wells for the graphics programs, and Mohan Ahuja for his encouragement.

## REFERENCES

**1.** Adamson, G. W. , Cowell, J. , Lynch, M. F. , McLure, H. W. , Town, W. G. and Yapp, M. A . "Strategic Considerations in the Design of a Screening System for Substructure Searches of Chemical Structure Files." *Journal of Chemical Documentation 13* (1973), 153-157.

**2.** Barr, A. and Feigenbaum, E. A. *The Handbook of Artificial Intelligence.* Kaufman, Los Altos, Calif. , 1981.

**3.** Bawden, D. . "Computerized Chemical Structure-Handling Techniques in Structure-Activity Studies and Molecular Property Prediction." *Journal of Chemical Information and Computer Sciences 23* (Feb 1983), 14-22.

**4.** Dittmar, P. G. , Farmer, N. A. , Fisanick, W. , Haines, R. C. , Mockus, J . "The CAS ONLINE Search System 1. General System Design and Selection, Generation, and Use of Search Screens." *Journal of Chemical Information and Computer Sciences 23* (Aug 1983), 93-102.

**5.** Feldman, A. and Hodes, L . "An Efficient Design for Chemical Structure Searching I, The Screens." *Journal of Chemical Information and Computer Sciences 15* (1975), 147-151.

**6.** Fugmann, R. , Kusemann, G. , and Winter, J. H . "The Supply of Information on Chemical Reactions in the IDC System." *Information Processing and Management 15* (1979), 303-323.

**7.** Gelertner, H . Realization of Geometry Theorem Proving Machine. In *Computers and Thought,* Feigenbaum and Feldman, Eds., Mcgraw-Hill, 1963, pp. 134-152.

**8.** Gelertner, H . "The Discovery of Organic Synthetic Routes by Computer." *Topics in Current Chemistry 41* (1973).

**9.** Hayes-Roth, F. , Waterman, D. , and Lenat, D. B . *Building Expert Systems.* Addison-Wesley, 1983.

**10.** Lenat, D. B. and Brown, J. S . Why AM and Eurisko Appear to Work. Proc. AAAI-83, 1983.

**11.** O'Korn, L. J . Algorithms in Computer Handling of Chemical Information. In *Algorithms for Chemical Computations,* Christoffersen, R. E. , Ed.,American Chemical Society, 1977, pp. 122-148.

**12.** Pentland, A. P. , Fischler, M. A . "A More Rational View of Logic." *AI Magazine 4,* 4 (1983).

**13.** Sowa, J. F. . *Conceptual Structures: Information Processing in Mind and Machine.* Addison-Wesley, 1983.

**14.** Sridharan, N. S. . Search Strategies for the Task of Chemical Organic Synthesis. Proc. IJCAI-3, 1973.

**15.** Willett, P . "The Evaluation of an Automatically Indexed, Machine-Readable Chemical Reactions File." *Journal of Chemical Information and Computer Sciences 20* (1980), 93-96.

**16.** Wipke, W. T. and Howe, W. J.(editors) . *Computer-Assisted Organic Synthesis.* American Chemical Society, 1977.