Artificial Intelligence 61 (1993) 209–261
Elsevier

ARTINT 959

DENDRAL: a case study of the first expert system for scientific hypothesis formation*

Robert K. Lindsay
University of Michigan, 205 Zina Pitcher Place, Ann Arbor, MI 48109, USA

Bruce G. Buchanan
Computer Science Department, University of Pittsburgh, Pittsburgh, PA 15260, USA

Edward A. Feigenbaum
Knowledge Systems Laboratory, Department of Computer Science, Stanford University, Stanford, CA 94305, USA

Joshua Lederberg
Rockefeller University, New York, NY 10021-6399, USA

Received April 1991
Revised June 1992

Abstract

Lindsay, R.K., B.G. Buchanan, E.A. Feigenbaum and J. Lederberg, DENDRAL: a case study of the first expert system for scientific hypothesis formation, Artificial Intelligence 61 (1993) 209–261.

The DENDRAL Project was one of the first large-scale programs to embody the strategy of using detailed, task-specific knowledge about a problem domain as a source of heuristics, and to seek generality through automating the acquisition of such knowledge. This paper summarizes the major conceptual contributions and accomplishments of that project. It is an attempt to distill from this research the lessons that are of importance to artificial intelligence research and to provide a record of the final status of two decades of work.

Correspondence to: R.K. Lindsay, University of Michigan, 205 Zina Pitcher Place, Ann Arbor, MI 48109, USA. Telephone: (313) 764-4227. Fax: (313) 747-4130. E-mail: lindsay@umich.edu.

* This research was supported by the National Aeronautics and Space Administrations, the Advanced Research Projects Agency of the Department of Defense, and the National Institutes of Health.

This paper draws upon [37], expanding and updating that account in light of subsequent developments and the benefit of hindsight. Large portions of descriptive material are taken nearly verbatim to make the present paper self-contained since the book itself is no longer in print.

Keywords: Expert systems; heuristic programming; organic chemistry; DENDRAL; mass spectrometry; scientific discovery.

1. Introduction

Within computer science the DENDRAL Project is noteworthy in several ways. It was the first major application of heuristic programming to experimental analysis in an empirical science, a practical problem of some importance. It was one of the first large-scale programs to embody the strategy of using detailed, task-specific knowledge about the problem domain as a source of heuristics, and to seek generality through automating the acquisition of such knowledge. It has achieved a high level of performance, because it used a substantial amount of knowledge of chemistry. The DENDRAL programs were knowledge-driven, in the sense of today's expert systems, with the *knowledge principle*—that knowledge is power—first articulated in the context of DENDRAL [16]. They were the first to use the concept of a separate knowledge base that could be edited or redefined for new problems while retaining all the same code for interpreting and using that knowledge. DENDRAL was the first rule-based system applied to a "real-world" problem. It has been used by chemists, other than its developers, in the pursuit of their own research goals. It was an interdisciplinary project that was continuously productive for over a decade. It was one of the larger, more sustained AI projects undertaken, giving it a certain prominence even apart from its successes. Perhaps most significant is that this research was an extensive empirical exploration of heuristic programming techniques; as such it was a validation of the strengths and weaknesses of these techniques and an instantiation of a philosophical concept of automatic discovery procedures whose status had long been in dispute.

The knowledge principle, as stated in [32] is:

> A system exhibits intelligent understanding and action at a high level of competence primarily because of the *specific* knowledge that it can bring to bear: the concepts, facts, representations, methods, models, metaphors, and heuristics about its domain of endeavor. (p. 1173).

This principle is now widely accepted, but at the time of the early development of DENDRAL it stood in sharp contrast to the prevailing view that ascribed intelligence primarily to reasoning power. The knowledge principle, in contrast, asserts that specific knowledge is the major source of machine and human intelligence, and will suffice with even a simple inference method. Empirical support for this principle is now abundant, and DENDRAL's performance is a major underpinning of this empirical evidence.

Whether DENDRAL was the first expert system is debatable; it was certainly the first application of AI to a problem of scientific reasoning. In his forward to [7], Allen Newell identifies MYCIN as "the granddaddy of expert systems", but acknowledges that those associated with both projects may not concur.

Issues of precedence aside, DENDRAL introduced several novel concepts of program organization that have found substantial application. The point of this case study is to illuminate the AI aspects of DENDRAL and to provide the basis for future studies of intelligent systems.

"DENDRAL" is the name of the project and also the name of the programs, sometimes further distinguished as *Heuristic* DENDRAL and *Meta*-DENDRAL. DENDRAL originally stood for DENDRitic ALgorithm, a procedure for exhaustively and non-redundantly enumerating all the topologically distinct arrangements of any given set of atoms, consistent with the rules of chemical valence. The original algorithm, which generated only ring-free (acyclic) structures (i.e., the aliphatic compounds) was devised by one of the authors (Lederberg [29–31]). The Dendritic Algorithm (as well as its later version, which in addition encompasses ring structures) is the heart of the DENDRAL programs. This algorithm defines the set of *possible* solutions through which the DENDRAL programs search for *likely* solutions. A basic feature of DENDRAL, and an important limitation on the range of applicability of its methods, is its *uniform notation for hypotheses*, here taking the form of graphs in the abstract sense of points (nodes) linked by lines (edges), depicting, respectively, the atoms and bonds of molecules.

The means by which the programs reduce the set of possible chemical graphs is the real story of DENDRAL. As Newell put it:

> DENDRAL has strong links to classical problem-solving programs, with a heuristically shaped combinatorial search in a space of all isomers at its heart and a representation (the chemical valence model) that provided the clean space within which to search. [7, page xiii]

The heuristics employed are based on judgment and specific chemical knowledge, the kinds of expertise that are popularly called experience and intuition. Heuristic DENDRAL comprises the programs that employ these methods. An important point, one on which the success of the project turned, was that the constructors of the DENDRAL programs eschewed the search for general principles of problem solving or learning in favor of specific knowledge of a special problem. The success of this project against the general background of failure, at that time, of the search for general systems does not, of course, decide the issue of better approach; but it attracted attention. The DENDRAL Project spanned approximately the second and third decades of

artificial intelligence research. This period was one of great change in computer technology and in the attitudes, aspirations, and activities of the research community working on problems of artificial intelligence.

The major goal of AI research is a productive understanding of the processes of intelligent thought. In its first decade (approximately 1954–1964), AI was characterized by enthusiasm and optimistic forecasts of the imminent solution of many problems of psychology, linguistics, mathematics, philosophy, technology, and management. By the end of the decade, when DENDRAL was in its initial stages, these optimistic forecasts were being reevaluated. The now classic example of unfulfilled forecast is mechanical translation of natural languages. This and most other problems outside previously formalized domains have proved far more difficult than many AI pioneers imagined.

In the mid 1960s AI was in a transition stage [19]. On the one hand, the newly discovered difficulty of the adopted problems was tempering optimism and in some quarters producing pessimists. Long-standing critics of determinism in general, and technology in particular, were coming out of the closet. On the other hand, some work on general problem-solving procedures still held promise in the view of many researchers. The General Problem Solver (GPS) [15] had yet to have its limitations established. A bright light was the resolution procedure [46] for proving theorems in the predicate calculus: a complete, uniform proof procedure for a general calculus seemed promising indeed. Ironically, the search for general problem-solving methods was being pursued on a broad front at the same time that specific solutions to special problems were failing to produce results.

A second aspect of this transition stage was the astoundingly rapid development of computer technology. Solid-state, second-generation hardware was replacing the slower, less reliable equipment. Telecommunications technology was being interfaced with computers on a large scale. Time sharing of computers had been conceived and was to become a reality in the 1960s. Software in general and executive systems in particular were reaching new levels of sophistication. List-processing and string-processing languages were becoming readily available. These developments combined, in the early 1970s, into facilities that eclipsed in power those available ten years earlier. Today's student using a high-level language on a workstation is interminably amused by tales of the days when a programmer would carry a box of punched cards to a computing center and return some days later only to find that a bug in the compiler, a mis-punched control card, or an operator error had terminated the job. The first large implementation of DENDRAL was done over long-distance telephone lines linking a model-33 teletype in Palo Alto with the Q32 computer at System Development Corporation in Santa Monica, with telephone bills of $2000 per month being common. When the DEC PDP-6 computer arrived at the Stanford AI Lab, the program was transferred there, running in 16K words when loaded with LISP 1.5. Although none of today's wealth of computer sophistication would have struck the early workers as

science fiction beyond their dreams, the additional load imposed by the bothersome conditions of the time were enough to swamp many well-conceived projects of realistic scope.

It was in this milieu that DENDRAL began. An important fact about DENDRAL—though not unique to it—is that it undertook a relatively narrow and well-defined problem for which there was a clear measure of success. The major lesson DENDRAL has for artificial intelligence, and for those disciplines interested in the application of AI techniques, is that it is possible to select problems of modest complexity that nonetheless baffle the novice, and to reduce these problems to some order, resulting in a problem-solving system that lends needed assistance to human intelligence. By lowering one's sights from solving broad, general problems to solving a particular problem, by applying as much specific knowledge to that problem as can be garnered from human experts, and by systematizing and automating the application of this knowledge, a useful system can be produced. This lesson underlies the success of today's expert systems.

In this paper we have a dilemma of needing to use chemical concepts to explain what the program does, what it knows, and therefore how it works. We have attempted to define many of the concepts, with apologies to those who are already familiar with them, but we have not attempted sufficiently detailed definitions to satisfy a professional chemist's need for details.

The fundamental problem of analytic chemistry is to determine the chemical structure of molecules. This is described in the next section. A *molecule* is composed of many *atoms* (mostly, in organic chemistry, carbon, hydrogen, oxygen, phosphorous, and sulfur) arranged and held in a stable three-dimensional structure by electron sharing and other energies or forces among the atoms. The electron sharing that binds one atom to another is called a chemical *bond*. Often the three-dimensional arrangement is suppressed in favor of representing only a two-dimensional schematic drawing of the atoms and their chemical bonds, a representation known as the *topology* of the molecule. It is conventionally shown as a labeled graph with non-directed arcs between nodes that are labeled as chemical atoms. The *geometry* of the molecular arrangement refers to the representation of bond angles and bond lengths that position the atoms in space. A collection of chemical atoms of known types and numbers is called an *empirical formula*, and is usually written with numeric subscripts on the atom names to indicate the number of atoms of each type, e.g., $C_8H_{16}O_2$. Any two different molecular structures with exactly the same empirical formula are called *isomers*. They are called *stereoisomers* if they have the same topology and differ only in their geometry, for example, structures that are identical except with one group of atoms positioned above or below a plane defined by the rest of the molecule. The four valence bonds of the carbon atoms are topologically equivalent, but with different labels can be disposed in two geometries, usually called *dextro* and *levo*, which are mirror images of one another.

2. The structure elucidation problem of organic chemistry

The general problem to which the DENDRAL programs apply is an important, substantive problem in organic chemistry: structure elucidation, that is, the determination of the organization of the chemical atoms in specific molecules. The problem is important because the chemical and physical properties of compounds are determined not just by the heap of their constituent atoms, but by the topological and geometric arrangement of these atoms as well. Several empirical means are available for obtaining information about the structure of a compound. Prominent among these is mass spectrometry, and DENDRAL originally addressed problems associated only with this method, although it evolved to deal with the problems of structure elucidation on more general terms. The earliest versions of DENDRAL were mostly concerned with topology, or connectivity information about molecules, because the context of its development was mass spectrometry, which is largely insensitive to geometry, or stereoisomerism. Thus to DENDRAL, different geometric forms of the same topology were entirely equivalent. Later versions added specialized postprocessing knowledge and procedures to deal with three-dimensional geometry, chemical reactivity, and data beyond mass spectra. The core structure of DENDRAL remained the same.

The identification of a molecule means at least that its topological organization is known; it is usually represented as a graph with atoms as nodes and bonds as edges. Initially DENDRAL was applied to aliphatic compounds only since an enumeration algorithm for cyclic compounds had not yet been programmed. The compounds studied were, in roughly chronological order, amino acids, ketones, ethers, alcohols, amines, thiols, and thioethers. Later versions of DENDRAL, incorporating the cyclic structure generator, have been applied primarily to steroids, in particular estrogens, marine sterols, and related compounds.

The DENDRAL programs are not limited in application to these classes of compounds, but are general mechanisms that could be applied to any compounds for which certain types of information are available. A practical limit on size of molecules amenable to the DENDRAL and conventional mass spectrometry analysis methods is, roughly, 100 atoms. Recently mass spectrometry has been successfully applied to the measurement of mass numbers of proteins with thousands of atoms [1]. If mass spectral analysis of fragments of proteins had been available, DENDRAL might have been applied to that analysis (and undoubtedly will be in the future), using superatoms (see below, Section 3.2.1) to represent individual amino acids of twenty types arranged in linear sequences, or to represent DNA sequences. As it was, the applications were selected in part for their value in developing the DENDRAL concepts and in part because they were of interest for their importance to contemporary chemistry.

The choice of structure elucidation as the target problem in the initial development of the DENDRAL system is an early illustration of the first phase of the development cycle of expert systems [21, Chapter 5]. Around 1963 Feigenbaum became interested in studying information-processing models of human empirical induction as an alternative to the puzzle-solving and game-playing task environments of Newell and Simon. In the summer of 1964 he worked with Buchanan on a simple concept formation program that induced a general concept definition from examples. In 1965 he mentioned to Lederberg that he was looking for a suitable problem in science in which to study induction of scientific hypotheses and Lederberg suggested the task of chemical structure elucidation using empirical data from mass spectrometry. There were numerous reasons why this was a good choice from a scientific point of view, including the technical merits of LISP for representing and manipulating the tree structure representation of chemical graphs. The non-AI motivation, and importance of the overall problem to an outside "client", came from Lederberg's involvement in NASA's Mariner mission and its on-board experiments to look for evidence of life on Mars. A mass spectrometer was part of the on-board instrumentation and Lederberg wanted means of identifying chemical compounds on Mars that were not also found and catalogued in Earth-based libraries. (It would have been desirable to do the analysis itself on Mars to avoid the long delays in sending and receiving radio signals. However the tempo of space technology outpaced that of computer engines. The design of Viking 1975, the first Mars mission, was frozen by 1970, before there were space-qualified computers powerful enough to run DENDRAL.)

A first prototype of the solution generator was programmed (by William C. White) in 1965, and by the end of that year a definition of the project was well in hand and a set of open problems was published [17]. Georgia Sutherland and Buchanan were recruited in 1966 and by 1968 the basic design principles of expert systems were instantiated.

Lederberg consciously invoked a recognized authority in mass spectrometry only after using his own knowledge sufficiently to demonstrate the program's potential to an outsider. In a well-orchestrated demonstration to Professor Carl Djerassi, an internationally known expert at Stanford, Djerassi saw how stupid the program was—but he saw a running program. The immediate question, which has become a cornerstone of knowledge engineering, was "What do you know about this problem that the program does not know?" Then Djerassi assigned additional experts from his laboratory, Dr. Alan Duffield in particular, to work with Buchanan almost daily in the transfer of expertise that has come to be known as knowledge engineering. One of the first descriptions of the knowledge engineering process, with a reconstructed transcript of part of a knowledge engineering session, was published in 1970 [9].

2.1. Mass spectrometry

Only recently, with the development of scanning tunneling microscopy, have atoms become examinable by optical means. To determine the structure of a molecule of any complexity, chemists normally resort to indirect methods. One of the most fruitful of these is mass spectrometry. The foundations of mass spectrometry were laid around the turn of the century by Wilhelm Wien and J.J. Thomson. By 1935, the main features of modern instruments had been developed by A.J. Dempster and others. An explosion of interest in mass spectrometry occurred in the 1960s as it became a laboratory tool of value to analytical chemists [40]. Carl Djerassi, for example, had co-authored one of the early, influential books [11].

The essence of the technique is to decompose large molecules into fragments, infer the composition of these overlapping fragments, and use this information to guess the structure of a molecule that would break to yield the same observed set of fragments. A molecule bombarded with electrons in the mass spectrometer does not always end up in just two fragments; some fragments decompose further. Target molecules in a sample do not all fragment in the same places. Thus the data reflect a statistical distribution of fragments, with relative frequencies determined by the relative extent to which some bonds are more frangible than others. For example, single bonds generally break more readily than double bonds. Also, certain groups are relatively immune from breaking in the mass spectrometer. For example, C=O will generally break off as a unit or remain as a constituent of a larger unit, but will seldom itself break.

Many of the fragments are electrically charged (that is, they are ions, normally singly charged), that can be accelerated by an electric field. Fragments of lesser mass reach higher velocities than those of greater mass (and equal charge).[1] The beam of ions thus produced passes into a magnetic field perpendicular to its path and is deflected, in accordance with the laws of electromagnetism and mechanics, with higher velocity (lower mass) fragments being deflected by greater amounts. The result is a sorting of fragments by mass (actually by mass to charge ratio). The early instruments were of low resolving power (thus the name low-resolution mass spectrometry), that is, they resolved masses of chemical compounds and fragments only to the nearest integral mass unit. High-resolution mass spectra, which allowed precise identification of the numbers of atoms of each type in a compound fragment, became available routinely after DENDRAL was begun and were incorporated into it. This also allowed more precise heuristics but did not alter the structure of the program.

[1] Positively charged ions are usually examined, but negative ions can be studied in specially designed instruments.

2.2. Other analytical methods

A chemist can (and usually does) gather other kinds of information about an unknown compound. At the very least there will be some information about its source and extraction methods, which often suggests probable constituents. Laboratory analyses may also provide further clues of this sort. In addition other instrument-based techniques provide information about the compound. The most common among these are *gas chromatography*, *infrared spectrometry*, *ultraviolet spectrometry*, and *nuclear magnetic resonance spectrometry* (NMR). Information from these and other techniques was incorporated in DENDRAL in three ways. First, constraints inferred manually from these data were added for each problem to the constraints used by the generator. Second, knowledge and programs were added to infer many of these constraints automatically. Third, knowledge and programs were added for postprocessing to predict experimental data that could discriminate among candidate structures.

3. The solution method

DENDRAL is not a single program but a set of programs. Some of these programs may be used alone to perform single subtasks of importance to the problem of chemical structure elucidation. Some may be linked in various ways by different executive programs to form coherent systems for larger tasks. To organize the description of this collection of intertwined programs we first note that they comprise basically two systems. The first, called Heuristic DENDRAL, is a system that incorporates specific knowledge of chemistry and mass spectrometry, accepts a mass spectrum and other experimental data from an unknown compound as input, and produces an ordered set of chemical structure descriptions hypothesized to explain the data. The second system, called Meta-DENDRAL, accepts known mass spectrum/structure pairs as input and attempts to infer the specific knowledge of mass spectrometry that can be used by Heuristic DENDRAL to explain new spectra. Heuristic DENDRAL is a performance system and Meta-DENDRAL is a learning system. Our emphasis in this paper is on the former. For information on Meta-DENDRAL, see [5,6].

It must be kept in mind that the descriptions here are somewhat idealized expositions of the programs. Early versions, described in other publications, were used to produce many of the results summarized later. No single version has been systematically applied to all the specific structure elucidation problems investigated by the project.

3.1. The plan–generate–test organization of DENDRAL

The basic method of Heuristic DENDRAL is an important extension of the generate-and-test paradigm in which a *generator* enumerates potential solutions, which are expressed as chemical graphs in the case of DENDRAL. It is often desirable, though not essential to the paradigm, for the generator to be non-redundant as well as exhaustive: that it guarantee every possible solution be enumerated exactly once. A non-exhaustive generator would be unconvincing; a redundant one would be inefficient or non-terminating. DENDRAL's way of limiting generation is with a planning program that can suggest constraints on generation. This component distinguishes plan–generate–test from generate-and-test. Constraints may take the form of guidance about parts of the solution space to ignore or parts to focus on, or both.

The DENDRAL planner is itself an hypothesis formation program that employs task-specific knowledge to find constraints for the generator. It is important that the planner be extremely flexible in the sense of permitting the ready addition of new knowledge. Ideally, the knowledge will be highly modular so that it is possible to add new knowledge without reevaluating the old. Further selection is performed by the third and final stage, a testing program called the PREDICTOR. This is a program that examines each proposed solution and rejects those that fail to meet certain criteria. The PREDICTOR incorporates a theory of mass spectrometry that predicts what fragmentations a proposed chemical structure will undergo in a mass spectrometer and constructs a mass spectrum accordingly. This predicted spectrum may then be compared to the one produced in the laboratory.

Both planning and testing programs constrain the set of likely solutions with respect to analytic data available (e.g., a mass spectrum) and knowledge about the data source (e.g., partial theory of mass spectrometry). It might be thought that it is much more economical to apply constraints first, in the planning stage, rather than last, in the testing stage. In some measure this is true, and every effort is made to pre-constrain the generator. However, pre-constraint is not always possible, in this case because the theory generally requires a completed structure, and not a partially generated one, to make a strong enough statement to allow pruning.

The feature that gives the plan–generate–test paradigm its cohesiveness is the uniform representation used by the three components. In the case of DENDRAL this representation is chemical graphs. The planner devises hypotheses that reject and/or propose certain classes of chemical graphs, the generator generates chemical graphs, and the tester represents fragmentation processes in terms of chemical graphs. This common representation is the glue holding DENDRAL together.

Table 1 introduces the names of some of the component programs and data structures of the DENDRAL system and outlines their interrelationships.

Table 1
Organization of the heuristic DENDRAL programs

Operation	Components	Input	Output
Planning	MOLION	Mass spectrum	Molecular ion constraints
	Planning rule generator	Planning rules	Constraints
	PLANNER	Planning rules	Superatoms GOODLIST BADLIST
Generating	Acyclic generator CONGEN GENOA STEREO	Constraints	Candidate molecular structures
Testing	PREDICTOR	Candidate molecular structures	Most plausible structures
	MSPRUNE	Mass spectrometry rules	Structures consistent with spectrum
	REACT	Reaction chemistry rules	Structures consistent with known reactions

3.2. The DENDRAL generator

The seminal insight for DENDRAL was the original algorithm for exhaustively and non-redundantly generating acyclic structures as reported in [29–31]. This algorithm, embodied in computer code, was the basis of the first DENDRAL system. When the limitation to acyclic structures was overcome [2, 3], DENDRAL increased its scope dramatically.

The cyclic generator includes the acyclic generator as a component. The complete algorithm is complex, but its correctness has been rigorously proved [2]. Because the graph generation algorithms were proven to be correct in theory, DENDRAL's procedures may be directly applicable to other knowledge-based systems in which hypotheses may be represented as labeled graphs.

The program's ability to generate hypotheses exhaustively and without redundancy is the key to assuring scientist-users that every plausible hypothesis appears in the output. Implausible hypotheses are pruned from a search space that is guaranteed to be complete for reasons explicitly stated in the constraints—and only for those reasons. This is known as the *exclusion paradigm* in AI and has guided the construction of other knowledge-based systems in science [23]. In philosophy at least since the 17th century this has been called *induction by enumeration*.

In spite of the fact that there is no way of proving that a program is a correct embodiment of a complex algorithm, the generator program has passed the important test of enumerating the correct *number* of structures for many cases where the number of structural isomers was computed independently, and has

been extensively checked against hand-calculated examples. We do not seriously doubt the program's correctness.

3.2.1. The generator

The generator of molecular structure descriptions is itself algorithmic. It is based on Lederberg's notational algorithm, which assigns a unique name to every topologically distinct structure. Considerable design and implementation effort was needed to transform the notational algorithm into an efficient generating algorithm. In the case of molecular graphs without rings, the acyclic structures, the transformation was straightforward, and the main effort was in making the implementation flexible enough to be guided by user-defined lists of substructures to avoid or to include (BADLIST and GOODLIST).

For structures with rings, however, the notational algorithm was not easily transformed into a generating algorithm. It was important, and difficult to prove, that the generation was complete, i.e., that every distinct structure with a given number of nodes of given types was generated. The particularly difficult issue, besides efficiency, however, was avoiding generation of duplicate structures. Because of the many-fold symmetries of some structures, it was extremely difficult to guarantee that no single structure was duplicated. The generator that was finally designed and implemented was proved to be both complete and non-redundant. It was later rewritten for greater efficiency. Although tuned to generate graphical descriptions of chemical structures, this generator can be used for any problem in which solutions can be described graphically. (We are not aware of any actual uses outside of chemistry, however. Computer programs and LSI circuit design might be promising.) As with generation of acyclic graphs, flexibility is a primary concern since the generator must allow guidance with respect to a variety of forms of information that may be available about plausible and implausible parts of the solution space. The algorithm is described more fully in [4, 9, 12].

A self-contained system called CONGEN embodies the cyclic structure generator. Aside from its importance as a powerful system in its own right, CONGEN illustrates the basic concepts of CONstrained GENeration that underlie the entire DENDRAL effort and the plan–generate–test paradigm.

We will now describe CONGEN because it demonstrates in clear fashion the concept of heuristic search in the space of possible chemical structures. CONGEN permits the user to constrain the enumeration by specifying several types of constraints. These are:

(1) substructures that must be present with specific cardinality;
(2) *sizes* of rings that must be present;
(3) the number of hydrogen atoms that must be associated with a given structure, without being specific as to where they are bonded;
(4) the number of isoprene units that must be part of the generated

structures. (An isoprene unit is a Y-shaped grouping of five carbon atoms; they are particularly abundant in natural products.)

The first step in using CONGEN is to define *superatoms*. A superatom may be any connected graph that is treated as a unit, acting in the role of an atom in the construction of larger structures.

The GENERATE step produces all possible structures of atoms and superatoms following the algorithm of the cyclic generator but rejecting all structures violating defined constraints.

The final step, called IMBED, expands ("explodes") the superatoms one type at a time so that the resulting final structures are representations of chemical structures in terms of atoms and bonds only. During imbedding, structures are also constrained according to the user's specifications, since the expansion of a superatom could give rise to violations of constraints that were not violated by intermediate structures.

The program is interactive, permitting chemists to revise their list of structures and constraints at any stage. Thus if the generation produces an inordinate number of possibilities, chemists may add further constraint information until they are able to winnow the set down to manageable size. The final enumeration ideally would be a single structure, though in practice this is not often the case. However, if the number of structures is sufficiently small to permit examination of each member, the chemist's stock of judgment and intuition that remains uncodified may, at times, be sufficient to make a good guess as to the correct structure.

The following is a record of a session with CONGEN. The example is very simple and of no particular chemical interest but serves to illustrate the program's syntax and behavior. The constraints listed below illustrate the flexibility and power of the program to use information about chemical structure that may be inferred (manually or automatically) from a variety of analytical techniques.

The chemical information available is as follows:

C1. The empirical formula is $C_{12}H_{14}O$.
C2. The compound contains a keto group in a five-membered ring.
C3. There are three protons (H's) alpha (adjacent) to the carbonyl group.
C4. There are two vinyl groups (–C=C–) and four vinyl protons.
C5. There is no conjugation (alternation of double and single bonds).
C6. There are no diallylic protons (hydrogens at the middle carbon of a diallylic structure: –C=C–C–C=C–), nor protons alpha to both a vinyl and the keto group.
C7. There are only two quaternary carbons, one in the keto group and one in one of the vinyl groups.
C8. There are no additional multiple bonds.
C9. It is assumed there are no three- or four-membered rings.

C10. There are no methyl groups.

[Comments are bracketed and refer to problem definition statements C1–C10, above. The following transcript has been abbreviated. Structure drawings are slightly modified from those done by a program authored by Ray Carhart that has come to be known as "Carhart's DRAW Program". It avoided the need for an expensive graphics terminal by producing teletype compatible drawings. Today of course much more readable drawings can be made on workstations.]

WELCOME TO CONGEN, VERSION VI.
#DEFINE MOLFORM C 12 H 14 O
MOLECULAR FORMULA DEFINED

#DEFINE SUBSTRUCTURE Z

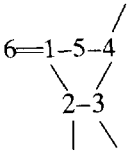
[Z is the structure required by constraints C2 and C3.]

(NEW SUBSTRUCTURE)

>RING 5 [C2: form five-membered ring.]
<LINK 1 1 1 [C2: add keto group.]
>ATNAME 6 O
>HRANGE 2 1 1 3 1 2 4 1 2 5 2 2
[C3: atom 5 will have 2 H's, which will be alpha to the carbonyl; atom 2 will have the third H alpha to C=O.]

>DRAW NUMBERED

SUBSTRUCTURE Z (HRANGES NOT INDICATED)
NON-C ATOMS: 6→O



#DEFINE SUBSTRUCTURE CH3 [C10: CH_3 is a methyl group.]
(NEW SUBSTRUCTURE)
>CHAIN
>HRANGE 1 3 3
>DONE
CH3 DEFINED

#DEFINE **SUBSTRUCTURE V** [V is the vinyl group: C=C]
(NEW SUBSTRUCTURE)
RING 2

[Comment: Here several other structures are defined. VV is a double vinyl:

C=C–C=C. VH is a "tagged" vinyl. VCHV is a diallyl: –C=C–C–C=C–. CH0 is a quaternary carbon (no hydrogens).]

#GENERATE
SUPERATOM: Z
RANGE OF OCCURRENCES; **AT LEAST 1**
SUPERATOM: V
RANGE OF OCCURRENCES: **AT LEAST 2**
SUPERATOM:
'COLLAPSED' FORMULA IS C 3 Z 1 V 2 H 9
CONSTRAINT; **LOOP Z NONE**
[This constraint prevents Z from bonding with itself.]
CONSTRAINT: **SUBSTRUCTURE CH3 NONE** [C10]

[Comment: Similarly several other constraints are specified here: Exactly zero CH0 structures (constraint C7), zero VV structures (C5), zero VCHV structures (C6), zero three- and four-membered rings (C9), exactly four VH structures (C4B), and exactly three double bonds (C8).]

18 STRUCTURES WERE GENERATED

[Comment: The following is one of the 18 structures; as required, it contains one Z and two V's that will be imbedded later. Most of the constraints have been employed already, but a few remain for the imbedding steps.]

#DRAW ATNAMED 1

#1:
C–C–V
| | |
V–C–Z

#IMBED

[Comment: Z will now be imbedded first. More superatoms could be imbedded now also.]

SUPERATOM: **Z**
NUMBER TO BE IMBEDDED: 1
SUPERATOM:
THE 'EXPANDED' FORMULA IS O 1 C 8 V 2 H 14
CONSTRAINT: **SUBSTRUCTURE VCHCO NONE**
[C6B: VCHCO defines protons alpha to both V and C=O to be prohibited.]
CONSTRAINT: **RING 3 NONE** [C9]
CONSTRAINT: **RING 4 NONE** [C9]
CONSTRAINT:

[Comment: As CONGEN begins to consider each structure it prints "#" and for each new structure produced it prints "."]

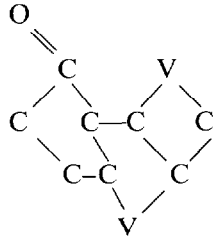
#..#..#..#.#....#....#..######....#....#..##..

29 STRUCTURES WERE OBTAINED

#DRAW ATNAMED 1

[Comment: The following is a sample structure in which Z has been imbedded. Note that V's are not yet imbedded.]

#1:



#IMBED

[Comment: The V's will now be imbedded.]
SUPERATOM: **V**
NUMBER TO BE IMBEDDED: **2**
THE 'EXPANDED' FORMULA IS O 1 C 12 H 14
CONSTRAINT: **SUBSTRUCTURE CH0 EXACTLY 2**
[C7: we must end up with exactly two quaternary carbons.]
CONSTRAINT: **RING 3 NONE** [C9]
CONSTRAINT: **RING 4 NONE** [C9]
CONSTRAINT:

#.#.#.#..#..#..#..#..#..#..#..#.#.#.#..#.#..

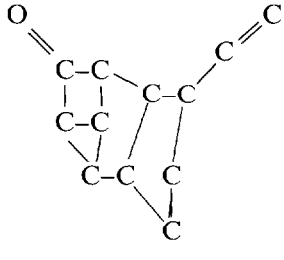
#.#..#.#..#..#...#..#..#.#.#.#..

47 STRUCTURES WERE OBTAINED

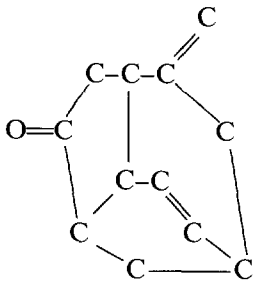
#DRAW ATNAMED (5 6)

[Comment: The following is a selection of final structures 5, 6.]

#5:



#6:



#EXIT
DO YOU WANT TO SAVE YOUR SESSION ON FILE?: **YES**
FILE NAME: **CONGEN.EXAMPLE**
SAVED ON CONGEN.EXAMPLE
EXIT

3.2.2. GENOA

CONGEN is capable of producing all possible structures consistent with any set of constraints that can be formulated in its constraint language. However, the entire set of constraints must be given at the outset; CONGEN then will produce a set of fully specified final structures ("leaves" of the tree of possibilities). The final structures are of course produced sequentially, and need not be displayed until the user requests. Because generation is depth-first, the user can be informed of the number of structures generated up to the present moment. In fact, the initial implementation was breadth-first but the program was modified to allow users to examine complete structures before all were generated. The program can be interrupted if the size of the set is too large, and more constraints can then be added in an attempt to bring the set of alternatives down to manageable proportions. At any time, some of the fully specified alternatives can be displayed to guide the selection of additional constraints, or to make certain that the generation is proceeding along sensible paths. However, it is not possible to stop CONGEN and examine the nonterminal nodes of the generation tree in an attempt to examine a class of solutions. The GENOA program incorporates a generation method that does permit the examination of intermediate stages of the generation. This provides the user with an important additional control over the solution process. This feature is a byproduct of the organization that was selected in order to resolve CONGEN's inability to deal efficiently with redundant and ambiguous substructures.

Frequently, the chemist's data specify that two (or more) substructures must be present in the solution, but it is not known whether the two substructures may have shared parts. To use CONGEN, the user must be certain that the substructures specified as constraints do not overlap. This means that suffi-

ciently small substructures be specified so that they share no parts, or each of the possible overlaps must be treated as a separate problem. GENOA removes this burden from the chemist by considering all possible overlaps and treating them as separate cases (GENOA is GENeration with Overlapping Atoms).

It is also possible that two or more substructures may be plausible alternatives for a given item of structural information, that is, two separate hypotheses could account for the datum. Again, CONGEN requires that in such a situation the alternatives be considered as separate problems. GENOA spares the user this burden by permitting the specification of a set of substructures as alternatives, and considering automatically the multiplicity of cases this generates by interaction with other alternative sets and patterns of overlap. This organization of the generation process is called constructive substructure search. It represents a substantial improvement in design by lessening the burden on the user and by providing more, and more timely, feedback of progress during the solution of a structure elucidation problem.

We have been discussing primarily connectivity isomers. As noted earlier, it is possible that two molecules with the same atoms and the same connectivity are nonetheless not superimposable in three-space, that is, they cannot be deformed, rotated, and translated to be congruent. Such non-superimposable instances of the same connectivity isomer are called stereoisomers. There are two classes of reasons why superimposability may be impossible. The first is because of geometric constraints: the configurations are inherently asymmetric because of differing configurations of substituents around a center of symmetry. The second class is due to physical constraints: the orientations of the bonds are fixed by atomic forces that are not overcome at normal temperatures. These two classes of stereoisomers are called *configurational* and *conformational*. In part, as with connectivity isomers, they are of interest because they may manifest different chemical and physical properties. They are also important because many compounds of biological and medical interest have stereoisomers, with only one of the isomers occurring naturally.

3.2.3. Stereoisomer generation

The DENDRAL system includes programs that non-redundantly generate all distinct connectivity, configurational, and conformational isomers. These programs operate in three stages in the order given, and each stage may employ user-supplied constraints. The connectivity isomers are generated by CONGEN (and GENOA), the configurational isomers are generated by augmenting the descriptions of the CONGEN/GENOA-supplied connectivity isomers using the STEREO program [42–44], and the conformational isomers are generated by augmenting the configurational isomers supplied by STEREO with descriptions supplied by the CNFCAN program [18].

After the connectivity isomers have been generated, it is possible to elaborate some of the three-dimensional properties of those structures or examine

the list in various ways. CONGEN has deliberately been kept unencumbered by these options so as to preserve its prospective avoidance of symmetric (redundant) structures. Additional programs have been written to allow users to look at the different relative three-dimensional orientations of any structure or set of structures and to prune a set of candidate structures with respect to constraints inferred from data about such orientation. This was largely the result of creative and highly technical work by Dr. James Nourse. Details are beyond the scope of this paper. The problem-solving heuristic at work here, though, is well known in AI: break up huge problems into subproblems.

3.3. Planning

Constraints to keep the generator from producing all possible structures are necessary for all but the simplest problems. We have seen how certain types of constraints can be stated in a manner appropriate for the cyclic generator. In CONGEN and GENOA these constraints are formulated by the user. An early version of the DENDRAL system used a set of planning rules for aliphatic compounds elicited from an expert mass spectrometrist [9], with a form of interaction that is now known as knowledge engineering. Each rule specified features of the spectrum that are associated with a particular structure. The planning phase then examined the spectrum for the specific evidence and if it was present the associated structure was added as a GOODLIST constraint. The declarative representation of BADLIST and GOODLIST in the first version of DENDRAL (as global lists of names of chemical substructures) facilitated writing the planning programs that added items to those lists. The planning rules are found in [8].

An extension of this planning method, in which the rules themselves are automatically generated by the planning rule generator, is described in [10]. Although we did not make the connection at the time, we were following the advice John McCarthy first published in 1958 (see [39]) that we should represent knowledge declaratively so that it can be changed easily by humans or programs. (Of course, his paper prescribes logic as the declarative language—part of the advice that we were deliberately avoiding.) Explicit, changeable representations of knowledge are now the cornerstone of expert systems.

The latest DENDRAL PLANNER program further automates some aspects of constraint formulation. It embodies a form of mass spectrometry theory that the users instantiate to produce a particular theory for the particular class of compounds with which they are working. This theory is not a sweeping generalization but a collection of specific hypotheses about the likely loci of breaks that will occur when a compound in the class is placed in the mass spectrometer.

It is important to note that there is not a one-to-one relationship between a

process described in the theory and data points in the mass spectrum. A process may occur in several places in a single compound, and a data point in the spectrum may be the result of several different processes. This makes the data interpretation problem more complex than in many expert systems, for example for troubleshooting, where the observations have more significance.

Input to the planning phase is: (1) the basic skeleton of the compound class, i.e., the structure common to all members of the class, (2) definitions of the various breaks that might occur, and (3) a mass spectrum (either low or high resolution). The program determines the empirical formula (using the MOL-ION program, a heuristic program that is usually able to identify the molecular ion from mass spectrum data alone) even if the spectrum does not contain a peak corresponding to it. It then formulates constraints that specify which sites on the skeleton are likely, and which are unlikely, locations for the nonskeletal groups of atoms. The substructures that are attached to the skeleton are called substituents, since they substitute for the hydrogens that would otherwise be there.

The first version of planning was written for the acyclic generator applied to the aliphatic ketones. Planning information for these compounds was obtained by careful questioning of mass spectrometrists about ketones in the mass spectrometer, that is, about the kinds of fragmentation processes that were most likely to occur. As is commonly the case with today's knowledge engineering assignments, this knowledge had never been completely codified and thus the effort to specify the information for DENDRAL was valuable to chemists as well. It proved successful and provided the first version of the system in the full plan–generate–test form where planning was based on planning rules that encompassed sophisticated judgmental rules from an expert. These rules of data interpretation took the form:

$$\{\text{data points}\} \rightarrow \text{situations in which these data are produced}.$$

In particular:

$$\text{Set of m/e peaks} \rightarrow \text{subgraph}$$

where "→" is read "implies that the molecular structure graph contains". Note that the rules of data interpretation are the "inverse" of rules that predict or simulate the behavior of a device that produces the data in the first place. The theory by which mass spectrometers are designed and built, for example, allows theoretical calculations of bond dissociations based on bond strengths and the energy level of the electron beam. Not only are these calculations computationally expensive, they are not invertible. That is, they do not allow one to calculate what chemical compound was put into the device, given its output. Data interpretation is a common problem posed to developers of expert systems for just these reasons. Furthermore, the solution is just the same: ask an expert how to interpret the data.

For many analytical problems it is known what class of compounds is under investigation, particularly when the source of the material is known and the empirical formula of the molecular ion has been determined. If chemists have experience with this class, they may know enough about the fragmentations that are most likely to have occurred to allow them to predict efficiently the behavior of an arbitrary compound of that class. This class-specific information—even in its predictive form—can be used to produce constraints that will reduce the DENDRAL generator's output, as was done for a few subclasses of steroids [49–51].

Predictive rules of mass spectrometry were represented in DENDRAL as productions of the form

situation → action

or

chemical subgraph → mass spectroscopy process

where the actions, or processes, were the fragmentations and rearrangements that occurred in the instrument that caused particular charged fragments to be produced and subsequently collected and recorded in the mass spectrum. Fragmentations or breaks are defined by specifying (1) the bonds that break, (2) the charge location, (3) any accompanying transfers of hydrogen atoms into or out of the charged fragment, and (4) any accompanying losses of small molecules such as water. Breaks are represented schematically as in Fig. 1.

To convert this knowledge into a form that is useful for interpretation requires fixing a substantial part of the situation description, e.g., the steroid skeleton. Then the variable parts of the situation (in our case the chemical atoms not assigned to the skeleton) can be added hypothetically in plausible combinations to the fixed part of the situation. The predictive rules can then be applied to a situation that is fully enough described to allow some rule(s) to

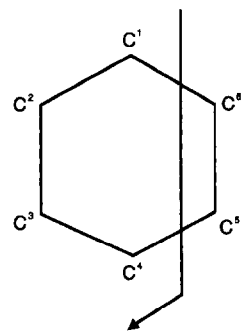


Fig. 1. Schematic representation of a fragmentation. This fragmentation involves breaking two bonds. The arrow indicates the positive charge placement.

fire. The right-hand side process then can be simulated to determine which data points one would see if this modification of the situation were true. As long as the combinatorics of modifying the situation are not crippling, this general technique should allow representing knowledge (when known) only in its predictive form. It may be more efficient or more precise to represent it in its inverse form in many cases, but it is sufficient to represent it only once in some cases.

In the case of DENDRAL this general technique was applied in the following way. From the empirical formula and the definition of the class skeleton, the program can readily determine the numbers and types of atoms and unsaturations that are not part of the skeleton and which thus are available to form the substituent structures that distinguish the particular compound under investigation. It remains to determine the exact composition of each substituent and its location on the skeleton. To do this determination, the mass spectrum is consulted for evidence that can be associated with each of the hypothesized breaks. In general, all these breaks may have occurred, but the fragments resulting from a particular break will have compositions that differ depending on how the substituent atoms are divided among the fragments. The program determines for each break all the ways the substituent atoms may be divided and the evidence in the spectrum associated with each division. This process of course does not determine the exact location or composition of a substituent. However, combining the information from all the breaks further narrows the possible arrangements.

The PLANNER, and its component MOLION, are good illustrations of the design of the DENDRAL system and the general philosophy of modular and flexible design that characterized the project. Little explicit formal theory underlies these programs. However, their ad hoc character is their strength when combined with program flexibility. Since so much of any scientist's knowledge is intuitive and judgmental, it cannot be readily formalized. Programs such as those we have just described nonetheless permit this knowledge to be employed in useful ways. The processing speed and clerical superiority of the computer in fact amplify, in many cases, the power of the scientist's informal knowledge. A particular case in point is the PLANNER's analysis of spectra taken from unseparated mixtures of estrogens, described in [49]. The record-keeping and cross-checking demands of this task are too great to permit a thorough job by hand, yet the program is able to identify correctly the number and structure of compounds in mixtures.

3.4. Testing

If planning were perfect and generating were exhaustive, no spurious solution candidates would be produced and no solutions would be missed.

DENDRAL planning, however, is not perfect and so a testing phase is applied to each candidate. There are two reasons why planning cannot be perfect even in principle. First there may be tests that can only be applied to fully specified solution candidates or to *sets* of them. Second there may be tests that are too costly to apply in the planning phase but are not too expensive to apply to the smaller set of solution candidates. Both these factors operate with DENDRAL.

The program that tests the generator's output has been called the PREDICTOR because of what it does. The production rule version was written by Buchanan in the late 1960s after a procedural version had been written. A summary of that account was published as part of the story of how production rules were chosen to encode knowledge for MYCIN:

> The [DENDRAL] programs are knowledge-intensive; that is, they require very specialized knowledge of chemistry in order to produce plausible explanations of the data. Thus a major concern in research on DENDRAL was how to represent specialized knowledge of a domain like chemistry so that a computer program could use it for complex problem solving.
>
> MYCIN was an outgrowth of DENDRAL in the sense that many of the lessons learned in the construction of DENDRAL were used in the design and implementation of MYCIN. Foremost among these was the newfound power of production rules . . . . [7, p. 8]

This program is driven by a set of production rules that define a theory of the behavior of compounds in a mass spectrometer. The production rules state that structures of a given form will fragment in certain ways. They are thus like the *predictive* rules used in PLANNER, except that the format is more general. Given a set of productions, PREDICTOR applies them to a proposed structure graph to predict what ions will be produced; the productions may be applied recursively to all ion graphs they produce. The result is a predicted mass spectrum that can be compared to the actual data from which DENDRAL began. Structures that produce spectra in close agreement with the data are ranked highly.

The situation-part of a PREDICTOR production describes a chemical graph. The action-part of the production consists of a set of operations that alter the graph to produce other graphs. The action-part may contain a complete production as a component, so that some operations are performed conditionally. Thus the productions are strongly hierarchical. The operations that PREDICTOR permits are any LISP functions; predefined operations define fragmentations, hydrogen transfers, loss of neutral molecules such as water, and means for computing relative abundances of the products of the fragmentation processes. Section 3.4.2 provides more details.

The PREDICTOR control structure applies the production rules to a set of

ions. First a molecular ion is constructed and placed as the only item on an *ion list*. The set of productions is scanned, and each one that is applicable is applied to produce one or more new ions that are then added to the end of the ion list. When all applicable productions have been applied, the current ion is put into the *spectrum list* and the next ion on the ion list is selected. Some ions may arise from more than one fragmentation process. When all ions on the ion list have finally been processed (or the maximum permitted depth of recursion has been reached), the spectrum list then corresponds to a mass spectrum. Each entry is an ion that has a mass or composition, and each has associated with it a number that denotes its relative abundance. The abundances are accumulated and normalized to produce the predicted mass spectrum.

The first class of compounds to which DENDRAL was applied was the amino acids. In predicting the behavior of these compounds in the mass spectrometer, it was assumed that every bond that could break, would break, and they would do so one at a time. This assumption is the *zero-order theory of mass spectrometry*, so-called because it takes into account no site-specific information. This theory proved insufficient for other classes of compounds although it is close to correct for the amino acids.

Intermediate versions of the predictor (early 1970s) used class-specific production rules of the form described below. A later version, called MSPRUNE, reverted to the half-order theory (Section 3.4.4) as a general predictive theory in order to avoid the necessity of encoding class-specific knowledge or of assuming that knowledge of all of the relevant classes had been encoded. Another predictive program, called REACT, was added to the system to make predictions about the reaction products that one would observe if each of the candidate compounds were put in various chemical reactions [56]. Thus, just as constraints may be inferred from data resulting from different techniques, tests can be specified that use different techniques. In general, of course, the precision of the interpretive or predictive form of knowledge, and the ease of specifying each, will determine whether knowledge is used for constraining a generator or testing its results.

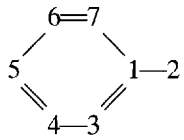
3.4.1. Representation of chemical graphs

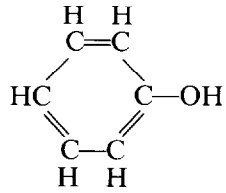
Each node of a chemical graph is given a number and a name. These are redundant and both exist only for reasons having to do with LISP conventions, but we include both in our examples so that they correspond to program listings. The node is described by indicating the type of atom, its connectivity to other nodes, the number of unsaturations (double bonds) from the node (these are called "dots" because dots are used in diagrams to stand for electrons, a pair of which can make an additional bond) and the number of hydrogens attached to the node. Connections are indicated by a list of node names enclosed in parentheses. The entire node description is enclosed in

parentheses. Finally the list of nodes defining a structure is enclosed in parentheses along with the name of the structure.[2]

To illustrate, consider C_6H_5OH. We arbitrarily use the number of a node as its name:

NODE DIAGRAMS





Atom name	Atom type	Node number	Neighbors	Dots	Number of hydrogens
C1	C	1	(2 3 7)	1	0
O2	O	2	(1)	0	1
C3	C	3	(1 4)	1	1
C4	C	4	(3 5)	1	1
C5	C	5	(4 6)	1	1
C6	C	6	(5 7)	1	1
C7	C	7	(1 6)	1	1

The complete notation is (spacing on the page is irrelevant and is chosen for readability):

```
(PHENOL
  (C1 C 1 (2 3 7) 1 0)
  (O2 O 2 (1) 0 1 )
  (C3 C 3 (1 4) 1 1)
  (C4 C 4 (3 5) 1 1)
  (C5 C 5 (4 6) 1 1)
  (C6 C 6 (5 7) 1 1)
  (C7 C 7 (1 6) 1 1))
```

[2] The original representation of a chemical graph was as a LISP list with sublists indicating branching. A second notation was as a list of LISP atoms [for example, (C1, O2, C3)], each with its own property list indicating the properties of NEIGHBORS, NUMBER OF HYDROGENS, DOTS, etc. The representation described here is faster and more compact because it uses fixed positions in lists instead of explicit names and values. Note that there is some redundancy in the connection tables regarding numbers of neighbors, "dots", and hydrogens for an atom with a given type. This was done to decrease computation time. The representation used in CONGEN is a compressed matrix representing the same connectivity information in a canonical form.

The same notation is used to describe classes of structures in which some of the information is not completely specified and partial matching of left-hand sides is required. In particular, substructures are defined in such a way that their connections to larger structures are not uniquely defined. The symbol "X" when used in a connectivity list denotes exactly one connection to a node *outside* the subgraph. The symbol "--" means "don't care". When used in the position of a node, "--" denotes zero or more connections outside the substructure; when used instead of an integer as a dot or hydrogen-count parameter, it indicates any integer value, including zero.

The alcohol substructure is X–C–OH, represented as:

```
(ALCOHOL
  (C1 C 1 (2 X --) --)
  (O2 O 1(1) 0 1))
```

3.4.2. Representation of production rules

Each production consists of a situation-part and an action-part. The situation-part is a predicate, which is either true or false of the structure under consideration. For example, the predicate ISIT (X) evaluates to true if X is a substructure of the *active ion*, that is, the one to which the production is being applied. The special argument-free predicates NIL and DEFAULT may be used as the situation-part. NIL always evaluates to false; this predicate is used to take a production out of action temporarily without actually deleting it from the production set. DEFAULT evaluates to true if and only if all previously tried productions have proved inapplicable.

Other situation-part predicates are specific to chemistry. DB(N, M) is true if there is a double bond between nodes numbered N and M. WHERE(S) is like ISIT, but returns a list of correspondences between atoms in S and in the active ion if there is one. MISSING(N, M) is true if nodes N and M are not connected. ON(S, N) is true if substructure S is contained in the active ion, with node N substituted for any unspecified atoms in substructure S. ON, and WHERE, where true, produce as side-effects an indication of all the mappings from subgraph to active ion that constitute a match of the subgraph. This information is available for use by functions in the action-part of the production.

The action-part of a production is more complex, but is intended to use names and descriptions commonly used for processes that occur in the mass spectrometer. It may have more than one component. Each component may itself be a production. A component also may be the name of a function that will be executed, selecting its parameters from the current context. Finally, a component may be a list of two or three functions specific to mass spectrometry. A *break function* produces new ions from the active ion and adds them to the ion list. An *intensity function* determines an intensity for each

newly formed ion. It may be a number indicating the relative abundance of the parent ion to be assigned to the daughter ion, or a function that specifies how to calculate an intensity. For example, a process may produce an ion whose intensity is half that of the parent. A *transfer function* produces additional ions according to rules of hydrogen transfer (or transfer of other radicals) but does not add these to the ion list. (The transfer function is optional and may be absent.) Each function list component has a label that is used in the output to indicate the source of each spectral peak. The label is the first symbol in the function list.

To summarize, a PRODUCTION is (SITUATION-PART ACTION-PART).

A SITUATION-PART is one of the following three forms:

(1) (ANY LISP PREDICATE), for example, ISIT, WHERE, DB, ON,
(2) (NIL),
(3) (DEFAULT).

An ACTION-PART is a list of ACTION-PART COMPONENTs.
An ACTION-PART COMPONENT is one of the following three forms:

(1) (LABEL; BREAK-FUNCTION; INTENSITY-FUNCTION; TRANSFER-FUNCTION (optional)),
(2) ANY LISP FUNCTION,
(3) a PRODUCTION.

The following production for estrogens is a production rule formulation of a basic five-rule theory of estrogen fragmentation. The *intensity function* of each component of the production is here the integer 100, which simply means that any generated ion will be assigned 100 percent of the intensity of the ion from which it derives. The estrogen production has five function list components; each function list is associated with a particular break definition; the break name is used as the label of its function list. The functions BREAKBND and HTRANS are a break function and a transfer function, respectively, to break a bond, or a set of bonds, and to transfer hydrogens. BREAKBND creates only one charged fragment in each break, namely the one containing the first atom of the first pair. Also, BREAKBND refers to the list of correspondences created when the left-hand side was matched, in this case by the WHERE function.

```
((WHERE ESTROGEN)
  (B (BREAKBND((14. 15) (13. 17)) 100(HTRANS −1 0))
  (D (BREAKBND((9. 11) (14. 13) (16. 17))) 100(HTRANS −2 −1))
  (C (BREAKBND((9. 11) (14. 13) (15. 16))) 100(HTRANS −1 −0))
  (E (BREAKBND((11. 12 (8. 14))) 100(HTRANS −1 0))
  (F (BREAKBND((9. 11) (8. 14))) 100(HTRANS −1 0))
```

A syntactically more complex production is one that defines McLafferty rearrangement, shown in Fig. 2. The structures that undergo this rearrangement contain the substructure named GRAFMC, shown below.

```
(GRAFMC
  (1 C X (2 4 --) 1)
  (2 -- X (1 --) 1)
  (4 -- X (1 5 --))
  (5 -- X (4 6 --))
  (6 C X (5 --)))
```

The production is labeled MCLAFFERTY. It corresponds to a mass spectrometric process that occurs in many cases and has thus been defined and named so that it may be referenced in defining class-specific productions. The situation-part of the MCLAFFERTY production is (WHERE GRAFMC). The action-part of MCLAFFERTY uses six functions. Two of them, LASTION and LASTINT, simply return the previous ion and its intensity, respectively. Three of them MRRFGT, MRRFGT2, and MRRFGT3, produce the required hydrogen migrations before performing breaks that produce new ions. The final function, GAMMACLEAVAGES, produces (possibly) several ions by producing all gamma cleavages.[3] These functions were named for three reasons: readability, explanation, and reusability. Each of these reasons is an important design consideration in today's expert systems, but this was not obvious in the late 1960s.

The other subgraphs referenced in the MCLAFFERTY production are defined in [37]. The integers used as intensity functions are interpreted as before: the intensity of the generated ion is the indicated percentage of the

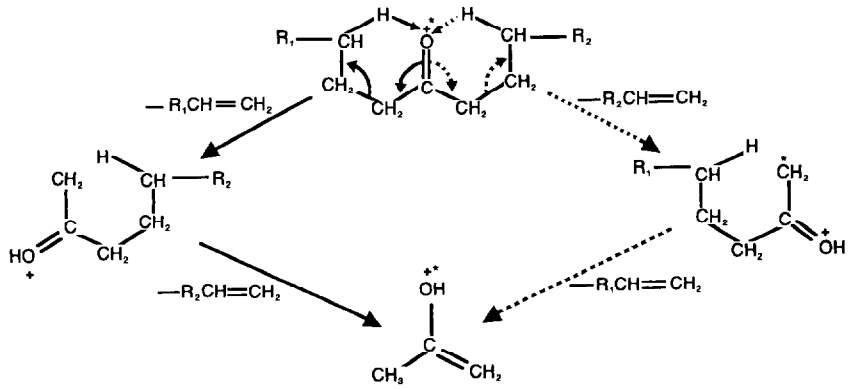


Fig. 2. McLafferty rearrangements. (Reproduced by permission. Source: G.W.A. Milne, ed., *Mass Spectrometry: Techniques and Applications* (Wiley, New York, 1971). Copyright © 1971, John Wiley & Sons, Inc.)

[3] The bonds connecting a structure to a group are said to be alpha to the group. Bonds more distant are called beta, gamma, and so forth.

intensity of the current ion. The MCLAFFERTY production is

```
(MCLAFFERTY
  ((WHERE GRAFMC)
    ((WHERE GRAFMCMETHYL)
      (MCLAFFCH3 MRRFGT3 100)
      (MCLAFFERTY MRRFGT 200))
    ((WHERE GRAFDBLMCALPHA)
      (DBLMCLAFF MRRFGT3 100)
      (DBLMCLAFF MRRFGT2 100)
      (MCLAFFERTY MRRFGT 200))
    (DEFAULT
      (MCLAFFERTY MRRFGT 200))
    ((WHERE GRAFMC5)
      (MCLAFFERTY +1 LASTION (LASTINT 10) (HTRANS 1))
      (GAMMA GAMMACLEAVAGES (LASTINT 100)))))
```

Two of the functions used by these productions are (the definitions, of course, are in LISP):

- GAMMACLEAVAGES
  (1) Find all bonds gamma to node 2.
  (2) Break each bond in turn.
  (3) For each bond broken, create an ion containing node 2.
  (4) Return the list of ions created.

- MRRFGT
  (1) Migrate a hydrogen atom from node 6 to node 2.
  (2) Break the bond between nodes 4 and 5.
  (3) Return a list containing the ion containing node 4.

3.4.3. Ranking the candidate explanations

When there is more than one hypothesis consistent with the data, additional tests are needed to discriminate among the candidates. Often this involves additional experimental work. In DENDRAL we looked at two different kinds of tests: those involving data point in the mass spectrum already in hand and those involving additional data. Additional data might come from NMR, C13-NMR, or from reaction chemistry as mentioned above. Here we just explain the use of data in the mass spectrum that were not already used to infer constraints for the generator.

Different models, or "theories", of mass spectrometry can be used to predict the fragmentations of a molecular skeleton. The type of fragmentation theory to be used depends largely on the context of the structure elucidation problem. When one initially studies a new class of compounds, or when one attempts to discriminate among different candidate structures obtained from some unusual CONGEN problem, it is usually appropriate to use some universal form of

fragmentation theory that expresses very general chemical principles. Although a general theory will be applicable and will not be biased in its predictions, it may well prove to have poor discriminatory power. Fine discrimination among related structures generally requires more refined fragmentation theories wherein one assigns different plausibilities to alternative fragmentation processes. When processing isomers from some well-characterized class, the appropriate fragmentation theory may well involve the detailed specification of substructures, their bond cleavage processes, and the accompanying specific transfers of hydrogen atoms or other molecular fragments.

In the following subsection we show how even the most general "half-order" fragmentation theory can serve to discriminate among isomers of moderately complex structures such as monoketoandrostanes. (The skeleton of this class is illustrated in Fig. 3.) Refinements of the simplest half-order theory involve, first, the use of estimates of relative plausibilities of fragmentation processes of differing degrees of complexity and, subsequently, the association of relative plausibility values with some classes of bond cleavage.

3.4.4. The half-order theory of molecular fragmentation

The simplest model of molecular fragmentation is the ALLBREAKS, or zero-order, "theory" that predicts ions arising from all possible bond cleavages and combinations of cleavages and transfers of atoms between fragments. Such a model is too general for almost every problem in computer analysis of mass spectra. (However, the zero-order theory was the method of spectrum prediction in the first application of Heuristic DENDRAL to amino acids.) DENDRAL's "half-order theory" of mass spectrometry is a constrained version of the ALLBREAKS model of molecular fragmentation: virtually any bond is broken subject to the constraints listed. The half-order theory is a very loose model that constrains a set of possible processes to those that are more plausible. It still does not represent a comprehensive model of fragmentation.

The constraints that can be expressed in the half-order theory include

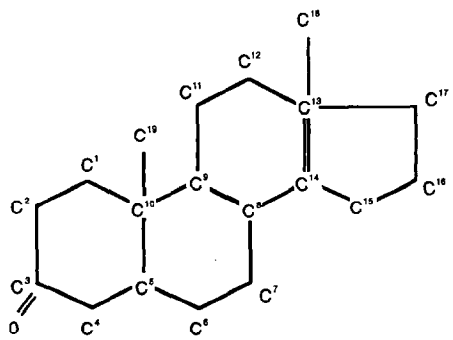


Fig. 3. Monoketoandrostane skeleton.

limitations on the number of bonds that may be broken and the number of allowed hydrogen transfers into or out of the charged fragment. Each predicted ion is formed by a "process" involving (1) one or more cleavages of the graph into two parts, (2) possible H-transfers, and (3) possible losses of neutral fragments. Each cleavage may be a break of one acyclic bond, two bonds within a ring, or a group of three bonds in an edge/fused ring system. One complete process shown in Fig. 4 would, for example, involve fused ring cleavage, simple ring cleavage, and acyclic bond cleavages; such a process would involve a total of six bond breaks. To summarize, the constraints used with the half-order theory include:

(1) Allow cleavages or pairs of cleavages each involving one or two single bonds, with not more than three bonds in total. Do not break double or triple bonds, or bonds in an aromatic ring.
(2) Prohibit the cleavage of two (non-hydrogen) bonds from the same carbon atom (for this cleavage would formally leave a fragment that is normally energetically unfavorable).
(3) Restrict transfers between fragments to at most two hydrogen atoms.

A simple use of the half-order theory for testing is implemented in the MSPRUNE function. MSPRUNE helps a chemist reject CONGEN structures by estimating the difficulty of determining the origin of any specific ion in the spectrum from each of the candidate structures. Even in this very limited form, the half-order theory can be of value in helping to identify candidate structures compatible with spectral data. For example, a simple ring cleavage and hydrogen transfer are a simpler explanation than cleavage of a fused ring system. MSPRUNE uses such differences to eliminate candidate structures [56].

Generally, we have found it to be more effective to employ the data in the entire observed mass spectrum, and rank candidate structures according to how

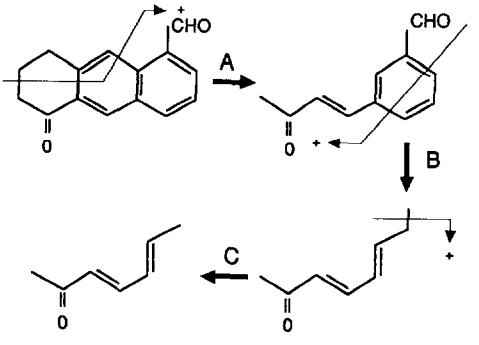


Fig. 4. Complex break process.

well they serve to explain the spectral data. This ranking is accomplished through the MSRANK function, which allows the user to define the constraints of the half-order theory and to specify the form of the scoring function. The score assigned to a candidate structure is determined from the importance accorded to those of the observed ions that can be generated by the allowed fragmentations of that candidate. As ions at higher mass and intensity values are generally of greater structural significance, degrees of importance are accorded to each observed ion in the spectrum by some function of its mass and intensity.

4. Results and conclusions

As has been emphasized, no one program is called DENDRAL. It should also be clear that DENDRAL is not a special-purpose system for solving just a few specific problems in chemistry. Heuristic DENDRAL is a *framework* for helping chemists with structure elucidation problems in various ways. Some of the knowledge embodied in the system, such as the stability knowledge codified in the a priori GOODLIST and the a priori BADLIST, is general. This is also the case for the basic mass spectrometry theory that is embodied in the PLANNER and PREDICTOR programs. The class-specific and problem-specific chemical knowledge used by Heuristic DENDRAL is supplied by the chemist-user; the programs, however, are intelligent enough to understand such specifications and make use of them.

In addition to being a mass spectrum problem-solving engine of various configurations, the programs can be used in a more general vein as aids to chemists with structure elucidation problems. CONGEN, for example, is a symbolic graph manipulator for the chemist, analogous to algebraic and analytic symbolic manipulators for the mathematician, such as REDUCE [22] and MACSYMA [38].

4.1. CONGEN results

Of all the DENDRAL programs, CONGEN (later extended and renamed GENOA) is clearly the most generally useful and has been applied to current research problems by several chemists. The range of problems to which CONGEN is applicable is great although only a few dozen applications were made by the Stanford Group. Each of these is a relatively specialized problem whose importance is only clear in the context of a larger chemical issue.

Applications of CONGEN are no longer under close supervision by DENDRAL project members. This fact speaks for the maturity of the program but makes it difficult to enumerate problems to which it has been applied. It works well on "medium size" problems in which it is combining eight to ten atoms

and superatoms at a time. If the chemist has interpreted data in terms of large superatoms (along with other constraints) then the program can manage structures with on the order of 100 atoms.

The final number of structures is strongly dependent on the chemist's ability to aggregate separate atoms into superatoms and infer other structural constraints from the available data. Because of the subtle and powerful interactions among constraints, it is difficult to estimate the final number of structures that will be produced. Nevertheless, a user can interrupt CONGEN at any time to request an estimate of the number of structures remaining to be generated. Although the estimates are rather crude, they are valuable for indicating that the program needs more constraints, for example, if 200 structures have been generated and CONGEN estimates that it is only 5 percent finished.

Much of the collaboration with other chemists has been informal and has been undertaken with the dual expectation of providing useful service while leading to new CONGEN research areas. Some examples of problems are listed here to give a sense of the range of structures for which CONGEN has been of service.

(1) *Chemical constituents of body fluids* (*e.g., organic acids, amino acids*). Generate structures within constraints derived from knowledge of chemical isolation procedures and human metabolic processes to identify compounds in the gas chromotogram/mass spectrum traces of patients with suspected metabolic disorders of genetic origin.

(2) *Terpenoids*. Generate structures within constraints (including four to six large superatoms) to identify C_{15} and C_{20} compounds isolated from natural sources. These were also tested to see if candidate structures obeyed the isoprene rule that restricts interconnections of 5-carbon superatoms [13, 52].

(3) *Marine sterols*. Determine the sterols (within numerous constraints) that could be metabolic precursors of known sterols in marine organisms in order to guide identification of unknowns.

(4) *Insect secretions*. Confirm the structural possibilities in problems involving the identification of insect hormones and other insect defense secretions.

(5) *Rearrangement products*. Obtain all structural possibilities under available constraints to help solve the structures of chemical and photochemical rearrangement products of unsaturated hydrocarbons (e.g., a tricyclic $C_{11}H_{12}$ hydrocarbon) and polycyclic molecules [55].

(6) *Ion structures*. CONGEN has also been applied to the problem of generating the structures of gaseous ions. This application has been done, for example, for the case of triethylamine, $(CH_3CH_2)_3N^+-$. See [53] for an explanation of the underlying mass spectrometry fragmentation mechanisms of this and related compounds.

(7) *Pharmacologic agents.* CONGEN has been used to help solve the structures of several compounds displaying pharmacologic activity including some nitrogen heterocycles, pesticide conjugates, and metabolic products of microorganisms.

4.1.1. Status

This project was specifically conceived because it addressed the AI problem of understanding scientific reasoning in the context of an important and ubiquitous practical problem in organic chemistry. It was developed initially by the members of the team oriented to computer science. After several years of transition, it was transferred to computationally oriented chemists on the team with a continuing view toward making it a practical and usable system. Eventually, after more than 15 years of intense academic effort, the DENDRAL programs were licensed by Stanford to a commercial concern marketing software to chemists. The ideas had been refined and tested in the academic environment beyond the point of justifying continued effort and funding on the basis of research. Moreover, a commercial company is usually a more effective organization for technology transfer than a university, and we hoped DENDRAL would be widely used. We know it was, in our hands, a useful tool for the chemist solving structure elucidation problems in its symbiotic (CONGEN/GENOA) form, and that its performance rivaled that of non-expert chemists in its full (plan–generate–test) form on classes of compounds for which it possesses the requisite knowledge. Efforts were made to make the programs usable and "friendly" to working chemists. Nonetheless, the impact of these programs on the conduct of organic chemistry has been quite limited. It is likely that few chemists even know of the existence of the programs. Only a few laboratories have made use of the whole system.

A version of GENOA (re-written in the C language) is commercially available,[4] but as of this writing development work on it has not been done for many years. Parts of the DENDRAL system such as algorithms for matching substructures are used commercially, for example to search a drug company's library of compounds for those that contain a biologically active substance.

There are of course a host of difficulties facing those attempting to transfer technology from the research laboratory to application, even when the applications themselves are to research problems carried out by other researchers sympathetic to these issues. However, there are also some limitations of DENDRAL that go beyond the usual problems of technology transfer. In brief, the system best handles smaller problems for which expert chemists need less assistance. The expert chemist can still reduce many problems that are beyond DENDRAL's scope to manageable, paper-and-pencil problems.

[4] Molecular Design, Ltd., 2132 Farallon Drive, San Leandro, CA 94577, USA.

There are some important qualifications that must be made to the preceding observation. In the first place, expert systems are often developed to be of assistance to the less-than-expert practitioner. Thus they may be valuable in the sense of making available a level of performance and ability that falls short of, or at best rivals, their best human counterparts, as when electronic troubleshooting skills are made available to new field service personnel. However, this does not seem to be a useful niche for DENDRAL; the difficult structure elucidation problems are those faced by the experts, and the easier problems that require solution, for instance in routine medical applications such as identification of toxic substances, are frequently solvable by other methods such as library search. DENDRAL is not useful for the largest, most difficult problems for which experts want assistance.

There is another role that DENDRAL may play. That is the role of solution checker. Since the DENDRAL generator is thorough in its consideration of the constraints provided by the data, if it were to rule out a solution "found" by a chemist, that would with certainty indicate an error either in the analysis by the chemist or the statement of the constraints. Closely related to this function is the role that DENDRAL can play in discovering alternative solutions not found by the chemist. Since DENDRAL's generator is mercilessly exhaustive in its consideration of alternatives, it may well find solutions that are consistent with the given constraints but were not considered by the chemist. Of course it may be the case that the chemist's own analysis employed constraints or other knowledge that was not explicitly reported or that could not readily be formulated in DENDRAL's constraint language. In an informal study of about a dozen published papers, DENDRAL often did discover alternative solutions using the published constraints. However, in all but one case the right solution (as determined by additional experimental data) was the one the chemist found. In the lone exception, the chemist recognized a few of DENDRAL's alternative solutions as potentially correct although he had overlooked them. These observations suggest that routine checking of solutions before publication could improve the accuracy and completeness of published reports and raise confidence in them. Whether this would be a worthwhile application depends on the costs of the computation involved, among other factors.

The model of use implicit in the CONGEN/GENOA systems is that a chemist will bring to the program a body of data about an unknown compound, will transform the data into a set of generator constraints, and will then wait for a list of possible solutions. In practice, if the generator produces only a few candidates the problem was possibly easy enough that the chemist would have succeeded unaided. If the candidates number one thousand or more, the problem remains virtually unsolved. It is only when perhaps a dozen to a hundred candidates are generated that the program has made an important contribution. This is a rather small window. Without computer assistance, the

chemist will not know exactly how large the space of remaining candidates is, but will be quickly aware if it is unmanageable. In that case, more data are sought. If that is not possible, data may be "invented"; that is, the chemist assumes that the compound must satisfy other plausible though unproven constraints, such as being a member of a particular class of compounds (perhaps because all compounds from the same source are members of that class).

Seemingly, DENDRAL would assist in this sequential model as well, by relieving the chemist of the burden of generation at each step, while guaranteeing completeness. The generation is modular; that is, additional constraints can be applied to a candidate set to further reduce it without regeneration *de novo*. We do not know why this strength of the CONGEN/GENOA model is not marketable.

There is a related way in which DENDRAL could help, namely by assisting in the selection of new data. For example, the candidate set could be subjected to "what if" experiments. What if we knew that the compound was of class C; how many candidates would remain? What if it contained substructure S; how many candidates would remain? The constraint that would most greatly reduce the candidate set would then determine the next piece of laboratory work to be done. Unfortunately, DENDRAL offers the user little assistance with this method (although some preliminary work on this problem was done).

We summarize with some speculations on why DENDRAL is not used more widely:

(1) Chemists do not know it exists.
(2) The hardware/software are too expensive.
(3) The chemist does not wish to invest the time to learn the system.
(4) Exhaustive generation is not seen as essential to the structure elucidation problem.
(5) An attitude that "That is not the way it is done", or that a tool "not invented here" is not worth using, or that "Machines can't think; that's my job".
(6) The chemist and DENDRAL do not collaborate intensively over a long period of time (much of the chemist's time is spent in the lab); so, like a fire alarm, its value is underestimated.
(7) DENDRAL is not cost-effective for any single individual (though it may be for an entire company).
(8) Chemical and pharmaceutical companies do not cooperate and share resources.
(9) Pieces of the DENDRAL system, e.g., structure-matching algorithms, were easier to market than the whole system.
(10) The niche that DENDRAL fills is not perceived by chemists as important enough to warrant use of the system.

4.2. Lessons for the implementation of intelligent systems

DENDRAL is an extensive case study of one of the first knowledge-based programs. Because it is a direct progenitor of MYCIN, and thus of today's expert systems, the development of DENDRAL is historically significant. It is designed around the exclusion paradigm of problem solving in which all implausible answers are excluded from a complete solution space, leaving only the plausible ones. It is implemented within the programming structure of the plan–generate–test paradigm. The performance of DENDRAL illustrates the power of this paradigm. It also clearly points to a number of details of this methodology that deserve careful consideration in its application. The existence of the generator is a *sine qua non*. Before the dendritic algorithm, there was no unique naming convention and no generating function that guaranteed complete, non-redundant coverage of the space of topological descriptions of chemical structures. Not every application has such a generator; most do not. The invention of one may require considerable intellectual effort.

A generator of English sentences might be briefly considered as a generator of hypotheses in some problem areas. The main drawbacks, however, are that (1) there are far too many legal sentences containing, say, fifty words or fewer, (2) there is no guarantee that the solution would be describable in a sentence of any arbitrarily fixed length—it may require 51 words, for example—and thus there is no decision procedure for problem solving, (3) the obvious ways of generating sentences are purely syntactic, yet information available for constraining the generator almost certainly carries semantic content.

Some specific lessons we can suggest are the following:

Lesson 1. The efficiency of the generator is extremely important. Even with effective planning and testing, the power of the problem solver will often be limited by how quickly candidate solutions can be enumerated. It is particularly important that constraints can be applied effectively.

One interesting improvement in CONGEN involves spending a little extra time at the beginning of a session to save enormous amounts of time later. There are numerous, logically equivalent ways of specifying constraints to CONGEN that differ greatly in efficiency. It is unreasonable to expect a chemist using CONGEN to be familiar enough with the program itself to know which specifications are more efficient than others. (It is also unreasonable to ask the chemist to seek advice every time from one of the local CONGEN experts.) Thus a "smart interpreter" of constraints was developed [20]. The goal is to have it accept the chemist's statement of the problem and understand both the chemistry and the computational procedure well enough to transform the given constraints into an equivalent, more efficient set of specifications.

Lesson 2. The use of depth-first search, which provides a stream of candidates, is generally better (in an interactive program) than breadth-first search, in which no candidates emerge for examination until all are generated. To a

programmer the two methods are equally easy to implement. To a chemist waiting for help with a structure problem the difference is substantial. It is far more pleasing to see some answers quickly, and it is more efficient in those cases where the first few answers reveal mistakes in the problem specification. In those cases, the chemist can interrupt the program, change the constraints, and restart.

Lesson 3. Planning is in general not simply a nice additional feature but is essential for the solution of difficult problems. As much knowledge as possible should be brought to bear at this stage rather than at the testing stage, because this point is where the search can be cut drastically.

Lesson 4. Every effort to make the program uniform and flexible will be rewarded. The user should be provided with as many options as one can think of (with defaults established to remove the burden when the flexibility is not needed). Every decision strategy and parameter that is hardened into the program will become a limitation not open to examination or easy modification and not easily remembered.

Lesson 5. An interactive user interface is not merely a nicety but is essential. For a high-performance computer program to capture the sustained, widespread attention of working scientists, it must contain a large number of features that make it easy and pleasant to use. These features are commonly termed "human engineering aspects" of a program. In very rare instances, a program will be so useful that it will be widely adopted even without proper attention to human engineering. More often, programs that are understandable only to programmers are used, if at all, only by programmers.

The prompts and descriptions printed by the DENDRAL programs have been designed by chemists to be terse, informative, objective, and courteous. These are not always consistent goals, but with careful attention the dialogue can be free of flagrant affronts to our feelings. Along the same lines, it should be noted that a person's right to privacy cannot be ignored in scientific programs. In CONGEN we rely on the standard protection mechanisms built into the computer system, but we will need more security if we are to satisfy users in chemical industries.

Lesson 6. An interesting extension of the plan–generate–test paradigm could improve its power: search and generation might be combined into a single problem solver. In the context of DENDRAL this combination would mean that the generation of isomers would be guided by a search through a related problem space. The problem states would most naturally be chemical structure graphs, and the transformations would append, delete, and rearrange constituents. A proposed alteration would be considered for its effects on the mass spectrum, and a hill-climbing technique, for example, might drive the search. Successful ("warming") modifications would become GOODLIST constraints on generation. Numerous variations on this theme can be envisioned. We briefly explored hill-climbing methods and the use of an evalua-

tion function to give us best-first search, but here our focus on mass spectrometry kept us from developing these ideas. We wanted to guide the search by comparing predicted partial spectra (from partial structures) with the original spectrum, and then adding more structure so as to reduce differences.

Goal regression has been proposed [24] as a way of modifying a hypothesis to improve its predictive accuracy. In this model, the difference between observed and predicted data focuses the search for modifications that will reduce the difference. Something like this model was explored briefly for DENDRAL in the following form:

(a) Given an observed mass spectrum MS_o and a candidate molecular structure S,
(b) Predict a mass spectrum MS_p for S,
(c) Analyze the difference between MS_o and MS_p,
(d) Adjust S to minimize that difference.

The problem that we encountered was that we lacked GPS-like difference-reducing operators. We were unsuccessful in finding any in mass spectrometry because very small changes in molecular structure may cause very large changes in the distributions of mass spectrometry processes, with correspondingly large changes in the resulting spectra. Thus we had to rely on forward generation of alternatives without benefit of back-propagation of useful adjustments.

Lesson 7. Choice of programming language is becoming less of an issue. We originally did not have access to a language that combines the flexibility and debugging power of LISP with the running speed and exportability of C, for example. This language conflict causes a dilemma at the start of a large programming effort whenever the designer hopes for widespread use of the resulting program. Networking provides a partial answer to the exportability question, since widespread use can be accomplished by long distance sharing of a complex program. The increased efficiency and availability of LISP also make this issue nearly irrelevant now.

Lesson 8. Providing assistance to problem solvers is a more realistic goal than doing their jobs for them. In the first place it removes some of the psychological barriers that people often exhibit toward machines. Also, the amount of work involved in automating the whole task may far outweigh the benefits and in any case will delay the appearance of any benefits considerably.

Lesson 9. Record keeping is an important adjunct to problem solving. Every laboratory assistant is expected to keep a good laboratory notebook: the same should be true for a computer apprentice. Of the many ways of realizing the goal of helpful records, only some have been explored in the context of DENDRAL.

We did not provide interactive explanations. But we did expect the DENDRAL programs to provide three different kinds of notes to back up its reasoning from initial data and constraints to ranked hypotheses:

(a) a record of initial conditions, intermediate conclusions, and final results;
(b) a complete record of the interaction between chemist and program (including erroneous entries and typing mistakes);
(c) a trace of the program's reasoning steps.

Lesson 10. In order to use a program intelligently, a user needs to understand the program's scope and limits. The scope, roughly, is the broad class of problems that the program is designed to solve and the context in which solutions will be found. The limitations of a program are those aspects that create exceptions to perfect performance over the whole scope. For example, enumerating polymeric structures is outside the scope of CONGEN, while its working definition of aromaticity is a limitation that is more easily changed. Operationally, the scope is the broad definition of the problem that can only be changed at the cost of writing an entirely new procedure. The limitations are the explicit and implicit items in the problem definition that are added to make the problem solvable and that may be changed or removed more readily. It is not a sharp distinction; the point is that a chemist needs to understand the program's interpretation of the problem—its implicit and explicit assumptions—before it can be used responsibly and confidently.

Lesson 11. The context in which problem solving proceeds is essential information for interpreting the solutions. The more an assistant can make explicit the assumptions and initial conditions of a problem, the easier it is for an investigator to understand the answers. This has always been true, but the emergence of computer programs as assistants brings the problem clearly into focus. The only step we have made along these lines with DENDRAL programs is to keep a good laboratory notebook, as described above. One of the items we try to make explicit at the time problem solutions are printed is the set of assumptions under which the program arrived at those solutions.

Lesson 12. DENDRAL employs uniformity of representation in two senses: (a) in the knowledge used to manipulate chemical structures, and (b) in the data structures used to describe chemical structures and constraints. In the knowledge base, uniformity is important as a means of understanding and conveying the contents of the knowledge base as well as in problems of acquiring new knowledge. In DENDRAL, knowledge is uniformly represented in a very general form: production rules. The uses to which the knowledge is put, however, are many. This arrangement achieves the best of both worlds: we have uniformity of representation with its virtues of modularity, simplicity of control structure, and perspicuity, and we have the inherent power of multiple sources adding to and making varied uses of the common knowledge base.

In the underlying data structures, uniformity is important for program efficiency and ease of program development. We were fortunate to have found just the "right level" of abstraction of chemical molecules in terms of graphs. The ball and stick model of molecules and its analogous graphical description

are grossly inadequate for many purposes but they were quite adequate for the reasoning that DENDRAL needed to do. It is a convenient shorthand for what chemists know about chemical bonding, and thus was readily understandable. However, it did require doing some new mathematics, graph theory and group theory in particular, in order to design correct algorithms, e.g., to generate ringed structures without duplication of symmetric graphs.

To summarize: successful problem solving in nontrivial domains (1) requires surprisingly large amounts of specialized as well as general knowledge, (2) requires different forms of organization for different tasks, and for different purposes within a given problem domain, (3) requires the productive interaction of this knowledge, not merely its accumulation, and (4) can benefit from representation schemes that bear part of the burden of the inferential process. For these reasons, which have the status of empirical propositions about cognitive systems generally and human minds specifically, we conclude that the current emphasis on knowledge engineering within AI, for which DENDRAL is a key example and important case study, is both central and prerequisite to the development of artifacts of general intelligence.

4.3. Project organization

There have been few successful, long-term interdisciplinary projects in the history of science, but we believe DENDRAL should be counted among them. The project worked cohesively for a decade, and it involved productive interaction of researchers from the disciplines of chemistry, computer science, genetics, philosophy, physics, mathematics, electrical engineering, management science, and psychology.

It is difficult to give a recipe for this success, but we believe we can list some important ingredients. First, the task was conceived in such a way as to appeal to many interests; it could have been described as a "pure" mass spectrometry problem, or a "content-free" hypothesis formation problem, but it was not. This task is not prohibitively difficult: it can be understood (with a moderate effort) by anyone with a modest technical background. One scientist, with knowledge of both chemistry and computer science, was willing to coordinate and arbitrate the often-conflicting efforts of the group, and was able to do it because others felt sufficient respect for his ideas and vision to sacrifice some of the traditional autonomy and rugged individualism of scientists. The project leaders were skilled managers who had learned to delegate responsibility through management of other academic organizations. They also shared a willingness to take risks with unproven personnel. Not the least important, a natural selection occurred, resulting in a staff of specialists each of whom was truly willing to go more than half way to understand the other's discipline, paradigms, and arcane jargon.

There was also a genuine desire among the computer science personnel to

create programs of value to chemists on the way to solving the big problem. Although many of these utilities did not use AI methods, they provided tangible benefits to the chemist collaborators whose assistance was essential. This piece of common sense ("quid pro quo") is missing in projects that "skim the cream" in a new problem area and that have left colleagues disgruntled about AI.

We may offer no magic advice here, but the lessons are important, and mistakes are costly. Interdisciplinary work is antithetical to most scientists, no matter how wistfully they long for it. It is expensive folly to establish a project or institute and fill it with scientists from a variety of disciplines, selected only on the basis of scientific credentials. Without leadership, specific common goals, mutual empathy, human consideration, and a great deal of effort, the result will be a collection of scientists none of whom has a colleague. Finally, it should be noted that it is not easy to get funds for a large, interdisciplinary project. It is important to find a sponsoring agency that is willing to invest in long-term research, because continuity is critical.

4.4. Knowledge engineering

One major thrust of this work has been the exploration of methods for acquisition, representation, and use of knowledge. We have referred to the design of such methods as *knowledge engineering*, which is the basic *engineering* aspect of the work.

At the time of inception of the DENDRAL Project, the major emphasis of most AI research was a search for general methods of problem solving. Relatively little effort was devoted to the design of systems that embodied and used specialized knowledge. The paradigm case of the search for general methods was research on the resolution method of proving theorems in the predicate calculus [46] as applied to a variety of problems, for example robotics [45], by a number of computer scientists. There were other important elaborations of this theme, for example the General Problem Solver [15]. There were also some exceptions, projects in which emphasis was placed on exploiting specialized knowledge (e.g., [34]). Chess and checker programs are also good examples, as are symbolic mathematics aids such as MACSYMA [38]; such work, however, was in the minority. Mass spectrometry was not an area of strength for Buchanan and Sutherland. Thus the time-consuming and error-prone dialogue between Buchanan and Duffield forced development of styles and tools that are also now commonplace.

The situation today is different. Many significant projects can best be characterized as the development and application of knowledge-based systems. The success and example of the DENDRAL project in all likelihood played an important role in the establishment of the new emphasis in AI, though it was not the only force acting in this direction. We now are more convinced than ever that the design of knowledge-based systems is an important emphasis.

We take it to be self-evident that problem solving in a specific task domain requires special knowledge of that task domain. This was not contested, merely not emphasized, in early AI work. In a predicate-calculus-based system, specialized knowledge was encoded in the axioms, the theorem-proving procedures, and the criteria of interest; it was not ignored. In the General Problem Solver, specialized knowledge was encoded into the definitions of the problem space and transformations; it was not ignored. What was not fully appreciated was the sheer *amount* and *variety* of such knowledge underlying intelligent behavior. General methods went awry when the unavoidable profusion of specialized knowledge swamped the heuristic methods and, further, outran the abilities of the programmers to encode it all. This breakdown happened as soon as attention was directed away from highly abstracted, simplified "toy" problems toward applications of utility outside AI itself.

A surprisingly large amount of specialized knowledge is needed to achieve expertise in even a very circumscribed field. The fact that long periods of time are required to become an "expert" is evidence that expertise is knowledge-intensive. For example, Simon and Barenfeld [48] present evidence that the difference between expert and novice chess players lies almost exclusively in their differing degree of familiarity with commonly occurring patterns of chess pieces. This familiarity is reflected in speed of recognition and ability to recall, and is acquired by extended experience. It is estimated that the chess expert is familiar with between 10,000 and 100,000 such patterns, which is also the range of the word-recognition vocabulary of a fluent speaker. In addition to pattern familiarity there is a host of other knowledge that novice and expert alike must share, and that passes unnoticed in a casual analysis of game playing. This knowledge includes, for example, contextual information about the nature and purpose of games and of competition generally. In the case of mass spectrometry, the knowledge base includes not only specialized knowledge of technique, but a large amount of information about the underlying subjects of chemistry and graph theory, any portion of which may profitably be brought to bear in the solution of a particular structure elucidation problem. A prerequisite of successful performance by DENDRAL was the encoding of significant portions of this knowledge base, a job that has taken thousands of man-hours.

To gain sufficient problem-solving power in the face of the needed quantity of knowledge, a knowledge representation scheme must be sufficiently specialized; making it so is a major part of the engineering problem. Consequently, successful knowledge engineering initially requires decisions about knowledge representation; in particular, specialized representations for specialized applications. A representation that is uniform for *all tasks* is doomed to impotence in problem-solving power, although we have argued previously that uniformity of representation has significant advantages, within a given task and for a given purpose. *Thus the basic representation used by DENDRAL, chemical graphs, is, we have noted, the glue that holds the system together and permits various*

qualities of knowledge to combine effectively. However, chemical graphs are manifestly not the appropriate knowledge representation for chess, or speech understanding, or quantum mechanics. Furthermore, even though it is conceivable, though unlikely, that *some* form of graph structure will suffice for encoding all knowledge (just as it is conceivable though unlikely that some linear logical calculus will suffice), it appears that the requirements of any given problem domain are so specialized that the appropriate form of graph will be in turn so specialized as to diminish seriously the importance of whatever insight such a commonality of language might hold. Therefore, knowledge representation takes on a status equal to that of heuristic reasoning in the struggle against combinatorial complexity.

To elaborate: the goal of knowledge engineering is to achieve a productive interaction of knowledge in the service of problem solving. An appropriate knowledge representation is an encoding that relates information that is naturally related in important ways in the referent application area. Further, it ought to do so in ways that ease the burden of inference [33, 34], just as class–subclass inheritance hierarchies can facilitate inference. To the extent that a measure of the inferential burden can be borne by the representation scheme, we have reduced the burden that must be borne by search and generation heuristics [35].

There has been a recent emphasis on "architectures" for general intelligence, as illustrated by [41, 47, 54]. These are attempts to incorporate knowledge representation, acquisition, and use into general problem-solving frameworks. In these systems, as in expert systems research, generality is sought by attempting to devise machines that apply general inferential mechanisms to the specialized knowledge that is acquired and represented.

It is worth noting that none of these proposed architectures could, in any very obvious or direct way, learn to behave as the DENDRAL generator, which is the central feature of the DENDRAL programs. That particular special form of knowledge, an algorithm specifically designed to work on the class of chemical graphs, is indeed specialized knowledge, but it is not itself readily represented as productions, weighted connections, frames, scripts, semantic networks or any of the other non-procedural forms of knowledge representation that are the present currency of artificial intelligence [36].

4.4.1. How much does DENDRAL know?

It is frequently asked *how much* any knowledge-based system such as DENDRAL "knows". Unfortunately there is no straightforward answer to this question, for a variety of reasons. For example, it has frequently been pointed out that humans and, with qualifications, computers know in essence an infinite number of facts (to cite one instance, we know the successor of any integer); and yet our memories are finite. Also, any "piece" of knowledge is meaningful only in a larger context of its use with other knowledge.

Moreover, cognitive capacities can be represented in indefinitely many ways. For example, we may assert that DENDRAL knows that the valence of carbon is 4. Knowing this fact, however, presupposes that the concepts of valence, atom, and a constellation of related concepts are also in some sense understood. (Even so, DENDRAL's concept of valence is clearly not the same as a chemist's, which is embedded in an even richer context of related knowledge.) It is, furthermore, knowledge that is distributed throughout many subprograms that define and manipulate chemical graphs or in some way make use of facts such as "the valence of carbon is 4". It would be nearly impossible to separate those pieces of DENDRAL computer code that in one way or another are associated with an understanding of valence from those that are not.[5]

In analyzing the issue of multiple representations of knowledge, the conventional distinction between "knowing how" and "knowing that", though itself not precise, is a helpful starting point. Even when this division is possible, however, it must be remembered that there exist many different but functionally equivalent, and hence equally "knowledgeable", programs that divide their knowledge in different ways between processes and facts.

For these reasons and more, any list of what DENDRAL knows in terms of concepts, facts, and processes is of limited descriptive power. Nonetheless, we will attempt to classify the content of some of DENDRAL's knowledge of chemistry in the list below.

4.4.1.1. DENDRAL's knowledge of chemical concepts and procedures

DENDRAL's knowledge base of facts and concepts was complete enough for many tasks, including the substantive one of generating all isomeric structures, but was admittedly incomplete. It was, however, intended to be extensible by using rules, lists, and tables, and values of LISP atoms or attributes. For example, only the half-dozen chemical atoms that most frequently occur in organic compounds were known, but any others could easily be added by editing one list of chemical atoms to be considered plus the property list of each new chemical atom. Some of the lists contained names of LISP functions to be used as special-purpose constraints (e.g., to see if a proposed structure violated the principle of stability known as Bredt's rule), or as descriptions of processes that occur in mass spectroscopy (e.g., elimination of water). These were more complex procedures than made sense to decompose into productions, and that were treated as named primitives by chemists anyway.

Perhaps it is obvious, but extensibility was considerably easier in the places where we only needed to edit lists and tables. When new LISP functions

[5] Some of the programs have been more meticulously written in this regard than others. For example, INTSUM always references the valence of chemical atoms through a single function. However, the more general concept of connectivity of graphs, which subsumes valence, is part of the whole framework assumed by almost all functions.

needed to be created to serve as functional primitives we attempted to generalize the functions so they could be used in other ways and to consolidate old functions with new ones to create reusable code. When completely new capabilities were considered, such as generating ringed structures, we often found we needed to rethink the design and implementation of large sections of code.

This obvious principle evolved from the need to develop a computable theory of stability (BADLIST) interactively, and from the difficulties we encountered in changing code to overcome experts' criticisms of the program's performance.

(1) Knowledge of chemical graphs:
- Atom types (C, H, N, O, P, S), along with essential properties of each atom: valence, atomic weight, isotopes, relative abundance of each isotope.
- Bond types (single, double, triple, aromatic).
- *How to:*
  - Generate all acyclic and/or cyclic isomers (including fused rings, spiro forms, etc.).
  - Compute the degree of unsaturation (number of double bonds and rings) from an empirical formula.
  - Compute the mass of a collection of chemical atoms (empirical formula of a molecular fragment) at high or low resolution, plus masses and relative abundance of collections with isotopic contributions.
  - Detect topological symmetry in graphs.
  - Generate all stereoisomers.
  - Find all cycles in a graph.
  - Define and name a new subgraph (with specialized editor).
  - Restrict generation of structures to those containing [GOODLIST] or not containing [BADLIST] named subgraphs.
  - Find all occurrences of arbitrarily complex subgraphs in a graph.
  - Find the greatest common subgraph among a set of graphs.
  - Label nodes and edges of a graph in all distinct ways, prospectively avoiding symmetric labelings.
  - Draw a chemical structure.

(2) Knowledge of chemical stability:
- Twenty classes of unstable families defined as acyclic subgraphs [named on BADLIST], others easily defined and added to BADLIST.
- Three complex constraints: terpene rule, isoprene rule, Bredt's rule.
- *How to:*
  - Recognize keto-enol tautomerism (special form of isomerism) and other tautomers when specified.

(3) Knowledge of mass spectrometry:
- Distinction between low-resolution and high-resolution spectra.
- Distinction between low-voltage and high-voltage spectra.
- *How to:*
  - Digitize an analog mass spectrum.
  - Find meta-stable peaks in a spectrum.
  - Interpret major features of a mass spectrum, using [about a half-dozen] rules that are specific fragmentation patterns for a specific family of compounds—rules already defined for alcohols, ethers, thiols, thioethers, amines, ketones, aromatic acids, estrogenic steroids, androstanes, marine sterols [roughly 50 rules in all are defined].
  - Predict a "complete" spectrum in which every bond of a molecule breaks and every bond of every resulting fragment breaks (recursively), with isotopic contributions of atoms.
  - Constrain the prediction of a "complete" spectrum to plausible breaks, as specified in the half-order theory table [about a dozen named LISP functions in a table], assigning relative measures of significance or likelihood of various processes and assigning placement of charge to one or more resulting fragments from each process.
  - Augment the prediction of a mass spectrum with respect to [about 1–3] rules that are general processes common to every family of compounds—rules already defined for hydrocarbon cleavage, McLafferty rearrangements, elimination of water, carbon monoxide, carbon dioxide, or other user-defined "neutral species" [about a dozen rules defined].

(4) Knowledge of synthetic chemistry:
- *How to:*
  - Define a new chemical reaction for the program to consider [about a dozen already defined with a specialized editor].
  - Reason about plausible biosynthetic pathways from a known starting material in order to restrict a set of candidate structures to those that are plausible products of the starting material.

There are many things that DENDRAL does not know, of course. Among missing items are knowledge of three-dimensional geometry beyond stereoisomerism, polymeric structures, quantum chemistry, and many properties of atoms or structures such as electronegativity, dipole moments, molecular susceptibility, melting points, and crystalline forms.

In addition, all the knowledge of LISP is presupposed by the DENDRAL programs. For example, arithmetic and set-theoretic operations, symbol manipulation, interpretation of complex procedures, and countless bookkeeping

operations. Considerable amounts of code are devoted to keeping track of intermediate results in the overall processing. This "specialized bookkeeping" knowledge is not very profound, yet it is indispensable for the integration of many complex procedures.

Almost all DENDRAL's knowledge is tailored to the task of molecular structure elucidation. Although some pieces such as the representation and graph matcher have found wider applicability, the problem-solving procedures in DENDRAL are still very special-purpose, complex, and voluminous.

4.5. Some observations about discovery in science

We have collected several observations about the process of scientific discovery from this work that may be pertinent to a more systematic examination of a theory of discovery. Since DENDRAL is, at least, an existence proof that the parts of science involving data interpretation and hypothesis formation can be automated, we believe many other problems of scientific interest can be automated along the same lines. Thus we suggest the following generalizations.

(1) *Scientific discovery uses the same basic methods of problem solving as do other scientific reasoning tasks and other forms of problem solving.*

We note that Meta-DENDRAL is not much different in organization from Heuristic DENDRAL. Both are a species of plan–generate–test. Both are guided by a strong model of the domain. It was a surprisingly small step from Heuristic DENDRAL to a rule-discovery program, Meta-DENDRAL. No basic additions or reorganizations of the problem-solving method were required, even though in the latter case the results of problem solving are hypotheses (in the form of productions) that embody a limited scientific theory (concerning the behavior of a class of compounds in the mass spectrometer).

(2) *Scientific discovery is judicious exclusion and selection from a space of possible hypotheses, as achieved through heuristic exploration.*

A cognitive agent has means for generating either the members of a set of possible hypotheses or the states of a problem space. This ability is productive in the sense that human language is productive (as stressed by linguist Noam Chomsky): a large set of novel combinations of a finite set of elements can be generated even though none has been previously encountered in the experience of the scientists. Possible hypotheses that are at odds with firmly believed data may be rejected.

(3) *In exploring the space of possible hypotheses, the scientist is strongly influenced by initial assumptions.*

Heuristic DENDRAL is a theory-driven mechanism: it finds only what it is looking for. For example, if it does not expect to find solutions (organic compounds) containing certain substructures, as signified by the presence of these substructures on BADLIST, then they are not gener-

ated. It is just such biases, (and the more the better), that allow the programs to discover a manageable set of candidates at all.

(4) *Scientific problem solving in general, and discovery in particular, outside the well-codified areas of science, involves the employment of a large number of vague and unverified ideas, rather than the application of logical deduction to previously verified propositions.*

It is clear that Heuristic DENDRAL does not have at its core a formal theory of chemical stability, but rather employs a large collection of weak partial assumptions each of limited range. We propose that this lack of formal theory is the rule in scientific discovery, in contrast to the classical description of scientific method as tight reasoning from established premises. Again, contemporary writers have expressed a similar opinion; what DENDRAL contributes is substantive detail in elaboration of this view.

Note that this observation, and the following one, contrast with the view of scientific discovery in the BACON family of programs [26–28]. In part this is due to DENDRAL's assumption that any piece of empirical data may be flawed, thus no single strict test of agreement (or disagreement) with data will suffice to keep (or reject) an hypothesis. Instead one must rely on as many parts of a partial theory and as many data points as one has available, and then weigh all the evidence.

(5) *Scientific problem solving in general, and discovery in particular, is an interaction of top-down (expectation-driven) exploration, and bottom-up (data-driven) exploration. Both are necessary.*

Heuristic DENDRAL is driven largely in a top-down manner. However, the data (spectra) are employed by MOLION and the PLANNER in planning and by the PREDICTOR to winnow the set of candidate solutions.

(6) *A generator that fails to guarantee completeness is not wholly satisfactory, since one then cannot say with certainty what hypotheses have been excluded from consideration.*

This observation, and the next one, differ from the model of discovery in Lenat's AM and EURISKO programs. There, a generator of plausible hypotheses (conjectures) is driven by a large number of plausible move generators, without any attempt to define the complete space of hypotheses. In domains where a complete generator cannot be defined, for whatever reason, a generator of plausible solutions may be worth trying if some (or any) solution will suffice as an answer.

(7) *A generator that cannot avoid duplicate (or equivalent) expressions of the same hypothesis is not wholly satisfactory, since the generation may never terminate.*

(8) *Knowledge about classes of hypotheses is more effective than knowledge about individual hypotheses.*

(9) *A small set of plausible alternative hypotheses resulting from the generation and testing may be as valuable as a single hypothesis. The upper bound on the acceptable size of the found set varies with problem complexity and with the ease of discriminating among the alternatives by other means.*

Aside from the plausibility of these generalizations that inheres in the existence of successful programs, no specific data support them. They suggest where we might look for support, of course, since we now have them as a source of expectations. There is anecdotal support in abundance. If the normal mode of processing empirical data is to verify expectations, as our theory has it, then science should appear essentially conservative, that is, exhibit few novel theoretical formulations. Kuhn [25] argues this point at length.

Under our analysis the traditional problem of finding an effective method for discovering hypotheses that best explain phenomena has been transformed into finding heuristic methods that generate plausible explanations. The problem of giving rules for producing true scientific statements has been replaced by the problem of finding efficient heuristic rules for culling the reasonable candidates for an explanation from an appropriate set of possible candidates.

4.6. Conclusion

Although its utility to working chemists has been limited, DENDRAL is well known to computational chemists, who have incorporated many of the pieces of DENDRAL in their own software. As a single software package DENDRAL no longer runs; without an enthusiastic user community no one has invested time enough to maintain it.

The major impact of DENDRAL has been on the AI community, where the program is well known and well enough understood to generate some half-truths as well as honest lessons. We take pride in knowing that the lessons learned from DENDRAL are now well entrenched in the AI literature and in the design and implementation of expert systems around the world. We hope this paper will correct some of the half-truths.

Daniel Dennett [14] suggests that natural (or artificial) minds are viewed by various writers in three different ways. Mind as Crystal is the view that cognitive science should be modeled after physics: the principles of AI would be the wave equation and general relativity equation of the mind. Mind as Chaos is the counsel of despair: the mind, though it may be a mechanism, is so complex and ad hoc, and its course so sensitive to initial conditions that it is essentially without discoverable principles. The third view, the one to which our work subscribes, is Mind as Gadget, which sees AI as engineering, and its products as special-purpose artifacts constructed according to good design principles, with abilities constrained by specialized knowledge acquired through specialized experiences. This is engineering not in the sense of

efficiency and cost effectiveness alone, but in the sense of applying good engineering principles to particular tasks-at-hand. It is a view compatible with a world in which minds and brains evolved by an opportunistic, environmentally sensitive process of evolution, rather than springing full blown as the product of either a grand design or a mystical force.

DENDRAL clearly is AI in the spirit of engineering gadget in Dennett's non-pejorative sense; most recent AI systems are as well. While DENDRAL cannot play chess, bake a cake, or diagnose septicemia, it nonetheless embodies general strategies, augmented by specialized knowledge, that give it a measure of intelligence and which can adapt to new information. The general strategies, recombined with other knowledge, have been shown to do well at other tasks. This is the contemporary view of AI, a view that was advanced and illustrated by DENDRAL. It is perhaps the major legacy of this work.

References

[1] R.C. Beavis and B.T. Chait, High-accuracy molecular mass determination of proteins using matrix-assisted laser desorption mass spectrometry, *Anal. Chemistry* **62** (1990) 1836–1840.

[2] H. Brown, L. Hjelmeland and L.M. Masinter, Constructive graph labeling using double cosets, *Discrete Math.* **7** (1974) 1–30.

[3] H. Brown and L.M. Masinter, Algorithm for the construction of the graphs of organic molecules, *Discrete Math.* **8** (1974) 227–244.

[4] B.G. Buchanan, A.M. Duffield and A.V. Robertson, An application of artificial intelligence to the interpretation of mass spectra, in: G.W.A. Milne, ed., *Mass Spectrometry: Techniques and Applications* (Wiley-Interscience, New York, 1971).

[5] B.G. Buchanan and E.A. Feigenbaum, Dendral and Meta-Dendral: their applications dimension, *Artif. Intell.* **11** (1978) 5–24.

[6] B.G. Buchanan and T.M. Mitchell, Model-directed learning of production rules, in: D.A. Waterman and F. Hayes-Roth, eds., *Pattern Directed Inference Systems* (Academic Press, New York, 1978).

[7] B.G. Buchanan and E.H. Shortliffe, eds., *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project* (Addison-Wesley, Reading, MA, 1984).

[8] B.G. Buchanan, G.L. Sutherland and E.A. Feigenbaum, Heuristic DENDRAL: a program for generating explanatory hypotheses in organic chemistry, in: B. Meltzer and D. Michie, eds, *Machine Intelligence* **4** (Edinburgh University Press, Edinburgh, 1969) 209–254.

[9] B.G. Buchanan, G.L. Sutherland and E.A. Feigenbaum, Rediscovering some problems of artificial intelligence in the context of organic chemistry, in: B. Meltzer and D. Michie, eds., *Machine Intelligence* **5** (Edinburgh University Press, Edinburgh, 1970) 253–280.

[10] A. Buchs, A.B. Delfino, A.M. Duffield, C. Djerassi, B.G. Buchanan, E.A. Feigenbaum and J. Lederberg, Applications of "artificial intelligence" for organic chemistry, VI: approach to a general method of interpreting low resolution mass spectra with a computer, *Helvetica Chimica Acta* **53** (1970) 1394–1417.

[11] H. Budzikiewicz, C. Djerassi and D.H. Williams, *Interpretation of Mass Spectra of Organic Compounds* (Holden-Day, San Francisco, CA, 1964).

[12] R.E. Carhart, D.H. Smith, H. Brown and C. Djerassi, Applications of artificial intelligence for chemical inference, XVII: an approach to computer-assisted elucidation of molecular structure, *J. Chemical Soc.* **97** (1975) 5755–5762.

[13] C. Cheer, D.H. Smith, C. Djerassi, B. Tursch, J.C. Braekman and D. Daloze, Applications of artificial intelligence for chemical inference, XXI: the computer-assisted identification of

[+]-palustrol in the marine organism *cespitularia ap.*, aff. *subvirdis*, *Tetrahedron* **32** (1976) 1807.

[14] D.C. Dennett, When philosophers encounter artificial intelligence, *Daedalus* (1988) 283–295.

[15] G.W. Ernst and A. Newell, *GPS: Case Study in Generality and Problem Solving* (Academic Press, New York, 1969).

[16] E.A. Feigenbaum, Artificial intelligence: themes in the second decade, in: A.J.H. Morrell, ed., *Information Processing 68, Proceedings of the IFIP Congress 1968* (North-Holland, Amsterdam, 1968).

[17] E.A. Feigenbaum and R.W. Watson, An initial problem statement for a machine induction research project, Artificial Intelligence Project Memo No. 30, Computer Science Department, Stanford University, Stanford, CA (1965).

[18] A.L. Fella, J.G. Nourse and D.H. Smith, Conformation specification of chemical structures in computer programs, *J. Chemical Inf. Comput. Sci.* **23** (1983) 43–47.

[19] I. Goldstein and S. Papert, Artificial intelligence, language and the study of knowledge, *Cogn. Sci.* **1** (1977) 84–123.

[20] N.A.B. Gray, Applications of artificial intelligence for organic chemistry: analysis of C-13 spectra, *Artificial Intelligence* **22** (1984) 1–21.

[21] F. Hayes-Roth, D.A. Waterman and D.B. Lenat, *Building Expert Systems* (Addison-Wesley, Reading, MA, 1983).

[22] A.C. Hearn, REDUCE-2: a system and language for algebraic manipulation, in: *Proceedings ACM Second Symposium on Symbolic and Algebraic Manipulation*, Los Angeles, CA (1971).

[23] D. Jardetzky, A. Lane, J.-F. Lefevre, E. Lichtarge, B. Hayes-Roth, R. Altman and B.G. Buchanan, A new method for the determination of protein structures in solution from NMR, in: *Proceedings Twenty-Third Congress Ampere*, Rome (1986).

[24] P. Karp, Hypothesis formation and qualitative reasoning in molecular biology, Ph.D. Dissertation, Computer Science Department, Stanford University, Stanford, CA (1989).

[25] T.S. Kuhn, *The Structure of Scientific Revolutions* (University of Chicago Press, Chicago, IL, 1962).

[26] P. Langley, BACON-1: a general discovery system, in: *Proceedings Second National Conference of the Canadian Society for Computational Studies in Intelligence*, Toronto, Ont. (1978) 173–180.

[27] P. Langley, G.L. Bradshaw and H.A. Simon, BACON-5: discovery of conservation laws, in: *Proceedings IJCAI-81*, Vancouver, BC (1981).

[28] P. Langley, G.L. Bradshaw and H.A. Simon, Rediscovering chemistry with the BACON system, in: R.S. Michalski, J.G. Carbonell and T.M. Mitchell, eds., *Machine Learning: An Artificial Intelligence Approach* **1** (Tioga, Palo Alto, CA, 1983) 307–329.

[29] J. Lederberg, DENDRAL-64: a system for computer construction, enumeration and notation of organic molecules as tree structures and cyclic graphs, Part I: notational algorithm for tree structures, Report No. CR-57029 and STAR No. N65-13158, National Aeronautics and Space Administration (1964).

[30] J. Lederberg, DENDRAL-64: a system for computer construction, enumeration and notation of organic molecules as tree structures and cyclic graphs, Part II: topology of cyclic graphs, Report No. CR-68898 and STAR No. N66-14074, National Aeronautics and Space Administration (1965).

[31] J. Lederberg, Systematics of organic molecules, graph topology and Hamilton circuits: a general outline of the DENDRAL system, Report No. CR-68899 and STAR No. N66-14075, National Aeronautics and Space Administration (1966).

[32] D.B. Lenat and E.A. Feigenbaum, On the thresholds of knowledge, in: *Proceedings IJCAI-87*, Milan (1987) 1173–1182; also: *Artif. Intell.* **47** (1991) 185–250.

[33] R.K. Lindsay, Toward the development of a machine which comprehends, Doctoral Dissertation, Carnegie-Mellon University, Pittsburgh, PA (1961).

[34] R.K. Lindsay, In defense of *ad hoc* systems, in: R. Schank and K. Colby, eds., *Computer Models of Thought and Language* (Freeman, San Francisco, CA, 1973) 372–395.

[35] R.K. Lindsay, Images and inference, *Cognition* **29** (1988) 229–250.

[36] R.K. Lindsay, Symbol-processing theories and the SOAR architecture, *Psychol. Sci.* **2** (1991) 294–302.

[37] R.K. Lindsay, B.G. Buchanan, E.A. Feigenbaum and J. Lederberg, *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project* (McGraw-Hill, New York, 1980).
[38] W.A. Martin and R.J. Fateman, The MACSYMA system, in: S.R. Petrick, ed., *Proceedings ACM Second Symposium on Symbolic and Algebraic Manipulation*, Los Angeles, CA (1971).
[39] J. McCarthy, Programs with common sense, in: M. Minsky, ed., *Semantic Information Processing* (MIT Press, Cambridge, MA, 1968) 403–418.
[40] F.W. McLafferty, *Interpretation of Mass Spectra* (Benjamin, New York, 1966).
[41] A. Newell, *Unified Theories of Cognition* (Harvard University Press, Cambridge, MA, 1990).
[42] J.G. Nourse, Generalized stereoisomerization modes, *J. Am. Chemical Soc.* **99** (1977) 2063.
[43] J.G. Nourse, Application of the permutation group to stereoisomer generation for computer assisted structure elucidation, in: J. Hinze, ed., *Lecture Notes in Chemistry* **12** (Springer, New York, 1979) 19.
[44] J.G. Nourse, R.E. Carhart, D.H. Smith and C. Djerassi, Applications of artificial intelligence to chemical inference, 29: exhaustive generation of stereoisomers for structure elucidation, *J. Am. Chemical Soc.* **101** (1979) 1216.
[45] B. Raphael, Robot research at Stanford Research Institute, Artificial Intelligence Center Technical Note 64, SRI Report 1530, Stanford Research Institute, Menlo Park, CA (1972).
[46] J.A. Robinson, a machine-oriented logic based on the resolution principle, *J. ACM* **12** (1965) 23–41.
[47] D.E. Rumelhart J. L. McClelland and the PDP Research Group, eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (MIT Press, Cambridge, MA, 1986).
[48] H.A. Simon and M. Barenfeld, Information-processing analysis of perceptual processes in problem solving, *Psychol. Rev.* **76** (1969) 473–483.
[49] D.H. Smith, B.G. Buchanan, R.S. Engelmore, H. Adlercreutz and C. Djerassi, Applications of artificial intelligence for chemical inference, IX: analysis of mixtures without prior separation as illustrated for estrogens, *J. Am. Chemical Soc.* **95** (1973) 6078–6084.
[50] D.H. Smith, B.G. Buchanan, R.S. Engelmore, A.M. Duffield, A. Yeo, E.A. Feigenbaum, J. Lederberg and C. Djerassi, Applications of artificial intelligence for chemical inference, VIII: an approach to the computer interpretation of the high resolution mass spectra of complex molecules. Structure elucidation of estrogenic steroids, *J. Am. Chemical Soc.* **94** (1972) 5962–5971.
[51] D.H. Smith, B.G. Buchanan, W.C. White, E.A. Feigenbaum, J. Lederberg and C. Djerassi, Applications of artificial intelligence for chemical inference, X: Intsum. A data interpretation and summary program applied to the collected mass spectra of estrogenic steroids, *Tetrahedron* **29** (1973) 3117–3134.
[52] D.H. Smith and R.E. Carhart, Applications of artificial intelligence for chemical inference, XXIV: structural isomerism of mono- and sesquiterpenoid skeletons, *Tetrahedron* **32** (1976) 2513.
[53] D.H. Smith, J.P. Konopelski and C. Djerassi, Applications of artificial intelligence for chemical inference, XIX: computer generation of ion structures, *Organic Mass Spectrometry* **11** (1976) 86–100.
[54] K. VanLehn, ed., *Architectures for Intelligence. The Twenty-Second Carnegie-Mellon Symposium on Cognition* (Lawrence Erlbaum, Hillsdale, NJ, 1991).
[55] T.H. Varkony, R.E. Carhart and D.H. Smith, Applications of artificial intelligence for chemical inference, XXIII: computer-assisted structure elucidation. Modelling chemical reaction sequences used in molecular structure problems, in: W.T. Wipke, ed., *Computer-Assisted Organic Synthesis* (American Chemical Society, Washington, DC, 1976).
[56] T.H. Varkony, Y. Shiloach and D.H. Smith, Computer-assisted examination of chemical compounds for structural similarities, *J. Chemical Inf. Comput. Sci.* **19** (1979) 104–111.