STANFORD ARTIFICIAL INTELLIGENCE PROJECT
MEMO AIM-145

COMPUTER SCIENCE DEPARTMENT
REPORT NO. CS-221

A HEURISTIC PROGRAMMING STUDY OF THEORY FORMATION IN SCIENCE

BY

BRUCE G. BUCHANAN

EDWARD A. FEIGENBAUM

JOSHUA LEDERBERG

JUNE 1971

COMPUTER SCIENCE DEPARTMENT

STANFORD UNIVERSITY

STANFORD ARTIFICIAL INTELLIGENCE PROJECT
MEMO AIM-145

JUNE 1971

COMPUTER SCIENCE DEPARTMENT REPORT
NO. CS 221

A HEURISTIC PROGRAMMING STUDY OF THEORY FORMATION IN SCIENCE*

by

Bruce G. Buchanan
Edward A. Feigenbaum
Joshua Lederberg

ABSTRACT: The Meta-DENDRAL program is a vehicle for studying problems of theory formation in science. The general strategy of Meta-DENDRAL is to reason from data to plausible generalizations and then to organize the generalizations into a unified theory. Three main subproblems are discussed: (1) explain the experimental data for each individual chemical structure, (2) generalize the results from each structure to all structures, and (3) organize the generalizations into a unified theory. The program is built upon the concepts and programmed routines already available in the Heuristic DENDRAL performance program, but goes beyond the performance program in attempting to formulate the theory which the performance program will use.

*This research was supported by the Advanced Research Projects Agency of the Office of the Secretary of Defense under Contract SD-183.

A Heuristic Programming Study of Theory Formation in Science

by

Bruce G. Buchanan

Edward A. Feigenbaum

Joshua Lederberg

I. INTRODUCTION

Theory formation in science embodies many elements of creativity which make it both an interesting and challenging task for artificial intelligence research. One of the goals of the Heuristic DENDRAL project has long been the study of processes underlying theory formation. This paper presents the first steps we are taking to achieve that goal, in a program called Meta-DENDRAL.

Because we believe there is value in reproting ideas in their formative stages -- in terms of feedback to us and, hopefully, stimulation of the thinking of others -- we are presenting here a description of work on Meta-DENDRAL even though not all of the program has been written. Just like the scientists we attempt to model, we often fail to make explicity the thinking steps we go through. Therefore, the designs of the unfinished peices of program are described as they will be initially programmed, and several outstanding problems are mentioned. It is hoped that this discussion will provoke comments and criticisms, for that is also part of its purpose.

The Heuristic DENDRAL project has concentrated its efforts on the inductive analysis of empirical data for the formation of explanatory hypotheses. This is the type of inference task that calls for the use of a scientific theory by a performance program, but not for the formation of that theory. When we started on Heuristic DENDRAL we did not have the insight, understanding, and daring to tackle ab initio the problem of theory formation. But now we feel the time is ripe for us to turn our attention to the problem of theory formation. Our understanding and our technical tools have matured along with the Heuristic DENDRAL program to the point where we now see clear ways to proceed.

As always, the proper choice of task environment is crucial, but for us the choice was absolutely clear. Because the Heuristic DENDRAL performance program uses the theory of a specialized branch of chemistry, formulating statements of that theory is the task most accessible to us. The theory itself will be briefly introduced in Section II, although it is not expected that readers understand it to understand the directions of this paper.

The goal of the Meta-DENDRAL program is to infer the theory that the performance program (Heuristic DENDRAL) uses to analyze experimental chemical data from a mass spectrometer. The follcwing table attempts to sketch some differences between the programs at the performance level and the meta-level.

	Heuristic DENDRAL	Meta-DENDRAL
Input	The analytic data from a molecule whose structure is not known (except of course in our test cases).	A large number of sets of data and the associated (known) molecular structures.
Output	A molecular structure inferred from the data.	A set of cleavage and rearrangement rules constituting a subset of the theory of mass spectrometry.
Example	Uses alpha-carbon fragmentation theory rules in planning and in validation.	Discovers (and validates) alpha-carbon fragmentation rules in a space of possible patterns of cleavage. Uses set of primitive concepts but does not invent new primitives.

In our view, the continuity evident in this table reflects a continuity in the processes of inductive explanation in science. Moves toward meta-levels of scientific inference are moves toward encompassing broader data bases and constructing more general rules for describing regularities in the data.

Beyond this level of Meta-DENDRAL there are still higher levels. Not all theory formation is as simple as the program described here assumes it is. For example, the representation of chemical molecules and the list of basic processes are both fixed for this program, yet these are concepts which a higher level program should be expected to discover. Also, there is no postulation of new theoretical entities in this program. But, again, higher levels of theory formation certainly do include this process.

The task of theory formation can be and has been discussed out of the context of any particular theory.<4> However, writing a computer program to perform the general task is more difficult than working within the context of one particular scientific discipline. While it is not clear how science proceeds in general, it may be possible to describe in detail how the scientists in one particular discipline perform their work. From there, it is not a large step to designing the computer program. Thus this paper attacks the general problems of theory formation by discussing the problems of designing a computer program to formulate a theory in a specific branch of science<cf. 2>.

The general strategy of Meta-DENDRAL is to reason from data to plausible generalizations and then to integrate the generalizations into a unified theory. The input to the Meta-DENDRAL system is a set of structure-data pairs. It receives essentially the same data as a chemist might choose when he attempts to elucidate the processes underlying the behavior of a class of molecules in a mass spectrometer. When chemists turn their attention to a class of chemical compounds whose mass spectrometric processes (MS processes) are not well understood, they must collect mass spectrometry data for a number of the compounds and look for generalizations. The generalizations have to be tested against new data and against the established theory.

If new data provide counterexamples, the generalizations are changed. If the generalizations are not compatible with the old theory either the old theory or the generalizations are changed.

This paper is organized by the three main subproblems around which the program is also organized. The first is to explain the experimental data of each individual molecular structure. That is, determine the processes (or alternative sets of processes) which account for the experimental data. The second subproblem is to generalize the results from each structure to all structures. In other words, find the common processes and sets of processes which can explain several sets of experimental data. The last is to integrate the generalizations into the existing theory in such a way that the theory is consistent and economical. Within each of the three main sections, the subsections indicate further subproblems which the program must solve.

II. THE PROBLEM DOMAIN

Because this paper discusses theory formation in the context of a particular branch of science, mass spectrometry, the theory of this science will be explained briefly for readers wishing an understanding of the Meta-DENDRAL program at this level.

The mass spectrometer is an analytic instrument which bombards molecules of a chemical sample with electrons and records the relative numbers of resulting charged fragments by mass. When molecules are bombarded, they tend to fragment at different locations and fragments tend to rearrange and break apart as determined by the environments around the critical chemical bonds and atoms. The description of these processes is called "mass spectrometry theory". The output of the instrument, the mass spectrum or fragment-mass table (FMT)*, is commonly represented as a graph of masses of fragments plotted against their relative abundance. By examining the FMT, an analytic chemist often can determine the molecular structure of the sample uniquely.

----

*The term 'fragment-mass table' is used here in place of the slightly misleading term 'mass spectrum'. The latter is well entrenched in the literature, but the former is more suggestive of the form of the data.

Mass spectrometry theory (MS theory), as used by the DENDRAL programs and many chemists, is a collection of statements about the fragmentation patterns of various types of molecules upon electron impact. It contains, for example, numerous statements about the likelihood that links (bonds) between chemical atoms will break apart or remain stable, in light of the local environment of the bonds within the graph structure of the molecule. The probability of a fragment splitting off the molecule is determined by the configurations of chemical atoms and bonds in the fragment and in its complement. Further splitting of the fragment is determined in like manner. In addition to rules about fragmentations, the theory also contains rules relating graph features of molecules and fragments to the probabilities that an atom or group of atoms will migrate from one part of the graph to another. Fortunately, mass spectrometry results are reproducible, or nearly so, which means that identical samples will produce nearly identical FMTs (under the same operating conditions of the same type of instrument).

As mentioned earlier, there are alternative levels for expressing this, as any other theory. The model in whose terms the theory is stated is a "ball and stick" model of chemistry, in which 'atom' and 'bond' are primary terms, and not, for example, an electron density model. Some of the primitive terms of the program's theory are listed in Appendix A.

III. FIRST SUBPROBLEM: EXPLAINING EACH SPECTRUM

The so-called "method of hypothesis" in science is sometimes proposed as the essence of scientific work. Restating it, in a deliberately imprecise way, the method is to formulate a hypothesis to account for some of the observed data and make successively finer adjustments to it as more observations are made. Very little is known about the details of a scientist's intellectual processes as he goes through the method. Thinking of hypotheses, for example, is a mysterious task which must be elucidated before the method can be programmed. That is the task we have designated as the first subproblem.

The program starts with individual structure- FMT (fragment-mass table) pairs as separate from one another. It constructs alternative explanations for each FMT and then considers the FMT's all together. An explanation, for the program, as for the chemist, is a plausible account of the MS processes (or mechanisms) which produced the masses in the FMT. The explanation is something like a story of the molecule's adventures in the mass spectrometer: certain data points appear as a result of cleavage, others appear as a result of more complex processes. At this stage of development of the theory, the chemist's story does not account for every data point because of the complexities of the instrument and the vast amount of missing information about MS theory.

A. REPRESENTATION

The well-known problem of choosing a representation for the statements of a scientific theory and the objects mentioned by the theory is common to all sciences. In computer science it is recognized as a crucial problem for the efficient solution (or for any solution) to each problem. Some ways of looking at a problem turn out to be much less helpful than others, as, for example, considering the mutilated checkerboard problem<5> as simply a problem of covering rectangles (with dominoes) instead of as a parity problem. At this stage there are no computer programs which successfully choose the representation of objects in a problem domain. Therefore we, the designers of the Meta-DENDRAL system, have chosen representations with which we have some experience and for which programmed subroutines have already been written in the Heuristic DENDRAL performance system.

It was natural to use these representations since the meta-program itself will not only interface with the Heuristic DENDRAL performance program, but is built up from many of the LISP functions of the performance program. Specifically, for this program, the input data are chemical structures paired with their experimental data:

```
structure-1  -  FMT-1
...             ...
structure-n  -  FMT-n
```

The representation of chemical structures is just the DENDRAL representation used in the Heuristic DENDRAL system. It has been described in detail elsewhere <see 1>: essentially it is a linear string which uniquely encodes the graph structure of the molecule.

The FMTs, also, are represented in the same way as for the Heuristic DENDRAL performance system. Each FMT is a list of x-y pairs, where the x-points are masses of fragments and the y-points are the relative abundances of fragments of those masses.

The Predictor program of the Heuristic DENDRAL system has been extensively revised so that the internal representations of molecular structures and of MS theory statements would be amenable to the kind of analysis and change suggested in this work. As mentioned, Appendix A contains examples of the terms which are used in statements of the theory.

B. SEARCH

It is not clear what a scientist does when he "casts about" for a good hypothesis. Intuition, genius, insight, creativity and other faculties have been invoked to explain how a scientist arrives at the hypothesis which he later rejects or comes to believe or modifies in light of new observations. From an information processing point of view it makes sense to view the hypothesis formation problem as a problem of searching a space of possible hypotheses for the most plausible ones. This presupposes a generator of the search space which, admittedly, remains undiscovered for most scientific problems.

In the Heuristic DENDRAL performance system the "legal move generator" is the DENDRAL algorithm for constructing a complete and irredundant set of molecular models from any specified collection of chemical atoms. Heuristic search through this space produces the molecular structures which are plausible explanations of the data. The meta-problem of finding sets of MS processes to explain each set of data is also conceived as a heuristic search problem. Writing a computer program which solves a scientific reasoning problem is facilitated by seeing the problem as one of heuristic search. This is as true of the meta-program which reasons from collections of data to generalizations as for the performance system which reasons from one set of data to an explanation. For this reason we have called the process of induction "a process of efficient selection from the domain of all possible structures."<3>

In broad terms, the program contains (1) a generator of the search space, (2) heuristics for pruning the tree, and (3) evaluation criteria for guiding the search. Except for problems inherent in the task, then, the problems of such a program are reasonably well understood. These three main components of the heuristic search program are considered one at a time in the immediate discussion.

1. GENERATOR

For this part of the Meta-DENDRAL system, the generator is a procedure for systematically breaking apart chemical molecules to represent all possible MS processes. In addition to single cleavages, the generator must be capable of producing all possible pairs of cleavages, all possible triples, and so forth. And, for each cleavage or set of cleavages it must be able to reproduce the result of atoms or groups of atoms migrating from one fragment to another. For example, after the single break labeled (a) in Figure 1 below, subsequent cleavage (b) may also occur. The result of (a) + (b) is the simple fragment CH3.

```
               O

CH3  -  C  -  CH2  -  CH2  - CH3
     (b)   (a)
```

FIGURE 1

Or, for the same molecule, cleavage (c) may be followed by migration of one hydrogen atom from the gamma position (marked with an asterisk) to the oxygen, as shown in Figure 2:

```
               O

CH3  -  C  -  CH2  -  CH2  -  CH3
                   (c)         *
```

FIGURE 2

The generator of the search space will postulate these processes as possible explanations of the FMT data points at masses 15 ($CH3$) and 58 ($C3H6O$) for this particular molecule. But it will also postulate the simple cleavage (b) in Figure 1 as the explanation of the peak at mass 15. And for the peak at mass 58 from the process in Figure 2 it will postulate the alternative migration of a hydrogen atom from the beta position (adjacent to the asterisk). From the generator's point of view these processes are at least as good as the more or less accurate processes shown in Figures 1 and 2.

Chemists also appeal to the localization of the positive charge in the charged molecule to explain why one peak appears in a set of data but another does not. Since it is known that only the charged fragments are recorded by the mass spectrometer, the generator program must also manipulate charges to account for the data.

The primitive mechanisms of the generator are charge localization, cleavage, and group migration (where a group can be a positive charge, a single atom, or a set of connected atoms). The generator is a procedure for producing all possible charged fragments, not just all possible fragments, in other words. Putting these mechanisms together in all possible ways leads to an extremely large space of possible explanations for the peaks in the experimental FMT of a molecule. The pruning heuristics discussed in the next section alleviate that problem. Briefly, let us turn to the actual design of the generator.

At the first level of branching in the tree all possible single cleavages are performed on the original molecular structure resulting in all possible primary fragments. At the next level, the positive charge is assigned to all possible atoms in the fragments. (Switching these two steps gives the same results and is closer to the conceptualization used by the chemist; it results in a less efficient program, however.) Starting with level 3, the procedure for generating successive levels is recursive: For each charged fragment at level n $(n > 2)$ produce the charged fragments resulting from (i) cleavage of each bond in the fragment and (ii) migration of each group from its origin to each other atom in the fragment, where 'group' currently means 'positive charge or hydrogen atom'.

2. PRUNING HEURISTICS

Three simple pruning techniques are currently used by the program. (1) Since the result of breaking a pair of bonds (or n bonds) is independent of the order in which the bonds are broken, allow only one occurrence of each bond set; (2) Since MS processes tend to follow favorable pathways, prune any branch in the tree which is no longer favorable, as evidenced by failure of a fragment's mass to appear in the experimental FMT; (3) Limit the number of allowable group migrations after each cleavage.

The first pruning technique hardly needs explaining: duplications of nodes in the search space are unnecessary in this case and can be avoided by removing a bond from consideration after all possible results of breaking it have been explored. The second technique carries an element of risk, because mass spectrometry theory includes no guarantee that every fragment in a decomposition pathway will produce a peak in the experimental FMT. In fact, the pruning can only be done after a complete cycle of cleavage plus migration because these processes occur together in the mass spectrometer -- without the appearance of the intermediate fragments. The third technique also is truly heuristic since there are no theoretical reasons why group migrations might not occur in complex and exotic patterns between cleavages. The bias of mass spectroscopists toward simple mechanisms, however, leads us to believe that they would place little faith in exotic mechanisms as explanations of peaks in the data, at least not without other corroborating evidence.

3. EVALUATION

Evaluation of alternative paths in the search tree is necessary, either during generation or after it is completed, in order to distinguish the highly attractive explanatory mechanisms from those which are merely possible. However, without building in the biases of experts toward their current theory it is difficult to evaluate mechanisms at all.

The program's evaluation routine presently contains only one a priori principle, a form of Occam's razor. In an attempt to measure the simplicity of the statements describing mechanisms, the program counts the number of primitive mechanisms necessary to explain a peak. Thus when there are alternative explanations of the same data point, the program chooses the simplest one, that is, the one with the fewest steps. Simple cleavage is preferred to cleavage plus migration plus cleavage, for example.

The result of the generation process as described so far, with pruning and evaluation, is a set of candidate MS processes for each structure which provides alternative explanations for data points in the associated mass table (FMT). For instance, the program breaks the molecular structure shown in Figure 3 at individual bonds or pairs of bonds to give the following information (atoms in the structure are numbered from left to right):

MASS EXPLAINED	PROCESS
103	Breakbond: C2-C1 or Breakbond: C6-C7
89	Breakbond: S3-C2

	or Breakbond: C5-C6
75	Breakbond: C4-C5
61	Breakbond: S3-C4
60	Breakbond: C4-C5 & C2-C1
57	Breakbond: C4-S3
46	Breakbond: S3-C4 & C2-C1
43	Breakbond: C5-C4
42	Breakbond: C6-C7 & C4-S3
29	Breakbond: C2-S3
28	Breakbond: C5-C6 & C4-S3

```
 1      2      3     4      5      6      7
CH3 - CH2 - SH - CH2 - CH2 - CH2 - CH3
```

FIGURE 3

In this example, the program used no migrations or charge localization information, for purposes of simplicity. The program explcred all simple cleavages and found peaks corresponding to every resulting fragment but two.* For each of the successful fragments, the program broke each of the remaining bonds. From all the secondary breaks considered, the resulting fragments corresponded to only four additional peaks in the FMT. So these four branches of the search tree were each expanded by one more simple cleavage. None of the tertiary fragments were found in the FMT sc the program terminated.

----

*The CH3 fragment was produced twice but peaks of low masses were not recorded in the FMT.

The output of this phase of the program is a set of molecule-process pairs. For the one example shown in Figure 3, thirteen such pairs would be included in the output: the molecule shown there paired with each of the thirteen processes.

IV. SECOND SUBPROBLEM: GENERALIZING TO ALL STRUCTURES

The method of hypothesis, mentioned earlier as a vague description of scientific work, suggests that a plausible hypothesis can be successively modified in light of new experience to bring a scientist closer and closer to satisfactory explanations of data. Apart from the problem of formulating a starting hypothesis discussed above and the problem of terminating the procedure, it is not at all clear how the adjustments are to be made nor how to select the new experiences so as to make the procedure relatively efficient, or at least workable. These are well-known problems in the methodology of science. In other terms, the problem of successive modifications can be viewed as a problem of generalizing a hypothesis from one set of observations to a larger set.

The task for the second main part of the Meta-DENDRAL system is to construct a consistent and simple set of situation-action (S-A) rules out of the numerous instances of rules generated by the first phase of the system. It is necessary for this program to determine (a) when two instances (molecule-process pairs) are instances of the same general S-A rule and (b) the form of the general rule. In other terms, the program is given a set of input/output (I/O) pairs, with respect to the MS theory in a "black box". The task of the program is to construct a model of what is inside the black box. Thus it needs methods for (a) determining when two outputs (processes) are of the same class and (b) constructing an input/output transformation rule which accounts for the inputs (molecules) as well as outputs.

For each molecule there will be several associated processes, as seen from the example from Section III (Figure 3). So the same molecule will appear in several I/O pairs. Moreover, since the molecules are chosen for the test because they are known to exhibit similar MS behavior, there will be a number of instances of each general MS rule. If the program is successful, the resulting set of explanations will be a unified description of the MS behavior of all the molecules in the class. In operational terms, this means, at least, that the final set of explanations will be smaller than the union of instances.

The program itself has not been completed. It is hoped that this sketch shows enough detail that it will be instructive and provocative. Yet we do not wish to emphasize unfinished pieces of programs.

As in Section III, the issues of representation and search are discussed separately in this section.

A. REPRESENTATION

The general form of the rules the program is to infer has been fixed as S-A rules, as mentioned above. But representing the instances from which to infer the rules presents other difficulties. It has been difficult to decide how to represent the instances in such a way that they can be compared and unified, without building in concepts which would beg the theory-formation question. For example, representing the chemical graphs by feature vectors is attractive because it is easy to give the program just the right information for efficient comparisons. But this is the danger, too, for omitting "superfluous" information gives the program much too great a head start on the problem. It might discover what we believe are in the data -- the old principles-- but it would never discover anything new.

The difficulty with the representation of the instances, i.e., the molecule-process pairs in the input stream, is that the numbering of atoms in the molecules, and the corresponding numberings in the function arguments of the processes, do not allow simple comparisions. However, by comparing rules two at a time it is possible to determine mappings between the atoms, and the function arguments, so that the program can make comparisons. This is described below, as part of the scheme for generalizing rules.

B. SEARCH

The program has been designed to generalize on situations which exhibit the same processes. If situations M1 and M2 both exhibit process P, for example, the program attempts to construct a rule (S --> P) where S captures the common features of M1 and M2. This procedure requires that the program knows enough about the syntax of the process language that it can recognize the "same" process in different contexts. Also, this procedure requires that the program can find common features of situations which satisfy some criteria of non-triviality.

As in any learning problem there will need to be many readjustments of the learned generalizations as new data are considered. In this case, the addition of each new molecule-process pair brings the potential for revising any S-A rule in the emerging MS theory. Since each molecule initially considered may be associated with a dozen or more processes, and the emerging theory may contain many dozens of S-A rules, the generalization process will be lengthy.

All of the molecule-process pairs, which are instances of the rules the program is supposed to find, are compared among themselves. The result of this comparison is a set of generalized descriptions which account for the input data. This resulting set is then organized hierarchically to form the program's MS theory by the process described in Section V.

The comparison of the instances is conducted pairwise. The first molecule-process pair is postulated as a situation-action rule, R. A new molecule-process rule, N, (the next one) is then compared with R in the following way. (1) The MS processes, or actions, of N and R are compared at a gross level. (2) If this comparison holds, the graph structures (situations) of N and R are compared to find common subgraphs. If there are no common subgraphs, N is compared with the next rule, or, if no more rules, N is postulated as a new rule. (3) Otherwise, the common subgraph, S, is expanded to S' to capture alternative allowable atoms beyond the common subgraph as indicated by the situations of N and R. (4) Finally, the graph of R is replaced by S'. These four steps will be illustrated and briefly described below.

Consider the rule

1 2 3 4 5 6

```
(R):   CH3 - CH2 - NH - CH2 - CH2 - CH3   -->  Breakbond(4 5)
```

and the new molecule-process pair

```
        1     2    3     4     5     6
(N):   CH3 - NH - CH2 - CH2 - CH2 - CH3   -->  Breakbond(3 4).
```

(1) Compare the processes of R and N (the right-hand sides of R and N), disregarding the arguments of functions. Comparison of just the names of the processes shows that both R and N follow the same syntactic rules, and thus deserve closer comparison. This is made possible by the generator of processes described in Section III, which names processes and sets of processes uniquely. Had the form of the processes been different, N would be compared with the next rule (if any).

(2) Compare the graph structures in N and R, ignoring hydrogen atoms (H) for the moment. Using the clue that the atoms involved in the processes of both N and R are important, the program looks for a way of matching these atoms. Then the "interesting" subgraphs in both N and R are expanded, starting with the important atoms and building the greatest subgraph, S, which is common to both N and R. The criteria of "interesting" subgraphs and for "greatest" common subgraph are heuristic and are specific to chemistry.

Since the nitrogen atom, N, and the adjacent right-hand carbon atom, C, are both involved in the Breakbond process for both rules (N) and (R), these are recognized as important atoms. Thus the subgraph common to (R) and (N) must contain these nodes. Using the numbering of the graph of (R), nodes 2-6 are found to be common to both graphs. This is an "interesting" subgraph because, for example, it contains at least one non-carbon atom and contains more than two nodes. Moreover, it is the greatest subgraph common to the two. Without H's, this subgraph, S, is:

```
        2   3   4   5   6
 (S):   C - N - C - C - C
```

(3) Expand the subgraph (S). Now, reconsider the hydrogen atoms ignored in step (2). Nodes 2 and 6 in S fail to match exactly on the number of hydrogens, but the rest do match. Both 2 and 6 are connected to at least two hydrogens, but in each case, the last connection may be to either an H or a C. This is reflected in the expanded subgraph

```
 (S'):  (C,H) - CH2 - NH - CH2 - CH2 - CH2 - (C,H)
```

The parentheses indicate alternative choices for the atom linked by the adjacent bond.

The program now extends subgraphs only one atom beyond the greatest common subgraph (in each direction), but this clearly should be a parameter which the system can set.

(4) Replace the graph of R with S'. The result of comparing N with R, then, has been to change the conditions under which the process of R has been observed to apply. The old rule R is replaced by a revised rule, R', in which the situation is modified, but the action remains the same:

```
               1      2      3     4      5      6       7
  (R'):   (C,H)  - CH2  - NH  - CH2  - CH2  - CH2  - (C,H)   -->
Breakbond(4 5)
```

The result of this whole process is a set of S-A rules which can account for the observed data. This part of the program cautiously tries not to generalize beyond the observed situations. So it may miss some sweeping generalizations ("brilliant insights") which explain several of these cautious rules. But its result will not, at least, contain n "rules" to explain n observations, unless the input data are wildly discrepant.

V. THIRD SUBPROBLEM: ORGANIZING NEW RULES AND INTEGRATING THEM INTO THE EXISTING THEORY

The scientist's problem does not necessarily end with the satisfactory formulation of general statements explaining all the observed data. If he is working in a discipline for which there is no existing theory, he will still want to organize the statements. But it is rare to be out of any theoretical context. Typically, the hypotheses are formulated as extensions of some existing theory. Thus, the Meta-DENDRAL program must be prepared to merge new MS rules into the theory previously constructed by the program (or by a chemist). However, as a test exercise we want to see whether the meta-program builds approximately the same MS theory as the performance program now contains.

One cf the reasons we have rewritten the DENDRAL system's mass table predictor was to separate the MS theory from the LISP functions it drives. Making changes to the theory, then, does not require reprogramming, in the usual sense. Consequently, writing a program which updates the theory no longer seems to be an insurmountable task.

The problems of organizing a set of new rules or integrating new rules into the old theory are independent of the source of those rules. In order to study these problems we have written a program which (a) accepts new rules from human chemists and (b) updates the theory table of the program. The program for doing (a), called the dialog program, is not central to this paper, thus this section will focus on the work to accomplish (b), organizing and updating the theory.

In short, the program organizes the new rules either into a fresh theory or into an old theory (depending on the test) in the same way. The rule table is organized hierarchically according to the situations in the rules. Because the situations are graph structures, determining situation levels is just determining whether one graph is contained within another. For example, the graph $-NH_2$ is contained in the graph $-CH_2-NH_2$, so the former is a higher-level situation in the rule table. If neither situation is a subgraph of the other and they are not identical, they are put at the same level in the rule table.

A. REPRESENTATION

The performance program's MS theory is represented as a table of situation-action rules (S-A rules), patterned after Waterman's table of heuristics for good poker play.<6> Situations are predicate functions which evaluate to 'true' or 'false' in a specific context. For simplicity, only two predicate functions are allowed as situations at this time (in addition to 'T') -- although a wide range of arguments may be supplied. Also, only one simple predicate function at a time can serve as a situation; Boolean expressions of predicates are not allowed. The first simplifying restriction will be easy to loosen as new predicate functions are discovered which will be useful. Limiting a situation to a single predicate, however, is an important way of limiting the difficulties encountered in revising the program's MS theory or analyzing it. Actions are sequences of primitive MS processes constituting rewrite rules for transforming one structural fragment into another. In this system, an action place can also be filled by another S-A rule, allowing nesting of rules in a manner quite natural to the current textbook descriptions of MS theory.

The structure of the rule table in the program, which constitutes the program's MS theory, can be expressed in Backus normal form:

```
    <rule table>    ::=  ((T <default> <S-A rule> ... <S-A rule>)
    <default>       ::=  <action>
    <S-A rule>      ::=  (<situation> <action> |
                         (<situation> <default> <S-A rule> ...
                                                 <S-A rule>)
    <situation>*    ::=  (ISIT <subgraph name>) |
                         (CHECKFOR <variable name> <value>) |
                         T
    <action>**      ::=  (<function name> <arguments>) |
                         (PROG () <action> ... <action>)
```

----

* The function ISIT determines whether the subgraph named in its

argument place is contained in the chemical graph under consideration.

The function CHECKFOR checks to see whether the current value of the named variable is equal to the value specified. This predicate allows checking global context before determining answers to specific questions about subgraph matching.

** The basic actions (function names) known to the system are listed in Appendix A. Any action which is built out of several basic functions can be given its own name. In fact, the MS theory in the present version of the performance program contains many named complex actions.

The performance program is driven by the MS theory in the rule table by the following procedure. The program picks up the S-A rule immediately following the default action and checks to see if the current context satisfies the situation by executing the named predicate function (with appropriate arguments). If it does, the program performs the associated action by executing the named (or described) function (with appropriate arguments). The very first situation, 'T', is certain to be satisfied (since 'T' evaluates to 'true'), so the default action will be executed if none of the other situations are satisfied.

A simple illustration will make the structure of the rule table clear. Suppose it contains rules for two distinct situations: ethers and alcohols, plus a subrule for a special class of ethers, named ether1. The table would look like

```
    (T default (alcohol-situation alcohol-action)
               (ether-situation   ether-action
                              (ether1-situation   ether1-action))
```

If a compound satisfies the ether1 situation, neither the default action nor the ether action will be executed. All the processes for each situation are collected in the corresponding action. This may cause duplication if some of the processes in a rule also apply to the subrules. But modification of the rule table is made easier because of this unification.

B. ORGANIZATION AND INTEGRATION

The output from the generalization program discussed in Section IV is a set of S-A rules (with accompanying definitions of the situations and actions). The set of new S-A rules is organized without reference to any existing theory or integrated into an existing theory by exactly the same process. Each S-A rule is considered in turn. It is postulated as a new S-A rule at the top level of the rule table if its situation does not appear elsewhere in the rule table. If a new situation, S1, subsumes a situation, S2, already in the rule table (i.e., S1 is more general than S2, or S1 is contained in S2), then the new rule is inserted in the rule table so that the old rule, with S2, is below the new one. Or, the reverse may be the case, namely, that the new situation (S1) is subsumed by a situation (S2) already defined. Then the new rule must be inserted below the old one in the hierarchy. These three cases all depend only upon the program's ability to determine when one graph is contained within another. They are briefly illustrated below.

(1) If the situation does not appear elsewhere in the rule table, the new S-A rule is merely added to the top level of the rule table. For example, adding an amine rule to the sample rule table above would result in

```
     (T default (alcohol-situation alcohol-action)
                (ether-situation   ether-action
                                (ether1-situation   ether1-action))
                (amine-situation   amine-action))
```

(2 & 3) If the situation of the new S-A rule subsumes a previously defined situation, the old S-A rule becomes a sub-rule of the new rule. If the situation of the new rule can be subsumed under an existing one, the new rule becomes a sub-rule of the old one. These two cases are both illustrated by the following example. Suppose the program adds a rule (ether2-situation ether2-action) to the rule table above, where ether2-situation is an instance of

ether but more general than ether1-situation. This would result in

```
(T default (alcohol-situation alcohol-action)
           (ether-situation   ether-action
                              (ether2-situation   ether2-action
                               (ether1-situation  ether1-action)))
           (amine-situation   amine-action))
```

After deciding where the rule must be inserted, the program adds the definitions of the new situation names and action names to the system.

As this part of the program becomes more sophisticated it will have to (a) check the rules to be sure there are instances which actually distinguish them, (b) look for less cautious ways of generalizing, and (c) associate a measure of confidence with each rule so that it can resolve conflicts between rules.

VI. CONCLUSION

The Meta-DENDRAL program described here is a vehicle for studying problems of theory formation in science. It is built upon the concepts and programmed routines already available in the Heuristic DENDRAL performance program, which uses a scientific theory to explain analytical data in organic chemistry. The Meta-DENDRAL system goes beyond the performance program, however, in attempting to formulate the theory which the performance program will use.

The Meta-DENDRAL program works much like a chemist who is extending his theory of mass spectrometry by looking at collections of experimental results. The data, for both the chemist and program, are the results of mass spectrometry experiments (called FMTs here) and the associated molecular structures. By selecting some "typical" examples, first-order general hypotheses about the whole collection of data can be proposed. Then, by subsequent adjustments, the generalizations are modified to explain all the data. The new rules are then integrated into the existing corpus of theoretical statements in ways dictated by considerations of simplicity and personal preference.

The version of the meta-program which is described here suggests that the design is workable. But it accentuates the arbitrariness of our design decisions and raises the questions of what alternative designs would look like and how good they would be. It also raises a number of issues important to understanding scientific methodology in general. The design question is certainly one such issue. Others are questions concerning the criteria of acceptable generalizations, criteria of good scientific theories, and criteria for deciding on a set of primitive concepts for a theory. None of these general issues will be resolved satisfactorily in the context of this program. Yet none can be resolved for this program without saying something about the general solutions.

APPENDIX A.

PRIMITIVE CONCEPTS OF MASS SPECTROMETRY KNOWN TO THE DENDRAL PROGRAM

This list is taken from an outline given to chemists who define new mass spectrometry rules for the system. The functions at the front of the list are most primitive, those at the end are more complex, and in fact are built out of the simpler ones.

To the chemist this list serves as a reminder of the names and associated syntax of the "building blocks" available to him for defining new rules. To the present reader it is meant to illustrate the concepts already programmed into the system.

FUNCTION	(Function Arguments)*	DESCRIPTION
HOUSEKEEPING FUNCTIONS:		
ADDCHARGE	(atm)	Assign a positive charge to atm.
ADDDOT	(atm)	Assign a free electron to atm.
IONIZE	(atm)	Assign a dot and a charge to atm.
PAIRELECTRONS	(list;nolist)	Look among the atoms of LIST for adjacent atoms with free electrons. Pair up the electrons to make an explicit bond unless the pair is named in NOLIST.
REMOVECHARGE	(atm)	Take away the positive charge from atm.
REMOVEDOT	(atm)	Remove the dot (if present) from atm
FUNCTIONS FOR MANIPULATING STRUCTURE WITHOUT HOUSEKEEPING:		
ADDH	(atm)	Put a hydrogen on atm.
CHANGEBOND	(atm1;atm2;n)	Add n (pos. or neg.) to the order of the atm1-atm2 bond.

JOINATOM	(oldatm;atm;bond;atomtype;nodenum)	Bring atm into the structure -- attach atm to oldatm with bond order BOND. Give atm the atom type and node number specified.
REMOVEBOND	(atm1;atm2)	Remove the bond between atm1 and atm2.
REMOVEH	(atm)	Take a hydrogen off atm.

STRUCTURAL MANIPULATION FUNCTIONS WITH HOUSEKEEPING:

BREAKBOND	(atm1;atm2)	Replace the atm1-atm2 bond with a pair of electrons. Try to pair any other free electron with one of the new free electrons.
BREAKRING	(atm1;atm2)	Do the same as BREAKBOND when it is certain that the atm1-atm2 bond is in a ring.
ELIMINATEH	(atm)	Eliminate a hydrogen from atm, leaving a free electron.
LOSEALPHARAD	(atm)	Lose the largest radical alpha to atm.
LOSENEXTRAD	(atm)	Lose the largest radical adjacent to atm.
MAKERING	(atm1;atm2;bond)	Join atm1 & atm2 with bond to form a ring.
MIGRATEH	(atm1;atm2)	Move a hydrogen from atm1 to atm2, leaving a free electron on atm1 (unless atm1 = ANYATOM, in which case the H comes from nowh
NCLEAVAGE	(n,pct)	Break the nth bonds away from the heteroatoms in the molecule and assign intensity=pct oldint/100. If n is 0 or (quote adjacent), the adjacent bonds are broken, 1=(quote alpha), 2=(quote beta), 3=(quote gamma).
NEWBOND	(atm1;atm2)	Replace adjacent free electrons on atm1 & at with an explicit bond.

----

* The arbitrary names given to function arguments here are meant to suggest the appropriate kinds of arguments for these functions. For example, 'atm' will be replaced by the name of a specific chemical atom in the context of th actual program.

REFERENCES

(1) B.G. Buchanan and J.Lederberg, "The Heuristic DENDRAL Program for Explaining Empirical Data". In Proceedings of the IFIP-71 Congress (in press). (Also Stanford Artificial Intelligence Project Memo No. 141, April, 1971).

(2) C. W. Churchman and B. G. Buchanan, "On the Design of Inductive Systems: Some Philosophical Problems". British Journal for the Philosophy of Science, 20 (1969), pp. 311-323.

(3) J. Lederberg, G. L. Sutherland, B. G. Buchanan, and E. A. Feigenbaum, "A Heuristic Program for Solving a Scientific Inference Problem: Summary of Motivation and Implementation", In Theoretical Approaches to Non-Numerical Problem Solving (R. Banerji and M.D. Mesarovic, eds.) Springer-Verlag, New York (1970). (Also Stanford Artificial Intelligence Project Memo No. 104, November 1969).

(4) Peter B. Medewar, Induction and Intuition in Scientific Thought. American Philosophical Society, Philadelphia (1969).

(5) A. Newell, "Limitations of the Current Stock of Ideas about Problem Solving". In Electronic Information Handling (A.Kent and

O.E. Taulbee, eds.) Spartan (1965).

(6) D. A. Waterman, "Generalization Learning Techniqes for Automating the Learning of Heuristics". Artificial Intelligence, 1:121 (1970).