12

HEURISTIC DENDRAL : a Program for Generating Explanatory Hypotheses in Organic Chemistry

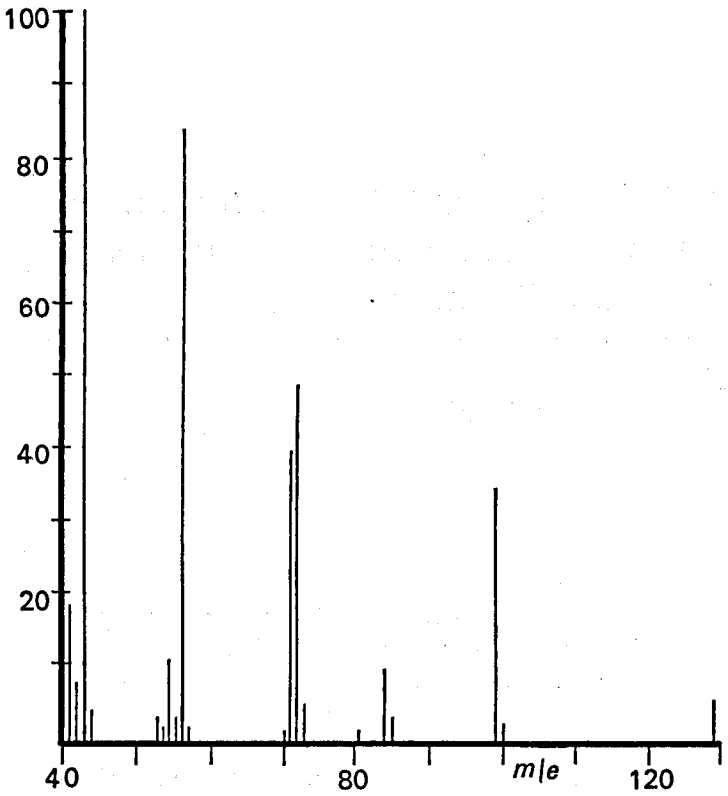
B. Buchanan
Georgia Sutherland and
E. A. Feigenbaum
Computer Science Department, Stanford University

A computer program has been written which can formulate hypotheses from a given set of scientific data. The data consist of the mass spectrum and the empirical formula of an organic chemical compound. The hypotheses which are produced describe molecular structures which are plausible explanations of the data. The hypotheses are generated systematically within the program's theory of chemical stability and within limiting constraints which are inferred from the data by heuristic rules. The program excludes hypotheses inconsistent with the data and lists its candidate explanatory hypotheses in order of decreasing plausibility. The computer program is heuristic in that it searches for plausible hypotheses in a small subset of the total hypothesis space according to heuristic rules learned from chemists.

INTRODUCTION

The computer program described below resulted from an interest in studying scientific hypothesis formation as a decision-making process. To make progress on this broad and general problem, it seemed useful to choose a particular scientific task involving inductive behaviour and to explore it in as much detail as possible. The task chosen is in a well defined but relatively new and complex area of organic chemistry, namely the analysis of mass spectra of organic molecules. HEURISTIC DENDRAL is a computer program which generates molecular 'graphs' (i.e., structures) as hypotheses to explain the data produced by a mass spectrometer.

The data produced when a mass spectrometer fragments molecules of a chemical sample can be interpreted as a list of masses of fragments paired



Composition: $C_8H_{16}O$

Molecular Structure: $CH_3—CH_2—\overset{\overset{O}{\|}}{C}—CH_2—CH_2—CH_2—CH_2—CH_3$

Computer Representation of the Mass Spectrum:

((41 . 18)(42 . 7)(43 . 100)(44 . 3)
(53 . 3)(54 . 1)(55 . 11)(56 . 3)
(57 . 80)(58 . 2)(70 . 1)(71 . 36)
(72 . 44)(73 . 5)(81 . 1)(85 . 6)
(86 . 2)(99 . 31)(100 . 2)(128 . 5))

Figure 1. The mass spectrum for 3-OCTANONE

with their relative abundances. An example of a mass spectrum is shown in figure 1. By studying the resulting list of number pairs, chemists can infer the molecular structure of the chemical sample. Some of the decision processes which chemists use in making such inferences are incorporated into a computer program along with a structure-generating algorithm which provides a systematic approach to the problem of deducing the structure of a chemical sample. The computer program is HEURISTIC DENDRAL; and it is now

capable of making inferences from mass spectra to molecular structures in a restricted domain.

The foundation for the HEURISTIC DENDRAL program is Lederberg's (1964) DENDRAL Algorithm (section 7 contains a summary of this algorithm). The algorithm gives a way of representing and ordering chemical molecules uniquely; thus it gives a method for generating all topologically possible molecules of a given composition without redundancy. It is a systematic and exhaustive topologist which can generate all non-cyclical graphs that can be made with the atoms of the composition, knowing no chemistry other than the valences of these atoms. The DENDRAL algorithm defines the hypothesis space in much the same way as a legal move generator for a chess-playing program defines the total move space within which good chess moves will be sought.

The computer program is written in the LISP language on the PDP-6 computer at the Stanford University Artificial Intelligence Laboratory. It occupies approximately eighty thousand words of memory. Working with Professor Joshua Lederberg at Stanford, William Weiher and William White developed the basic representation and wrote the initial program. The program has also benefited greatly from the attention of members of the Stanford Mass Spectrometry Laboratory: Professors Carl Djerassi, Alex Robertson, Jerry Meinwald, and especially Dr Alan Duffield.

The program itself is segmented into five subprograms: the PRELIMINARY INFERENCE MAKER, the DATA ADJUSTOR, the STRUCTURE GENERATOR, the PREDICTOR, and the EVALUATION FUNCTION. The interrelation of these subprograms is shown in figure 2.

The PRELIMINARY INFERENCE MAKER (described in section 1) examines a spectrum and determines what general classes of chemical substructures are confirmed or disconfirmed by the data. All hypothesized structures generated later by HEURISTIC DENDRAL must contain all the indicated substructures (all of which are put on a list called GOODLIST); and no structure may contain any substructure which is disconfirmed by the spectrum. (All the forbidden substructures are put on a list called BADLIST.)

The DATA ADJUSTOR subprogram (described in section 2) chooses significant spectral peaks for the STRUCTURE GENERATOR to use for its Zero Order Theory. At present there are four independent ways of interpreting the spectrum.

The STRUCTURE GENERATOR (see section 3) uses the information deduced by the PRELIMINARY INFERENCE MAKER and the DATA ADJUSTOR to produce a list of all topologically possible chemical structures which are consistent with the spectrum. The consistency criteria are the lists of 'good' and 'bad' substructures and the Zero-Order Spectral Theory, described in detail in section 3.2.

The PREDICTOR subprogram is a rough model of a mass spectrometer (see section 4). It is used to predict significant features of the mass spectrum

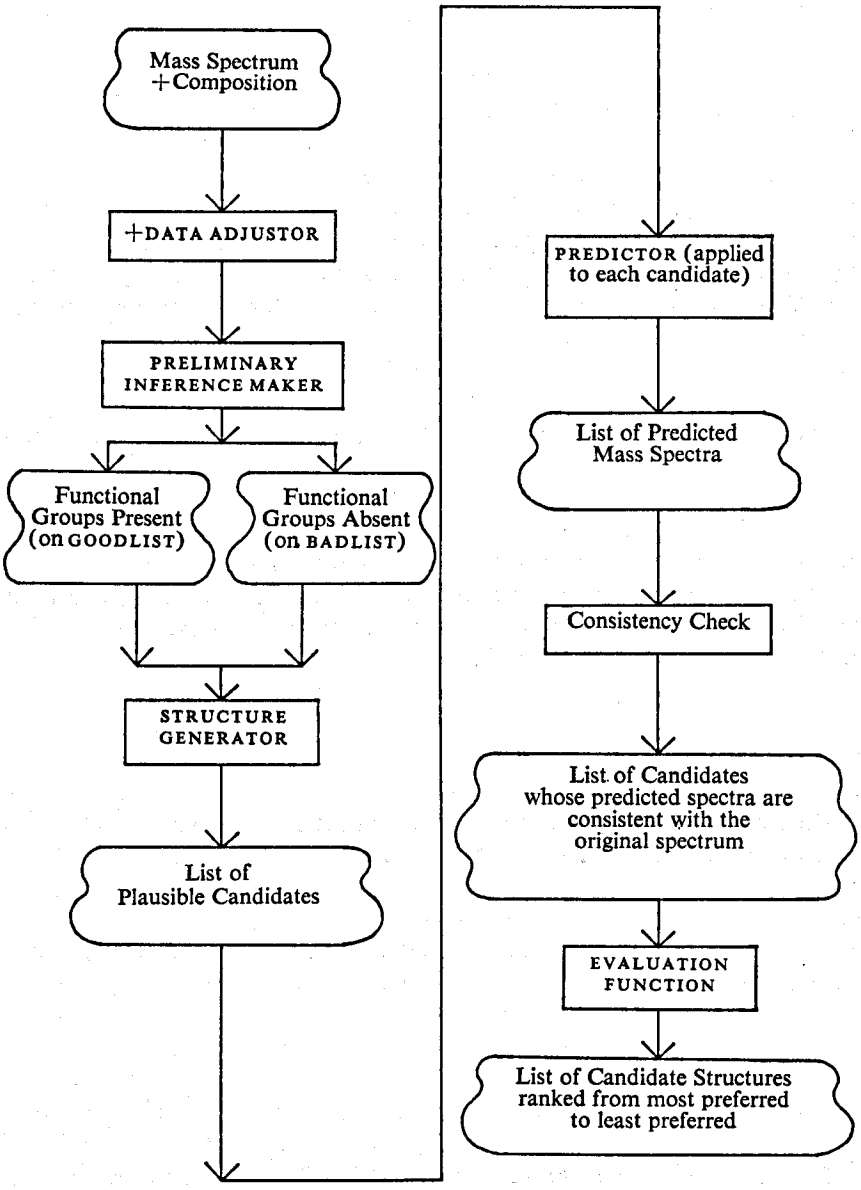


Figure 2. General design of HEURISTIC DENDRAL

corresponding to each candidate structure output by the STRUCTURE GENERATOR.

The EVALUATION FUNCTION (see section 5) compares each predicted spectrum against the original spectral data and assigns a score representing similarity of the two spectra. This enables the candidate hypotheses output by the STRUCTURE GENERATOR to be listed in order of their 'plausibility' or estimated degree of confirmation.

An ideal program for deducing the structure of a chemical sample would output exactly one structure as *the* explanation for the spectral data. Up to now the usual case has been that several different structures are suggested as plausible explanations for the data. However, even a short list is a far better result than was obtained by the original program, which listed all the topologically possible structures and made no use of any real data at all.

Because the constraints which have been included in the program to limit the search space are heuristic, nothing guarantees that the correct structure will not be bypassed. When a test run does fail, however, the program is modified after our chemist-informants study the output and analyze their own decision procedures. The purpose of this report is merely to describe the current state of HEURISTIC DENDRAL and to sketch some of our plans for future program developments.

1. THE PRELIMINARY INFERENCE MAKER

The PRELIMINARY INFERENCE MAKER is conceptually very simple: it looks for the presence or absence of sets of peaks in a mass spectrum and updates GOODLIST or BADLIST, thus constraining HEURISTIC DENDRAL from generating large numbers of molecular structures as possible explanations of a given mass spectrum. By looking for patterns of peaks in the spectrum which are characteristic of some structural fragment, such as the keto group, this preliminary program can tell the STRUCTURE GENERATOR to concentrate on some fragments and to avoid others. It does this by temporarily putting desirable structures on GOODLIST (see section 3.5) and undesirable ones on BADLIST (see Section 3.3).

The program has access to translations of Tables 1 and 2. As Table 1 indicates implicitly, this program knows the name, structure, valence, valence locations, empirical formula, and symmetrical atoms, as well as some characteristic peaks for several functional groups. It also recognizes priorities of groups, as Table 2 indicates. Addition of new information is simplified by a short routine (QUEST) which asks the chemist at the console for the essential information – and explains what it wants if he does not understand. An example is shown in Table 3.

In this example the function QUEST is called to prompt information about identifying a new group, in this case the ester group. The lines preceded by asterisks are messages from the machine. Lines following a machine prompt (i.e., after a colon) were typed in from the console. This dialogue is much

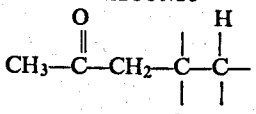
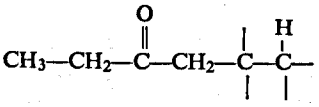
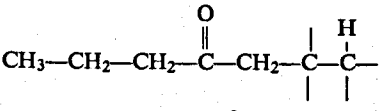
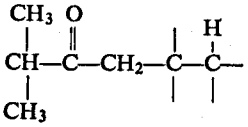
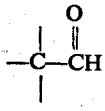
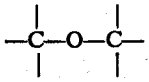
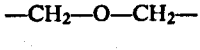
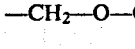
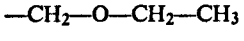
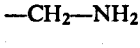
functional group and characteristic subgraph	*identifying conditions*
A. KETONE 	1. There are 2 peaks at mass units $x1$ & $x2$ such that a. $x1+x2=M+28$ b. $x1-28$ is a high peak c. $x2-28$ is a high peak d. At least one of $x1$ or $x2$ is high
B. METHYL-KETONE3 	1. Ketone conditions are satisfied 2. 43 is a high peak 3. 58 is a high peak 4. $M-43$ is a low peak 5. $M-15$ is low or possibly zero
C. ETHYL-KETONE3 	1. Ketone conditions are satisfied 2. 57 is a high peak 3. 72 is a high peak 4. $M-29$ is a high peak 5. $M-57$ is a high peak
D. N-PROPYL-KETONE3 	1. 71 is a high peak 2. 43 is a high peak 3. 86 is a high peak 4. 58 appears with any intensity
E. ISO-PROPYL-KETONE3 	1. 71 is a high peak 2. 43 is a high peak 3. 86 is a high peak 4. There is no peak at 58
F. ALDEHYDE 	1. $M-44$ is a high peak 2. 44 is a high peak
G. ETHER 	1. $M-17$ is absent 2. $M-18$ is absent
H. ETHER2 	1. Ether conditions are satisfied 2. There are 2 peaks at $x1$ & $x2$ such that a. $x1+x2=M+44$ b. At least one of $x1$ or $x2$ is high
I. METHYL-ETHER2 	1. Ether2 conditions are satisfied 2. 45 is a high peak 3. $M-15$ is low or possibly zero 4. $M-1$ appears (any intensity)
J. ETHYL-ETHER2 	1. Ether2 conditions are satisfied 2. 59 is a high peak 3. $M-15$ appears (any intensity)
K. PRIMARY-AMINE2 	1. 30 is a high peak 2. No other peak is high

Table 1. Important chemical groups and their identifying conditions

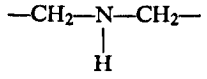
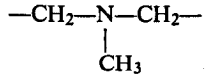
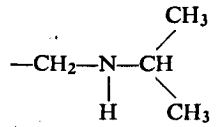
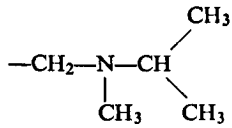
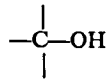
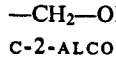
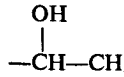
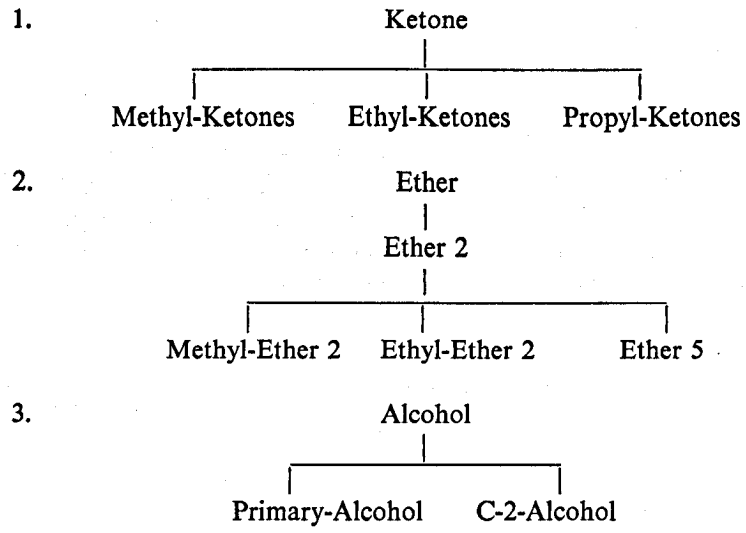
functional group and characteristic subgraph	*identifying conditions*
L. SECONDARY-AMINE2 	1. There are 2 peaks at $x1$ & $x2$ such that a. $x1+x2=M+43$ b. At least one of $x1$ or $x2$ is high 2. 30 is a high peak
M. TERTIARY-AMINE2 	1. There are 2 peaks at $x1$ & $x2$ such that a. $x1+x2=M+71$ b. At least one of $x1$ or $x2$ is high 2. 44 is a high peak
N. ISOPROPYL-2ARY-AMINE2 	1. 44 is a high peak 2. 72 is a high peak 3. $M-15$ is a high peak
O. ISOPROPYL-3ARY-AMINE2 	1. 58 is a high peak 2. 86 is a high peak 3. $M-15$ is a high peak
P. ALCOHOL 	1. M is low or possibly zero 2. Either $M-18$ or $M-17$ appears (any intensity) 3. $M-46$ appears (any intensity)
Q. PRIMARY-ALCOHOL 	1. Alcohol conditions are satisfied 2. The 31 peak is approximately 10%
R. C-2-ALCOHOL 	1. Alcohol conditions are satisfied 2. 45 is a high peak

Table 1. (contd.)

A. Family Trees: priorities within families



B. Hierarchies of families

Ketone > Ether

(i.e., check Ethers only if the Ketone tests fail).

Table 2. Family priorities

(QUEST)
*THIS PGM REQUESTS INFORMATION TO ALLOW THE INFERENCE MAKER TO PUT RADICALS ON GOODLIST OR BADLIST ON THE STRENGTH OF THE APPEARANCE OR NON-APPEARANCE OF A FEW SPECTRAL LINES. IF YOU DO NOT KNOW THE PROPER FORM FOR YOUR ANSWER TO ANY REQUEST TYPE A QUESTION MARK. IF YOU MAKE A MISTAKE IN A LINE YOU CAN CORRECT IT WHEN QUEST IS DONE BY CALLING THE FUNCTION "CHGPROPS" OF ONE ARGUMENT, THE GROUP NAME. E.G., (CHGPROPS (QUOTE KETONE))

*NAME OF FUNCTIONAL GROUP OR RADICAL:
?
*ANY ATOMIC NAME WILL SUFFICE:
ESTER

*PEAKS WHOSE ABSENCE INDICATES THE ABSENCE OF THIS GROUP:
?*THE PGM WANTS A LIST OF DOTTED PAIRS INDICATING MASS-INTENSITY PAIRS IT SHOULD LOOK FOR. MASS UNITS MAY BE SPECIFIED AS
1. A NUMBER,
2. M (THE MOLECULAR WT), OR
3. A LIST OF THREE ELEMENTS:
   3.1 THE LETTER M ,
   3.2 SLASH AND MINUS (OR PLUS) SIGN
   3.3 A NUMBER (TO BE SUBTRACTED OR ADDED TO THE M).

Table 3. Conveying information to the PRELIMINARY INFERENCE MAKER

```
INTENSITY UNITS MAY BE SPECIFIED AS
1. A NUMBER BETWEEN Ø AND 1ØØ INCLUSIVE,
2. THE WORD "ANY" (ANY INTENSITY ABOVE Ø),
3. THE WORD "LOW" (INTEN BETWEEN Ø AND 5 INCL),
4. THE WORD "HIGH" (INTEN BTW 11 AND 1ØØ INCL), OR
5. THE WORD "POSSØ " (INTEN LOW OR ZERO)
FOR EXAMPLE THIS LIST WOULD BE ACCEPTABLE:
((45 . HIGH) ((M / -45) . Ø) (M . LOW))

*PEAKS:
((45 . Ø) (6Ø . Ø))
*PEAKS SUFFICIENT TO INDICATE THIS GROUP:
?*TYPE "SAME" IF THE NECESSARY
CONDITIONS ARE ALSO SUFFICIENT, "NIL" IF THERE ARE
NO SUFF CONDITNS, OR "??" IF YOU WANT THE
CONDITION LIST EXPLAINED:
SAME
*STRUCTURE IN LIST NOTATION:? *TYPE A LIST WHOSE FIRST ELMT IS 1 AND
INDICATE ALL HYDROGEN ATOMS. E.G., (1 C (1 H) (1 H) (1 H)) FOR THE METHYL
RADICAL.
*STRUCTURE: (1 C(2 0) (1 0(1 C)))
*VALENCE:? *IF ALL FREE BONDS ARE ON ONE ATOM, TYPE THIS NUMBER. OTHER-
WISE TYPE A LIST OF NUMBERS INDICATING HOW THE FREE BONDS ARE SPLIT UP,
READING THE STRUCTURE YOU TYPED FROM LEFT TO RIGHT, IGNORING ATOMS
WITH NO FREE VALENCE: (1 2)
*LIST OF SYMMETRIES:?
*TYPE A LIST OF THE FORM
  ((3 2 1)) TO INDICATE THAT THE FIRST & THIRD ATOMS WITH FREE VALENCE
ARE SYMMETRICAL:()

*PLACE OF GROUP IN ITS FAMILY:? *TYPE A LIST CONSISTING OF (1) THE NAME
OF THE NEXT HIGHER FAMILY MEMBER OR NIL
(2) THIS GROUP NAME
(3), (4), . . . (N) NAMES OF ALL NEXT
LOWER MEMBERS. E.G., FOR ETHER2:
  (ETHER ETHER2 METHYL-ETHER2 ETHYL-ETHER2) *PLACE OF GROUP IN ITS
FAMILY:NIL
*THANKS CALL AGAIN

(CHGROPS (QUOTE ESTER))
*PROPERTY TO CHANGE:?
*PROPERTY NAME     *DESCRIPTION
(TYPE ONE)
  NESS       LIST OF PEAKS WHOSE ABSENCE INDICATES ABSENCE OF GROUP
  SUFF       LIST OF PEAKS INDICATING THE PRESENCE OF THIS GROUP
  FORM       EMPIRICAL FORMULA AS A LIST OF DOTTED PAIRS
  VALENCE    A SINGLE NUMBER OR A VECTOR
  STRUCT     STRUCTURE IN LIST NOTATION
  SYM        LIST OF SYMMETRIES
*PROPERTY: VALENCE
*VALUE: (1 3)
*REPLACE (R) OR ADD (A)? R
(1 3)
*PROPERTY TO CHANGE: NIL
DONE
```

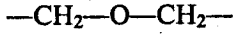
Table 3. (contd.)

shorter when the user knows the correct form for his response (and does not type a question mark). If the user calls QUEST1 instead of QUEST, the machine begins with the first prompt, bypassing the initial descriptive sentences. If the user wishes to change any information typed in previously, he calls the function CHGPROPS.

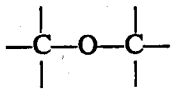
The PRELIMINARY INFERENCE MAKER is given as input the spectrum, the empirical formula of the molecule, and a noise threshold to apply to the spectrum.[1] The LISP function INFER accepts this information and controls the subsequent inferences about the presence or absence of the structural groups. The program performs the following three tests for each structure:

1. Is the empirical formula of the structure compatible with the empirical formula of the molecule? If not, get the next structure.
2. Is any necessary condition falsified by the spectrum? If so, put this structure on BADLIST and get the next structure.
3. Are all sufficient conditions satisfied by the data? If so, put this structure on GOODLIST and get the next structure. Note: at present all sets of conditions are both necessary and sufficient.

The Family Trees shown in Table 2 reduce the effort of the PRELIMINARY INFERENCE MAKER and eliminate redundant effort in the STRUCTURE GENERATOR. When the spectral data indicate that a group is absent from the structure (resulting in the addition of this group to BADLIST), no lower members of the same family are even checked. On the other hand, if both a higher and a lower member of a family are indicated by the data (resulting in the addition of both groups to GOODLIST), only the lower, more specific, group is used. For example, if both of the subgraphs named ETHER and ETHER2 are on GOODLIST, the program deletes the more general one, ETHER, since ETHER2 constrains structure generation more; that is, there are fewer isomers of a given composition containing



than there are which contain

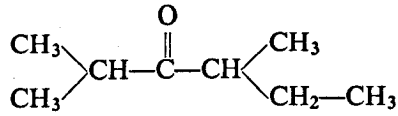


The family hierarchy list also reduces the effort of this program. If *any* member of the first family is on GOODLIST, no members of the second family are even checked.

In the cases of the general Ketone, Ether2, Secondary-Amine2, and Tertiary-Amine2 subgraphs, the preliminary inference maker can, in fact, isolate the position of the functional group as well as determine which

[1] Spectral peaks are deleted if their amplitudes are lower than the threshold. This option has not yet been exercised since it may confuse the inference maker. A threshold value of 1 bypasses this option.

functional group is present. It can do this because of highly favourable alpha-cleavage (cleavage of the bond between the carbon atom attached to the heteroatom and the rest of the molecule) which is an identifying condition for each of these subgraphs. For example, in the ketone shown below, the program can tell that the keto radical (C=O) is between some C_3H_7 structure and some C_4H_9 structure, even though it cannot specify terminal radicals uniquely.



This positional information is passed to the STRUCTURE GENERATOR's partitioning routine which is discussed in section 3.4. The effect in this case is that the only ketones which will be generated are those with the keto group bounded by three carbon atoms on one side and four carbon atoms on the other.

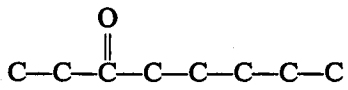
```
(INFER (QUOTE C8H16O) S:09046 1)
*GOODLIST=(*ETHYL-KETONE3*)
*BADLIST=(*C-2-ALCOHOL* *PRIMARY-ALCOHOL* *ETHYL-ETHER2* *METHYL-
ETHER2* *ETHER2* *ALDEHYDE* *ALCOHOL* *ISO-PROPYL-KETONE3* *N-PROPYL-
KETONE3* *METHYL-KETONE3*)
------------------------------------------------------------
(JULY-4-1968 VERSION)
C2*ETHYL-KETONE3*H8
MOLECULES  NO DOUBLE BOND EQUIVS
  1. CH2..  CH2.C3H7  C=.O C2H5,
  2. CH2..  CH..CH3 C2H5  C=.O C2H5,
  3. CH2..  CH2.CH..CH3 CH3  C=.O C2H5,

DONE
S:09046
((41..18.)(42..7.)(43..100.)(44..3.)(53..3.)(54..1.)
(55..11.)(56..3.)(57..80.)(58..2.)(70..1.)(71..36.)
(72..44.)(73..5.)(81..1.)(85..6.)(86..2.)(99..31.)
(100..2.)(128..5.))
```

Table 4. Example from the PRELIMINARY INFERENCE MAKER

Although the chemical heuristics used in this program are more like suggestions than rules, they have demonstrated their usefulness in a number of trials. The results of one of these trials appear in Table 4. The dashed line separates the lines printed by the PRELIMINARY INFERENCE MAKER from the lines printed by the STRUCTURE GENERATOR. The complete output for this example is discussed in detail in section 6. In this case, total output is reduced from 698 isomers[1] to 3 isomers as a result of applying the PRELIMINARY INFERENCE MAKER.

[1] The number of chemically stable acyclic structures with empirical formula $C_8H_{16}O$ is 698; the total number of topologically possible graphs which satisfy just the valence restrictions is 790. Section 3 discusses the program which generates these structures.

The program was given the mass spectrum and empirical formula of 3-octanone. S:09046 is the mass spectrum for the structure:



The first output structure is the correct structure for this spectrum. The rest of the output structures are other ethyl-ketones, because of GOODLIST.

2. THE DATA ADJUSTOR

The DATA ADJUSTOR subprogram determines which mass points of a real spectrum are significant enough to be used by later programs. This process is separable into three steps:

1. Determine the mass of the molecular ion (M). If this number is not in the real spectrum, insert it with large amplitude.
2. Delete peaks at impossible mass points. Specifically, delete peaks at $1, 2, \ldots, 10, 11, 19, \ldots, 23, M-1, M-2, \ldots, M-23$.
3. Delete all but the most significant peaks. Significance has to be decided without knowledge of the molecular structure of the sample producing the spectrum. Four methods of determining significance are included at present, with the choice of method being left to the program user.
   (i) The Threshold Method selects those mass points which have amplitudes higher than a certain number.
   (ii) The Biemann Method selects the two mass numbers with highest amplitudes in each interval of 14 mass numbers.
   (iii) The Lederberg Method selects the n mass numbers with highest amplitudes. The number n depends upon the number of atoms in the chemical composition.
   $$n=\frac{12(count-1)}{5}-1$$
   (iv) The fourth method allows the user to specify the number of mass points to be used (the n highest peaks).
   Each of these four methods reduces the real spectrum to a set of mass numbers judged to be the 'significant peaks' in the data. This revised spectrum is then given to the STRUCTURE GENERATOR which treats it as the data to guide the process of generating structures.

The data-adjusting routine is invoked by calling the function REALSPEC with three arguments. The first two arguments specify the composition and the spectrum. The third argument indicates which method to use. The four possibilities for the third argument are:

(i) $-m$ – use threshold m
(ii) T – use Bieman's method
(iii) NIL – use Lederberg's method
(iv) n – take the n highest peaks.

The relative merit of these methods has not been determined. In the examples processed so far, it appears that the STRUCTURE GENERATOR needs only a few of the mass points in a typical spectrum.

3. THE STRUCTURE GENERATOR

The DENDRAL Algorithm described in section 7 is a procedure for generating all of the topologically possible acyclic structures (isomers) of a chemical composition. This algorithm is based on a canonical notation for chemical structures and an ordering procedure which determines which of two canonical structures is 'higher'.

The STRUCTURE GENERATOR is a computer program implementing the DENDRAL algorithm but with the inclusion of heuristic constraints to prevent the program from generating structures which are incompatible with chemical theory or mass spectral data. Applying these constraints in the course of structure generation greatly increases the efficiency of the program, decreasing amount of output and total run time by several orders of magnitude.

The STRUCTURE GENERATOR is designed to solve the following problem:

GIVEN: a list of defined atoms with their valences and weights
a composition (empirical formula)
a spectrum (mass numbers only)
a list of likely substructures
a list of impossible substructures

TASK: generate all structures compatible with the given data. If there are no data-oriented lists of likely or impossible substructures and no spectral data, the program generates all structural variants (isomers) of the given composition.

INSTRUCTIONS:

1. Make certain that the composition is compatible with the spectrum, if there is one.
2. Consider only those structures which have exactly the types and amounts of atoms specified by the composition.
3. If certain substructures are required, remove their atoms from the composition and insert the name of a 'superatom' to represent that substructure. Be sure the superatom substructure is compatible with the spectrum. After generating a structure containing superatoms, translate superatoms into the original substructures before printing the output.
4. If the partitioning option is to be exercised, consider all subgroupings (partitions) of the composition. Determine whether a given partition is 'plausible'. Generate only those structures which come from plausible partitions.
5. Generate substructures to combine into isomers. The isomers must contain no forbidden substructures; and each substructure must be compatible with the spectrum, if there is one.

6. Provide the user of the program periodic opportunities to observe and change the direction of structure generation. (Optional)
7. Remember past work. (Optional)

The mechanisms for following these instructions are described in the following sections.

3.1. Brief description of the structure-generating algorithm

The basic steps for generating chemical structures are to generate radicals (structures with a free bond) and to connect radicals to make larger structures. Radicals are generated recursively from a composition list of atoms by deciding on the first atom (apical node) and free bond (afferent link) and then making one or more radicals out of the remaining composition. The function GENRAD[1] constructs a single radical by this method; MAKERADS constructs two, three, or four radicals from a single composition; and GENMOL determines the center of a molecular structure and causes two or more radicals to be constructed to attach to the center.

The function UPRAD takes a radical and returns the next higher radical which can be made from the same elements. UPMOL does the same for molecules. The function ISOMERS causes all the structures for a given composition to be generated and printed in ascending canonical order.

The program's constraints are controlled by a number of switches (global variables) which are pre-set before calling ISOMERS. The switches are named: SPECTRUM, GOODLIST, BADLIST, NOPARTS, DIALOG, DICTSWITCH, and OUTCONTROL. Individual constraints may be bypassed at the discretion of the user of the program. When all constraints are turned off, the STRUCTURE GENERATOR becomes a routine graph maker, generating an exhaustive list of all possible acyclic graph structures of n nodes, where different nodes may have different numbers of links (valence). The switch settings for unconstrained program operation are:

```
(SETQ SPECTRUM NIL)
(SETQ GOODLIST NIL)
(SETQ BADLIST NIL)
(SETQ NOPARTS T)
(SETQ DIALOG (QUOTE OFF))
(SETQ DICTSWITCH (QUOTE OFF))
(SETQ OUTCONTROL (QUOTE OFF))
```

[1] The LISP functions which perform certain operations will be identified in this report. To simplify the discussion, however, their arguments and operation will not be discussed. A separate paper lists all LISP functions contained in the STRUCTURE GENERATOR and outlines their use.

3.2. The SPECTRUM and the Zero-Order Theory of Mass Spectrometry

The SPECTRUM of the STRUCTURE GENERATOR is a single list of numbers, corresponding to significant mass numbers in the real spectrum. The DATA ADJUSTOR sub-program provides the STRUCTURE GENERATOR with this list of numbers, all of which have equal importance as far as the STRUCTURE GENERATOR is concerned.

The SPECTRUM is consulted to confirm the presence of compositions and radicals. The first reference to SPECTRUM is by the function ISOMERS which must make certain that the mass of the input composition is present. If it is not, then no structures can be generated for that composition. Any smaller composition can be made into structures if it is not inconsistent with the Zero-Order Theory described below. This constrains the program to consider only those sub-compositions which have some promise of leading to structures compatible with the SPECTRUM.

The Zero-Order Theory assumes that every bond of a structure to which it applies will break (one bond at a time) and that at least one of each pair of substructures obtained from a single break will contribute its mass to the spectrum. The Zero-Order Theory does not apply to double bonds, triple bonds, or bonds leading to certain small structures. That is, in order for a structure to be consistent with the Zero-Order Theory, at least one of the following conditions must be met;

1. The structure contains exactly one atom other than hydrogen.
2. The afferent link is greater than 1.
3. The mass of the structure is less than 30.
4. The mass of the structure is in the SPECTRUM list.
5. The complement mass of the structure is in the SPECTRUM list.

This Zero-Order Theory of Mass Spectrometry is crude but easily implemented. A more elaborate spectral theory is contained in the PREDICTOR (section 4), but obtaining such a spectrum for an arbitrary structure consumes more computer time than would be practical in a program such as the STRUCTURE GENERATOR. The Zero-Order Theory is sufficient to limit the output of the STRUCTURE GENERATOR to a small class of hypotheses, but it will need major revisions before it can be classed as a 'smart' limiting heuristic.

To make use of spectral information in the STRUCTURE GENERATOR it is merely necessary to execute (SETQ SPECTRUM L) where L is a list of integers, corresponding to the desired mass numbers. To terminate use of spectral information, execute (SETQ SPECTRUM NIL).

When structure generation is proceeding in the presence of a SPECTRUM, the work that is remembered for future reference (see section 3.7 describing the dictionary) is independent of the spectral data, so it is permissible to use several different spectra in succession in the same program core image.

3.3. Preventing the generation of forbidden substructures

Some chemical structures are so implausible (unstable) that they would never exist, either alone or imbedded within any larger structure. The STRUCTURE GENERATOR has a list (BADLIST) of these implausible structures; and no output of the STRUCTURE GENERATOR will contain any substructure on BADLIST.[1]

The STRUCTURE GENERATOR avoids generating structures containing forbidden substructures by checking rigorously before attaching new atoms to a piece of structure. At every step in generating a radical, the program knows the partially built structure and can determine whether the atom and bond which are about to be attached to it will include one of the forbidden substructures. The following process insures that no forbidden structures will be formed:

1. The partial structure is guaranteed to be plausible because of previous checking.
2. Form the new partial structure by adding the next bond and atom.
3. Consider all elements of BADLIST which have a top atom identical to the atom just added to the partial structure.
4. For each such element of BADLIST, compare the radicals which are attached to the top atom of the new structure with the radicals attached to the top atom of the BADLIST structure.
5. If every radical on the BADLIST structure is found in the list of radicals on the new structure, then the new structure must be rejected.
6. Rejecting a structure means that it is necessary to change either the added bond or the additional atom (or both) in order to generate an allowable structure.

Note that this process prevents the STRUCTURE GENERATOR from creating many implausible molecules, since the addition of each new node causes a check to be made for forbidden substructures including that node. Usually only part of the structure has been generated because unallocated atoms are added only to stable pieces.

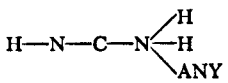
Each forbidden substructure appears on BADLIST several times, once for each possible top (apical) node. Structures are added to or removed from BADLIST by the function FIXBADLIST, which first generates all the forms of the forbidden substructure, and then adds to or deletes from BADLIST according to the desire of the user. Naturally if there are no structures on BADLIST then there are no constraints on the output of implausible structures.

[1] As described in section 1, BADLIST may be expanded according to given spectral data. The permanent part of BADLIST belongs to the program's theory of chemical instability. But substructures of both the theoretical and the context dependent parts of BADLIST are treated alike.

The current form of BADLIST has been suggested after several iterations of the following loop:

suggest forbidden substructures.
generate output.
inspect output.

The currently forbidden substructures are listed in Table 5. Hydrogen atoms must be specified explicitly, and lists of atoms enclosed in parentheses indicate that any member of the list may be used in that position on the substructure.

1. C=C—(N,O)—H
2. C≡C—(N,O)—H
3. N=N—(N,O,H)
4. H—O—C—(N,O)—H
5. H—C—N=O
6. N≡C—O—H
7. O—O
8. (N,O)—(N,O)—(N,O)
9. O—S
10. S—S—S
11. 

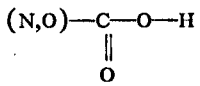
12. 

Table 5. Forbidden substructures comprising BADLIST

3.4. Partitioning a composition into plausible sub-compositions

The unconstrained structure-generating algorithm produces molecules by first determining the center of the structure (bond or particular atom) and then generating all possible radicals out of the remaining composition and attaching them to the center in all possible combinations. When all possible centers have been considered, the process of structure generation is complete.

The task of generating a set of n radicals from a single composition requires that the composition be divided (partitioned) into n subcompositions. Then a radical is generated from each smaller composition.

All possible partitions are considered in the unconstrained program, regardless of whether the sub-compositions are 'plausible' or compatible with a spectrum. The lowest partition is considered first, where 'lowest' means that it has two sub-compositions, of equal size if possible, and with the lowest ranked atoms all in one of the compositions (where the arbitrary ranking

from carbon to sulfur is: C<N<O<P<S). After the lowest partition has been used, it is incremented to the next higher form, used to make radicals, and incremented again until the highest set of compositions has been used.

As each partition is generated it can be checked for plausibility before any attempt is made to generate the corresponding radicals. Each sub-composition is checked against the spectrum. (See section 3.2.) If its weight is not present, the whole partition can be bypassed. Similarly, each sub-composition of a partition can be checked against the dictionary of previous work (see section 3.7) to determine whether any radicals can be made from the sub-composition. If the dictionary indicates an impossible composition, then the whole partition can be bypassed.

The advantages of these constraints are evident in even a simple case. Suppose we wish to partition the composition ((U . 1)(C . 6)(O . 6)) into two parts. The 'lowest' partition is [((U . 0)(C . 6)), ((U . 1)(O . 6))]. There are 17 radicals corresponding to ((U . 0)(C . 6)) but there are no allowable radicals corresponding to ((U . 1)(O . 6)). Thus, the work done in generating all the six-carbon radicals is wasted because there are no six-oxygen radicals to go along with them. If this is determined in advance, then much time will be saved by eliminating this partition. Similarly, this partition might have been eliminated if there were no spectral evidence of a C_6H_{13} fragment (a saturated radical with six carbons). Currently, the only spectral evidence that the program accepts for a composition is a peak at the corresponding mass. This drawback will soon be eliminated, for we are programming the PRELIMINARY INFERENCE MAKER to look for evidence of a more subtle kind. For example, a cluster of peaks may indicate a significant fragmentation although no peak alone indicates it.

Every composition of a partition may satisfy the spectral and dictionary constraints, yet the partition may be implausible when considered as a whole. Plausibility criteria, suggested by chemists, include such considerations as the ratio of carbon to non-carbon atoms in each composition compared to the ratio of carbon and non-carbon atoms in the whole partition. Spectral considerations may be included in the plausibility criteria at a later date. A partition plausibility score is calculated, and if this score does not lie within a given range, the partition is bypassed. The usual lower limit of plausibility scores is LLIM=4 (0 is the lowest) and the usual upper limit is ULIM=10 (the highest) but these two global variables can be reset by the program user, i.e. (SETQ LLIM 0).

The LISP function MAKELSTCLS (make-a-list-of-composition-lists) is the usual procedure for making a *n*-part partition out of a composition list. The LISP function MAKEGOODLSTCLS is called instead to insure that the partition satisfies the dictionary, spectrum, and plausibility constraints.

The partitions are usually generated one at a time as needed by the program. But the program can be made to generate all the plausible partitions in

advance and put them on a list (PARTLIST) for future reference by the program. The program assumes that the partitions obtained from PARTLIST are plausible and it makes no further checks. The main advantage in using PARTLIST is that the list can be filtered or re-ordered, either by an arbitrary scheme of the user or on the basis of plausibility scores. Then the 'most interesting' structures will be generated first. At the present time the PARTLIST is only constructed for the top level of molecule-building, although it would be possible to generate partition lists for deeper levels of structure generation.

The partition constraints are activated by the program user as follows:

1. To bypass partitions on the basis of their plausibility scores, either set LLIM>0 or set ULIM<10. If ULIM=10 and LLIM=0, all partitions will be plausible.
2. To activate the partition list, first set the switch NOPARTS=NIL. Then, before generating any molecules, the program will ask the user if he wishes the partition list to be constructed.

3.5. Specifying required substructures

The basic components used by the STRUCTURE GENERATOR are chemical atoms which possess two properties, valence and atomic weight. These atoms are connected by bonds to form radicals (structures with a free bond) which may in turn be connected with other atoms and radicals to form larger radicals and molecules.

The STRUCTURE GENERATOR can also treat complex structural fragments as atoms. Structures so treated have come to be known as 'superatoms'. The STRUCTURE GENERATOR replaces a group of atoms in the given composition by the name of a corresponding superatom, and generates structures with the revised composition (including the superatom name). Only at output time (if then) do the constituent atoms of the superatoms re-appear. Two obvious benefits arise in the use of superatoms:

1. The generation of isomers of a composition is faster because there are fewer atoms in the composition.
2. Structural fragments essential to an explanation of a mass spectrum may be made into superatoms. All isomers will contain the selected substructure, thus the output list is more relevant to the data.

A third benefit was realized as a result of the use of superatoms:

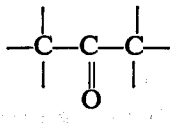
3. Ring structures can now be generated by the previously acyclic STRUCTURE GENERATOR by specifying each different ring as a superatom.

Normal atoms are known to the STRUCTURE GENERATOR because their names are on a global list called ORDERLIST. The usual value of ORDERLIST is '(C N O P S)' and the position on this list defines the 'DENDRAL Order' of the atom: C<N<O<P<S. Each atom on ORDERLIST has the properties

VALENCE and WEIGHT. When superatoms are created, their names are also added to ORDERLIST. Superatoms have the properties VALENCE, WEIGHT, STRUCT and SYM (the last two to be described later).

Any new atom or superatom may be introduced to the STRUCTURE GENERATOR by calling the LISP function (ADDATOM (QUOTE X)) where X is the name[1] of the atom. A common superatom is the keto radical *CO*, >C=O, with two free bonds on the carbon atom. This superatom has properties VALENCE=2 and WEIGHT=28 and STRUCT=(1 C(2 O)). It is list notation representation[2] that is stored under the property STRUCT of the superatom name. It is used at output time to put the generated isomer back into canonical DENDRAL notation in terms of ordinary atoms. Note that the free bond leading into the superatom is a 1 rather than a 2, since the most saturated form is always used.

But consider the case of the general ketone substructure, *KET*:



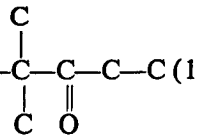
It is desirable to be able to treat this as a superatom, yet there must be some way of stating that the valence of 6 is split up between two different atoms. This is done by stating that VALENCE=(3 3). The property WEIGHT has value 52 and STRUCT=(1 C(1 C(1 C)(2 O))). The numbers in the valence list correspond to atoms with free valences in the list structure, reading from left to right. The first atom which does not have all its valences filled by bonds has an effective valence equal to the first number in the valence list, and so on.

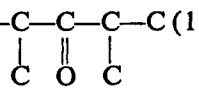
During normal structure generation, the effective valence of this superatom is six, and as many as six radicals may be generated to attach to a ketone superatom. The actual locations to which the radicals are attached are indicated by a locant vector associated with the superatom name in the list notation for a larger structure. The locant vector is a list of numbers, one for each attached radical. Each number specifies the atom to which the radical is attached. The ketone superatom has atoms 1 and 2 which are available for attached radicals. (An atom's number corresponds to its position from left to right in the valence list and the list notation of the structure.) Since each available atom in the ketone structure has three free valences, each atom (1 and 2) may be listed up to three times in the locant vector.

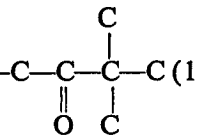
As an example, suppose four methyl groups (radicals of one carbon atom, three hydrogen atoms each) are to be attached to the ketone superatom, represented by '*KET*'. The possibilities, with associated locant vectors and list notation representation, are:

[1] Any atom with a name longer than 1 character should be enclosed in asterisks.

[2] List notation is analogous to dot notation described in section 7. The difference is that bonds are represented by numbers and radicals are enclosed in parentheses.

1. 
(1 1 1 2) (NIL (*KET* 1 1 1 2)(1 C)(1 C)(1 C)(1 C))

2. 
(1 1 2 2) (NIL (*KET* 1 1 2 2)(1 C)(1 C)(1 C)(1 C))

3. 
(1 2 2 2) (NIL (*KET* 1 2 2 2)(1 C)(1 C)(1 C)(1 C))

Note that 1's in the locant vector are listed before 2's, when the two representations would be equal. For example, in this case

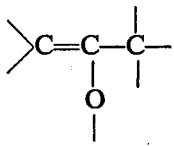
$$(1\ 1\ 1\ 2)=(1\ 1\ 2\ 1)=(1\ 2\ 1\ 1)=(2\ 1\ 1\ 1).$$

A firm rule with locant vectors is that the 'lowest' of equivalent forms is always used.

Further, note that structures 1 and 3 are mirror images, obtained by interchanging nodes 1 and 2. This arises because the ketone superatom is symmetric for these nodes and because the radicals attached to the symmetric nodes are equal. Thus it becomes necessary to check for symmetrical structures to avoid redundancy.

An important property for each superatom with split valences (i.e., the valence property is a list rather than a single number) is the list of symmetries. Each symmetry is a list of numbers, the same length as the valence list, indicating which node is equivalent to the *n*th node under a given transformation. The identity transformation gives the symmetry (1 2) (node 1 equivalent to node 1, node 2 equivalent to node 2). The identity symmetry is possessed by every split valence superatom and thus is not included in the symmetry list. The reflection transformation applied to the ketone superatom gives the symmetry (2 1) (node 2 equivalent to node 1, node 1 equivalent to node 2). This is the only non-trivial symmetry, so the SYM property for ketone is ((2 1)).

In the case of a superatom such as

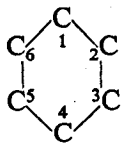


the properties would be

STRUCT=(1 C(2 C(1 C)(1 O)))
VALENCE=(2 3 1)
WEIGHT=52
SYM=()

In the case of the six membered carbon ring shown below the properties would be:

VALENCE=(2 2 2 2 2 2)
WEIGHT=72
SYM=((2 3 4 5 6 1)(3 4 5 6 1 2)
(4 5 6 1 2 3)(5 6 1 2 3 4)
(6 1 2 3 4 5)(6 5 4 3 2 1)
(5 4 3 2 1 6)(4 3 2 1 6 5)
(3 2 1 6 5 4)(2 1 6 5 4 3)
(1 6 5 4 3 2)).



All possible rotations and reflections produce equivalent structures, so all symmetries are listed.

A ringed structure cannot be written in list notation conveniently, so the composition list is given as the value of STRUCT, for rings, STRUCT= ((U.1)(C.6)) in this example. (At least one degree of unsaturation is always present in a ring.) Rings are not converted back to basic atoms at output time since there is no convenient way to write them on a single line. The locant vector notation is retained during output of structures containing rings. The first number in the locant vector refers to the attachment position of the afferent link if there is one.

The presence of a superatom name on ORDERLIST defines the superatom for the system but does not force the STRUCTURE GENERATOR to use it. Another global list called GOODLIST, is set by the program user to indicate which superatoms are relevant at a certain time. An element of GOODLIST has the form (NAME MAX MIN), where MAX and MIN specify the upper and lower limits on the number of these superatoms which may be substituted into a given chemical composition. For example, if MAX=0, this superatom is ignored, and if MIN=1, then this superatom is required. If the MAX and MIN are left unspecified, then the assumed values are MAX=100 and MIN=0.

When the STRUCTURE GENERATOR is given a composition from which to generate structures it must first check GOODLIST to see which superatoms can be formed from the given composition. If any superatom has a specified minimum greater than 0, its atoms are removed from the composition and the corresponding number of superatoms are inserted instead. If any superatom with a minimum greater than 0 cannot be made from the atoms of the composition, then all structure generation terminates for that composition. If the Zero-Order Theory is operating, structure generation also terminates if the mass of the superatom is not in the spectrum.

When the required superatoms have been inserted into the composition, the STRUCTURE GENERATOR then places all possible combinations of remaining superatoms into the composition and generates all possible structures from the new set of corresponding compositions.

Consider an example:

```
GOODLIST=((*COOH*)(*CO*)(*NOH*)(*KET*))
*COOH*: STRUCT=(1 C(1 O)(2 O))
        VALENCE=1
        WEIGHT=45
*CO*:   STRUCT=(1 C(2 O))
        VALENCE=2
        WEIGHT=28
*NOH*:  STRUCT=(1 N(1 O))
        VALENCE=2
        WEIGHT=31
*KET*:  STRUCT=(1 C(1 C(1 C)(2 0)))
        VALENCE=(3 3)
        WEIGHT=52
        SYM=((2 1))
```

The composition $C_3H_7NO_2$ has the composition list ((U.1)(C.3)(N.1)(O.2)). The composition lists for the four superatoms are:

```
((U.1)(C.1)(O.2))
((U.1)(C.1)(O.1))
((U.0)(N.1)(O.1))
((U.1)(C.3)(O.1))
```

The following revised composition lists are all possible ways of generating isomers of $C_3H_7NO_2$.

1. ((U.0)(C.2)(N.1)(*COOH*.1))
2. ((U.0)(C.2)(*CO*.1)(*NOH*.1))
3. ((U.0)(C.2)(N.1)(O.1)(*CO*.1))
4. ((U.1)(C.3)(O.1)(*NOH*.1))
5. ((U.0)(*NOH*.1)(*KET*.1))
6. ((U.0)(N.1)(O.1)(*KET*.1))
7. ((U.1)(C.3)(N.1)(O.2))

There are two global variables which may be set to limit the total number of superatoms in any single composition. These are MXSAT and MNSAT, representing the maximum and minimum number, respectively. In the example above, if MXSAT=1 then #2 and #5 would not be permitted; or if MNSAT=1 then #7 would not be permitted. If GOODLIST had the form

```
((*COOH* 2 1)(*CO*)(*NOH*)(*KET*))
```

then only #1 would be permitted.

When GOODLIST is present during structure generation, the program does not allow implicit superatoms to be formed from the remaining normal atoms. The structure (1 C(1 O)(2 O)), for example, is a forbidden substructure when the explicit superatom *COOH* is on GOODLIST. Thus, if isomers

were generated for all the seven composition lists given above, the total number of isomers would be exactly the number generated by the program for the composition $C_3H_7NO_2$ with no GOODLIST present. For that reason, the variables specifying the maximum and minimum for total and individual superatoms are nearly always specified to prevent uninteresting isomers from being generated. If a GOODLIST superatom happens to contain one of the forbidden substructures on BADLIST, BADLIST prevails. That is, the BADLIST theory of chemical instability must be changed before conflicting superatoms will be generated.

The superatom type of structure generation is activated whenever GOODLIST is not NIL. A general form of GOODLIST is stored in the program and can be used by executing (SETQ GOODLIST SAVEGOODLIST). A function called STRUCTURES will ask the user to specify the maximum and minimum number for each superatom on GOODLIST. As described earlier, the PRELIMINARY INFERENCE MAKER gives superatoms to the STRUCTURE GENERATOR automatically so that generated isomers will correspond as closely as possible to the information contained in the real spectrum.

3.6. Observing the process of structure generation

The basic structure-generating function is a LISP function called GENRAD which generates a single radical from a composition list. GENRAD operates recursively by calling itself for successively smaller subsets of the composition list.

It sometimes happens that the work done in generating radicals from a composition subset is wasted because the partial structure to which these radicals will be attached is implausible. Or else, a user of the program may be uninterested in certain classes of structures, but has to watch impatiently while the program works its way down to the interesting structures.

A dialog option allows the on-line user of the program to inspect GENRAD at each level and to decide whether GENRAD should be allowed to proceed to a deeper level of recursion. At each dialog point, the current partial structure and remaining composition are printed. This gives the user the opportunity to decide if the structure-generating algorithm is proceeding along a fruitful path. If not, a sort of user-implemented tree pruning can be evoked just by answering 'N' to the program's query about whether to continue. It is hoped that the program can learn to make use of the reasons for altering its operation in this way.

The dialog option is initiated by executing (SETQ DIALOG (QUOTE ON)) and it is terminated by executing (SETQ DIALOG (QUOTE OFF)). The dialog may be stopped during structure generation by typing a left arrow instead of the usual answer to questions typed by the program. Then type (SETQ DIALOG (QUOTE OFF)) and the dialog will no longer appear.

The user has another device available to cause the program to terminate structure generation even though the output list is incomplete. To control

output in this way, execute (SETQ OUTCONTROL (QUOTE ON)). The program will pause after a specified number of outputs have been printed and ask permission to continue. The interval for pausing is determined by the number stored as the value of the global variable called AMT.

Both these devices for controlling output are usually ignored, but have been very useful during program debugging and demonstrations.

3.7. Rote memory

The normal operation of the STRUCTURE GENERATOR causes a dictionary to be built which contains all the structural isomers of every composition (and every subset of every composition) which has been encountered. This dictionary contains lists of radicals, saved under names which can be reconstructed from the compositions. Whenever structure generation is under way, the program first searches the dictionary to see if the current composition has been encountered previously. If a dictionary entry exists for a composition, it is assumed to be an exhaustive list of all radicals which can be made from the composition. No further structure generation is performed; the dictionary list provides the output.

Dictionaries which are built during a run of the program may be saved on tape for further use. The function SAVDICT writes a dictionary on tape. The dictionary is recalled for further use by the function GETDICT. The global variable called DICTLIST has as its value a list of all names of entries in the current dictionary.

Because every dictionary entry is assumed by the program to be an exhaustive list of radicals corresponding to the named composition, the dictionary may be used as a program constraint. An existing dictionary (either in core or on tape) may be edited manually (or by computer, using techniques which will be described in a later report). The editing may either delete or add radicals, and subsequent structure generation with the edited dictionary present will result in reduced or expanded output lists.

Sometimes a previously edited dictionary is used unintentionally. For example, BADLIST prevents certain structures from entering the dictionary at all. If BADLIST is later changed, but a previously built dictionary is left in the program, the output will appear as though the old form of BADLIST were still present; or worse, it may appear that the old BADLIST is present for small structures (ones that were previously in the dictionary) but the new BADLIST is present for large structures (ones that are being built for the first time). A certain amount of caution in changing BADLIST will prevent this type of erroneous structure generation.

The dictionary-building process may be turned off so that previous work is not remembered. The command for this is (SETQ DICTSWITCH (QUOTE OFF)), but seldom is it advisable to do this. It speeds initial work, but later work that depends on previous work is slower. A core dictionary may be deleted by executing (UNDICT DICTLIST) and (SETQ DICTLIST NIL). To

restart the dictionary building process after it has been turned off, execute (SETQ DICTSWITCH (QUOTE ON)).

4. THE PREDICTOR: FIRST STEPS TOWARD A COMPUTER THEORY OF MASS SPECTROMETRY

Introduction

Part of HEURISTIC DENDRAL is a computer program which predicts major features of mass spectra of acyclic organic molecules. The program contains a rough theory of mass spectrometry which is still in its formative stages.

In the course of designing the HEURISTIC DENDRAL program for formulating hypotheses to explain mass spectral data, it became apparent that the program needed a detailed theory of mass spectral fragmentation processes. This is because the STRUCTURE GENERATOR suggests plausible candidate structures for explaining the data, but has no way of testing its candidates. Thus, a theory by which the computer could make some verifiable predictions about each candidate would help to reduce the set of likely candidates. Through the program described here, the prediction now takes the form of a suggested mass spectrum for each candidate structure. The EVALUATION FUNCTION described in section 5 then compares the predicted spectrum and original spectrum to determine which structural candidate is the most satisfactory choice.

A mass spectrometer is, briefly, an instrument into which is put a small sample of some chemical compound and out of which comes data representable as a bar graph. The instrument itself bombards molecules of the compound with electrons, producing ions of different masses in varying amounts. The x-points of the bar graph represent the masses of ions produced,[1] and the y-points represent the relative abundances of ions of these masses. The spectrum predictor program is an attempt to codify the numerous descriptions of what happens inside the instrument, and thus to generate mass spectra in the absence of both the instrument and the actual sample of the substance. The input to the program is a string of characters representing the graph structure of a molecule. The output is a bar graph representing the predicted mass spectrum for this molecule.

In broadest outline, the mass spectrum predictor calculates a spectrum (list of mass-intensity pairs) for a molecule in the following series of steps:

1. Calculate the mass of the molecular ion and an associated intensity, depending on the degree of unsaturation.
2. Determine the nature and extent of eliminations and rearrangements of the molecular ion.
3. Break a bond between a pair of adjacent atoms in the molecule, looking at each bond only once.

[1] More accurately, the x-points of a mass spectrum represent the mass to charge ratio (m/e), where most, but not all, recorded fragments are singly charged.

4. Calculate the masses of the two resulting fragments. Then calculate an intensity (on an absolute scale) for each fragment ion by estimating the probability that this bond will break and the probability associated with ionization of each fragment.
5. Determine the nature and extent of eliminations and rearrangements of each fragment ion.
6. Add isotope peaks and peaks at $m+1$, $m+2$, to account for hydrogen addition to some fragment ions. (Optional.)
7. Recycle through 3–6 until every bond in the molecule has been considered once.
8. Eliminate low mass peaks and adjust the intensities to percent of the highest peak.

The following discussion elucidates the theory by explaining in detail what the program does in each of the above steps. Justification for some decisions exists in print. Many decisions, however, have been made out of considerations of simplicity, elegance, deference to the intuitions of professionals, or out of ignorance.

4.1. The Program

The molecule is represented in a slight variant of Lederberg's DENDRAL notation described in section 4.4. This notation omits explicit mention of hydrogen atoms but shows all the other connections of a chemical graph. The program attaches a unique name to each atom, and keeps track of each atom's place in the graph by putting names of adjacent atoms on the property list of each atom under indicators FROM and TO.

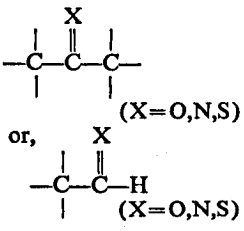
The molecular ion's mass is the sum of the masses of all atoms plus the masses of all implicit hydrogen atoms. The intensity associated with this mass is a function of the degree of unsaturation of the molecule. This function is easily changed, but it is currently x times the product of all bond orders (on an absolute scale) where $x=2$ [LISP function called MOLVAL].[1] For example, in a molecule with all single bonds except for one double bond and one triple bond, the intensity of the molecular ion would be

$$2(1\times 1\times,\ldots,\times 1\times 2\times 3)=12.$$

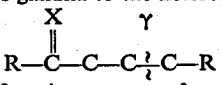
Peak heights on this absolute scale are translated onto a 0–100 scale at the end.

Since any ion, including the molecular ion, tends to a more stable form if possible, the program must take account of the relative stability of each ion, as compared to the stability of possible products it may form. The program looks for ways of losing certain neutral molecules or rearranging the atoms already present. It has a list of very stable ion products, which are preferred structures, and must determine whether, and to what extent, the given molecular ion can form one of these products. To do this, it looks for any occurrence

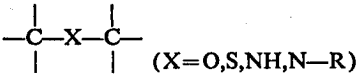
[1] The most important LISP functions and parameters will be bracketed throughout this section.

1. 

a. Only two types of bonds break:
(i) Bonds alpha to the heteroatom; i.e., the bonds between C and

C, with only the heteroatom-containing fragment ionized. [test function ALPHAP]
(ii) Bonds gamma to the heteroatom; i.e., 

[test function GAMMAP]
b. The McLafferty rearrangement, if possible, is invoked for the molecular ion.
c. Carbon monoxide is lost from each α-cleavage fragment containing the carbonyl radical.

2. 

a. Only two types of bonds break:
(i) α-bonds, as above. Preference is given to loss of the most highly substituted fragment or to loss of the longest carbon chain (when degrees of substitution are equal).
(ii) γ-bonds, as above.
b. In each resulting fragment, subsequent rearrangements favor loss of highly substituted carbons and loss of long carbon chains.

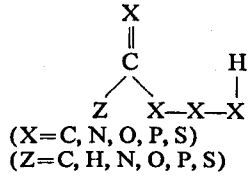
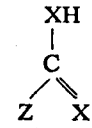
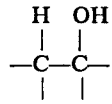
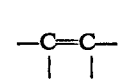
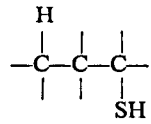
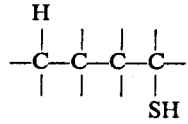
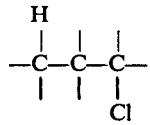
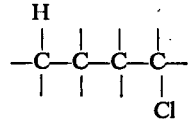
3. —C—OH

a. Reduce the intensity of the molecular ion to zero.
b. Add peaks at the following x, y points (intensities on absolute scale, $a=2$)

mass	*intensity*
M–18	$2a$ (=4)
M–18–15	$4a$ (=8)
M–18–28	$8a$ (=16)
M–18–29	$3/4 \cdot 8a$ (=12)
M–18–42	$8a/2$ (=8)
M–18–43	$3/4 \cdot 8a/2$ (=6)
M–18–56	$8a/4$ (=4)
M–18–57	$3/4 \cdot 8a/4$ (=3)

until $(M\text{–}18\text{–}X)<29$.
[The value of a is controlled by the parameter AFACT.]
c. Add α-cleavage peaks (with only the hydroxyl fragment ionized). Preference is given to loss of the most highly substituted fragment or to loss of the longest carbon chain.

Table 6. Significant radicals and their effects

type	characteristic subgraph	allocation of intensity units of the molecular ion and [parameter names]* $K1\%$ for parent	$K2\%$ for daughter	skeleton of daughter ion
McLafferty Rearrangement	 (X=C, N, O, P, S) (Z=C, H, N, O, P, S)	40 [KMCOLD]	60 [KMCNEW]	
1, 2† elimination of H_2O (thermal)		60 [KWOLD]	40 [KWNEW]	
1, 3† elimination of H_2S		0 [KS3OLD]	200 [KS3NEW]	‡
1, 4† elimination of H_2S		0 [KS4OLD]	300 [KS4NEW]	‡
1, 3† elimination of HCl		0 [KCL3OLD]	200 [KCL3NEW]	‡
1, 4† elimination of HCl		0 [KCL4OLD]	300 [KCL4NEW]	‡

* Global parameters' current values are listed; parameters may be reset by the user.

† The number notation *l,n* refers to the relative positions of the combining atoms or radicals.

‡ The program now calculates only a mass-intensity pair of these daughter ions since we are uncertain about their structures.

Table 7. Rearrangements and eliminations in molecular ions

of the significant radicals listed in Table 6, for example the carbonyl radical >C=O. Associated with each of these significant radicals is a set of rules for restructuring the ion to make it more stable and a set of parameters for determining the extent to which this restructuring should (or does) occur. Heuristic programmers recognize such a plan as a list of situation-action rules of the form: in situation *X* perform action *Y*. A very desirable feature of this is that the list can be extended or amended very easily.

Table 7 is a summary of the program's rules for eliminating neutral molecules or rearranging atoms in molecular ions. The characteristic subgraph is the part of the ion which the program looks for. The two parameters determine the percent of the abundance of the molecular ion (as originally determined) which should be allocated to the original molecular ion ($K1\%$) and to the daughter ion, that is, to the ion after restructuring ($K2\%$). Resetting the parameter whose name is bracketed changes the corresponding allocation. The skeleton of the daughter ion is indicated. The program has rules for removing atoms, changing the order of bonds, and moving atoms from place to place, so that the structure of rearrangement products will be printed, if requested.

After the program has considered the molecular ion [function PARENT] it considers the likelihood that the molecular ion will fragment at each of the bonds between atoms. Its theory says that only single bonds will break apart, thus it skips over double and triple bonds in the molecule. Of the single bonds, it distinguishes bonds between carbon atoms from bonds between a carbon and a non-carbon atom. The probability that the ion fragments at a given bond depends upon the environment of the bond and the functional groups present in the molecule. The probability associated with the ionization of one or the other of the resulting fragments also depends on these features. This part of the program is also organized as a set of conditional sentences: if the molecule contains functional group X and this bond environment has feature b then include the factor resulting from calculation $f(b)$ in figuring the probability that the molecule fragments at this bond. The features which the program considers are explained in detail in section 4.2, together with the associated functions. Briefly, however, the program first checks for the presence of the significant radicals listed in Table 6 and then looks at some or all of the following eleven features to determine both the probability that any particular bond will break and the probability of ionization for each resulting fragment:

1. The order of the bond itself [HTVAL].[1]
2. The types of atoms joined by the bond [HTVALCC, HTVALCX, INDUCTIVE].
3. The orders of the bonds which are one atom removed from this bond [VINYLIC].
4. The number of non-hydrogen atoms which are one atom removed from this bond, i.e., the degree of substitution on the first atom in each fragment [LCALC, METHYLP].
5. The number of heteroatoms which are one atom removed from this bond [CONTIGHET].
6. The types of heteroatoms which are one atom removed from this bond [HETERO].

[1] The bracketed names of the LISP functions responsible for these features are given for later reference.

7. The types of radicals that are one atom removed from this bond [CARBONYL].
8. The orders of the bonds which are two atoms removed from this bond [ALLYLIC].
9. The length of the carbon chain in each fragment [CHAINLTH].
10. The total number of heteroatoms in each fragment [NHET].
11. The total number of carbon atoms in each fragment [NUMCARBS].

The result of calculating the probability of ionization of a fragment is a number pair, $(x.y)$. The first component is the mass of the fragment. The second is the relative abundance of these fragment ions ($y \geq 0$).

Since fragment ions, as well as the molecular ion, may eliminate neutral molecules or rearrange atoms to form more stable ions, the program must be able to predict the most significant occurrences. After the program calculates the mass of a fragment and the relative frequency of its ionization, it checks the fragment for elimination and rearrangement possibilities. Exactly the same procedure is used as for the molecular ion, but the list of characteristic subgraphs may be different depending on the functional groups present in the molecule. Table 8 lists the different possibilities now in the program.

Thus the program examines each bond to calculate the probability of cleavage and each fragment to calculate both the probability of ionization and the possibility of rearrangements. In addition, it has already calculated a molecular ion peak and has looked for the possibility of eliminations and rearrangements in the molecular ion. By the time it has finished, it should have calculated a list of mass-intensity pairs corresponding to the most significant peaks in the actual mass spectrum for the same molecule. To conform to common practice, mass units below 29 are deleted and the intensities are converted to percent of the highest peak (base peak).

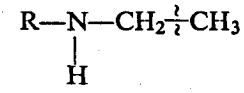
Some annotated examples of predicted spectra appear in section 4.3 together with the actual mass spectra for the same molecules. Section 4.4 explains how to use the program; and section 4.5 explains the options that are available from the console.

4.2. Rules for calculating relative intensities of primary fragments

Cleavage of Single Bonds between Carbon Atoms. Under special conditions the program bypasses the general rule for calculating intensities of fragments given below. Thus, before stating the general rule, which is relatively complex, the exceptions will be noted.

A. *Exceptions*

1. Assign zero as the intensity of the two fragments when considering the bond between CH_2 and CH_3; that is, do not break off a methyl which is part of an ethyl. This rule is preempted by the special rules of Table 6 for significant radicals. For example, the methyl radical *will* be lost in this molecule



because this is α-cleavage in an amine. [test function METHYLP].

2. Assign zero as the intensity of both fragments when considering a vinylic bond; that is, do not break bonds which are adjacent to double bonds. The rules of Table 6 preempt this rule also. [function VINYLIC].

3. Assign zero as the intensity of all primary fragments of ketones, aldehydes, amines, ethers, thioethers, and alcohols except for the fragments resulting from the rules of Table 6.

4. For α-cleavage in amines, ethers, and thioethers, calculate the intensity of the heteroatom-containing fragment as a function of

(a) the degree of substitution of the first carbon atom of that fragment, and

(b) the number of carbon atoms lost.

Specifically, the intensity is the sum of two factors X_1 and X_2 where

$X_1 = 0$ if 3 or 2 hydrogens are attached to the first carbon,
30 if 1 hydrogen is attached, or
45 if 0 hydrogens are attached. [function AMINESUBST]

$X_2 = 3$ if 2 or 1 carbon atoms are lost,
10 if 3 are lost, or
15 if 4 or more are lost. [function AMINECARBS].

B. *The General Rule*

The general rule for calculating the intensity of each fragment resulting from dissolution of the bond between two carbon atoms is

$$I = (Z1 + Z2 + Z3 + W1) \times W2 \times W3 \times W4$$

The Z-factors are context-dependent factors. That is, it is necessary to look at features of both fragments (the total context) in order to calculate each Z-factor. The W-factors are context independent, which is to say that each one can be calculated by looking only at the fragment under immediate consideration.

1. $Z1$ is calculated in two steps according to the number of non-hydrogen atoms alpha to the bond under consideration:

(a) Compute a factor ($T1$) which is equal to the sum of intensities of both fragments (estimated probability that this bond will break given this information):

$T1 = 10$ if there are 0, 1, or 2 branches to non-hydrogen atoms,
12 if . . . 3 branches . . .,
16 if . . . 4 branches . . .,
18 if . . . 5 branches . . .,
20 if . . . 6 branches . . . . [parameter list TINLTH]

(b) Compute a ratio for splitting $T1$ between the two fragments (relative probability that each fragment will be ionized as a result of this break given this information):

ratio = 1:1 if the difference between the number of non-hydrogen branches is 0,
5:7 if the difference . . . is 1,
1:3 if the difference . . . is 2,
1:5 if the difference . . . is 3. [parameter list RINLTH]

The ratio is weighed in favor of the fragment with more alpha branches. $Z1$ for each fragment is then the result of applying this ratio to $T1$.

[function LCALC]

2. $Z2$ is calculated in a similar two-step manner, this time taking into account the number of heteroatoms (non-hydrogen, non-carbon atoms) alpha to the bond under consideration:

(a) Compute a factor ($T2$) which is equal to the sum of intensities of both fragments (estimated probability that this bond will break given this information):

$T2 = 0$ if there are 0 branches to non-hydrogen, non-carbon atoms (heteroatoms) from both of the carbon atoms,
3 if . . . 1 branch . . .,
10 if . . . 2 branches . . .,
20 if . . . 3 branches . . .,
30 if . . . 4 branches . . .,
40 if . . . 5 branches . . .,
50 if . . . 6 branches . . .. [parameter list TINCON]

(b) Compute a ratio for splitting $T2$ between the two fragments (relative probability that each fragment will be ionized as a result of this break given this information):

ratio = 1:1 if the difference between the number of branches to heteroatoms is 0,
3:10 if the difference . . . is 1,
1:9 if the difference . . . is 2,
1:19 if the difference . . . is 3. [parameter list RINCON]

(again, weighted in favor of the fragment with the more branches). $Z2$ for each fragment is then the result of applying this ratio to $T2$.

[function CONTIGHET]

3. $Z3$ is an attempt to integrate the principle that longer carbon chains are lost preferentially to smaller ones. The longer a carbon chain in a fragment, the higher the probability that the molecule will split apart at that bond. Also, though, the long-chain fragment is less likely to be ionized than the other fragment at this break-point. So $Z3$ is calculated for fragment #1 at a break-point as a function of the chain length of fragment #2.

[function CHAINLTH]

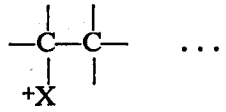
Currently the function just multiplies the chain length by two [the value of CHFACT] although this parameter, like every other in the program, can be easily changed.

The next factors in the intensity calculation for any fragment are context-independent. The program considers only features within the fragment, first on one side of the bond under consideration, then on the other.

4. $W1$ is equal to the number of heteroatoms in the fragment. The program now simply counts the number of occurrences of non-hydrogen, non-carbon atoms, although it could return some function of the count.

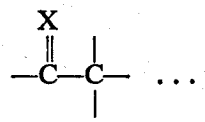
[function NHET]

5. $W2$ attempts to capture the principle that in a fragment the types of heteroatoms alpha to the bond under consideration greatly influence the probability that the fragment retains the charge (is ionized) when this bond is broken. That is, the program looks at atoms in the place occupied by X in the following schema and assigns $W2$ by the accompanying rule:



$W2=5$[FHETN] if X is Nitrogen,
4[FHETS] if X is Sulfur,
3[FHETO] if X is Oxygen,
2[FHETCL] if X is Chlorine,
1 otherwise. [function HETERO]

6. $W3$ is a similar factor taking account of certain heteroatoms doubly bonded to the carbon at the break-point. The program looks at atoms in the X-place in the schema and assigns $W3$ by the following rule:



$W3=4$[FCARBN] if X is Nitrogen
3[FCARBO] if X is Oxygen
2[FCARBS] if X is Sulfur
1 otherwise. [function CARBONYL]

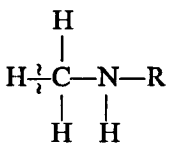
Thus for a bond connecting two carbon atoms in the molecule, the intensities of the two fragments depend upon the context-dependent and context-independent factors (the Zs and Ws) as just described. The atoms closest to the bond have the greatest effect, but two of the factors ($Z3$ and $W1$) depend upon atoms farther away from this bond.

7. $W4$ is a factor which attempts to capture the favorable influence of allylic bonds on the fragmentation process. For example, in fragment (a) below the bond marked with an asterisk is an allylic bond (relative to the double bond of the fragment) and thus increases the probability of fragmentation to produce fragment (a).

(a) —*C—C=C—R [function ALLYLIC, parameter KALLYLIC]

Cleavage of Carbon-Heteroatom Bonds. For a single bond between a carbon atom and a heteroatom, several of the same calculations are made as for carbon-carbon bonds. In accordance with existing theory, the carbon-containing fragment is much more likely to be ionized as a result of this cleavage than the other one. (In rearrangement products, however, the hetero-atom-containing fragment often retains the charge; see the amine entries of Table 8 for example.) But this bond is less likely to be broken than a single bond between two similar carbon-containing fragments.

A. *Exception.* A bond between a carbon and hydrogen atom breaks if this is an α-cleavage in an amine. For example,



B. *The General Rule.* The equation for calculating the intensity of the carbon-containing fragment at a C—X break is:

$$I_c = Z4 \times Z5 \times W5.$$

As before the Zs are context-dependent factors and the W is context-independent.

For the heteroatom-containing fragment the intensity is merely:

$$I_x = Z4.$$

1. $Z4$ is directly analogous to the factor $Z1$ for carbon-carbon bonds. The carbon-containing fragment resulting from such cleavage ordinarily should have a smaller intensity than the corresponding fragment in cleavage of a carbon-carbon bond. The program accounts for this in the first two steps for calculating $Z4$.

(a) Compute a factor ($T4$) which is to be equal to the sum of the intensities of both fragments:
$T4$ = the intensity which would be assigned to the carbon-containing fragment in a similar carbon-carbon bond environment. (The program 'pretends' that the heteroatom is a carbon atom and computes $T4 = ((Z1+Z2+Z3+W1) \times W2 \times W3)$ as above for the intensity to be divided between the two fragments.)

(b) Compute a ratio for splitting $T4$ between the two fragments according to the number of heteroatoms in the carbon-containing fragment which are alpha to the bond under consideration:
ratio = 10:1 if the number of heteroatoms attached to this carbon atom is 0
20:1 if the number of heteroatoms . . . is 1
30:1 if the number of heteroatoms . . . is 2
40:1 if the number of heteroatoms . . . is 3

(weighted in favor of the carbon-containing fragment). $Z4$ for each fragment is then the result of applying this ratio to $T4$. The smaller intensity is returned for the non-carbon fragment.

[function LCALCX]

For the carbon-containing fragment two additional context-dependent factors $Z5$ and $W5$ are calculated. The intensity for this fragment is the product

$$Z4 \times Z5 \times W5.$$

2. $W5 = 5$[FHETN] if nitrogen is singly bonded to the carbon at the break point
   4[FHETS] if sulphur . . .
   3[FHETO] if oxygen . . .
   2[FHETCL] if chlorine . . .
   1 otherwise. [function HETERO]

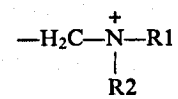
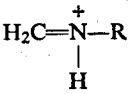
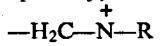
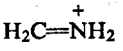
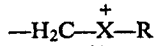
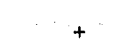
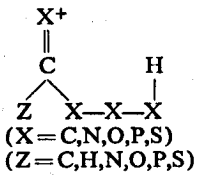
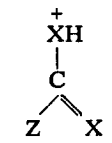
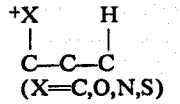
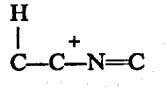
type	characteristic subgraph	allocation of intensity units of the parent ion [and parameter names] $K1$% of parent's intensity (for parent)	$K2$% of parent's intensity (for daughter)	skeleton of daughter ion
Rearrangement of *Amines*		100 [KAM3OLD]	80 [KAM3NEW]	
	(Check degree of substitution and number of carbon atoms in R1 and R2 to see which drops away) 	100 [KAM2OLD]	80 [KAM2NEW]	
Rearrangement of *Ethers and Thioethers*	 (X=O,S)	100 [KAM2OLD]	80 [KAM2NEW]	
McLafferty Rearrangement	 (X=C,N,O,P,S) (Z=C,H,N,O,P,S)	40 [KMCOLD]	60 [KMCNEW]	
Type F (Biemann)	 (X=C,O,N,S)	80 [KFOLD]	20 [KFNEW]	
Type G (Biemann)		20 [KGOLD]	80 [KGNEW]	

Table 8. Rearrangements for fragment ions

3. $Z5 = 5$[KINDCL] if the heteroatom X at this C—X breaks is chlorine
   4[KINDBR] if the heteroatom . . . is bromine, oxygen or sulphur
   3[KINDI] if the heteroatom . . . is iodine
   1 otherwise [function INDUCTIVE]

The next section shows some examples with brief explanations of the PREDICTOR's work. Whenever a chemist finds major discrepancies between predicted and actual spectra, we try to localize the contributing functions or parameters and change them. The specialized rules of Table 6 and Table 8 in particular, directly resulted from finding major errors in predictions for ketones, amines, ethers, and alcohols. Instead of adjusting the core of the theory in these cases, however, special tests and branches were added. At a later date, we hope to be able to reunify the PREDICTOR's theory.

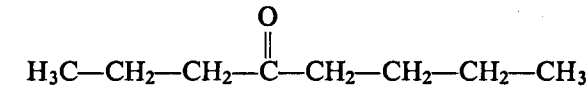
4.3. Examples

The command DRAW in each case started the predictor's work on the indicated structures. The list of number pairs following the command is the output from the program: the mass-intensity pairs of the most significant ionized fragments.

Example A

(DRAW (QUOTE C2110C1C1CC1C1C1C$))
((43 . 100) (57 . 88) (58 . 22) (71 . 100) (85 . 88) (86 . 22) (128 . 14))

The graphical representation for this molecule, 4-octanone, is



The mass spectrum for this compound from the Stanford University Mass Spectrometry Laboratory is

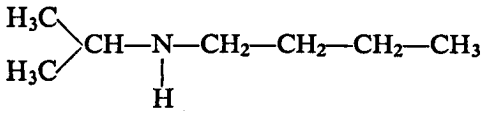
((41 . 48) (42 . 8) (43 . 100) (44 . 3) (53 . 2) (55 . 8) (56 . 2) (57 . 92) (58 . 56) (59 . 2) (64 . 1) (67 . 1) (69 . 3) (70 . 1) (71 . 91) (72 . 4) (81 . 1) (83 . 1) (84 . 1) (85 . 60) (86 . 23) (87 . 2) (99 . 3) (113 . 2) (128 . 13) (129 . 1))

The molecular ion has mass 128. The two other even numbered peaks of high intensity, 86 and 58, are the results of the McLafferty rearrangement of the molecular ion (twice). The peaks at 85 and 71 result from alpha-cleavages, in each case with only the heteroatom-containing fragment retaining the charge. The peaks at 57 and 43 come from loss of carbon monoxide (mass 28) from each of the alpha cleavage fragments. The remaining peaks in the actual spectrum are of little informative value to chemists, thus they remain unpredicted. Several of these could be regarded as isotope peaks and thus could have been predicted (by setting IPEAKS = T).

Example B

(DRAW (QUOTE C111CCN1C1C1C1C$))
((30 . 17) (44 . 80) (72 . 21) (100 . 100) (115 . 2))

This molecule is graphically represented as

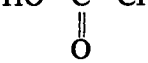


and its actual mass spectrum (from the Stanford Mass Spectrometry Laboratory) is

((41 . 38) (42 . 21) (43 . 25) (44 . 88) (45 . 2) (54 . 1) (55 . 3) (56 . 8) (57 . 13) (58 . 16) (70 . 11) (71 . 4) (72 . 100) (73 . 4) (84 . 2) (85 . 2) (98 . 3) (100 . 61) (101 . 4) (114 . 2) (115 . 5))

The molecular ion is of mass 115. Alpha-cleavage accounts for the peaks at 100 and 72, in each case with only the nitrogen-containing fragment retaining the charge. The amine rearrangement shown in Table 8 affects each of the alpha-cleavage fragment ions resulting in the peaks at masses 44 and 30. In the actual spectrum, peaks below mass 41 were not recorded, but it is not unreasonable to believe that the peak at mass 30 would be a strong peak. Some of the other discrepancies may be due to isotope peaks; many of the rest from lack of rules for amine fragmentation processes.

4.4. Using the mass spectrum PREDICTOR

To run the program, once it is in core, call the top level function DRAW. This function requires one argument, a name of the quasi-DENDRAL symbol string which represents the molecule whose spectrum is to be predicted. For example, either (a), or (b) below would serve for predicting a spectrum for glycine: 

(a) (DRAW (QUOTE C12OOC1N$))
(b) (SETQ GLYCINE (QUOTE C12OOC1N$))
(DRAW GLYCINE)

Quasi-DENDRAL notation is just DENDRAL dot notation (not necessarily canonical) with four changes:

1. dots are replaced by numerals to indicate bonds,
2. the symbol string is terminated with a dollar sign,
3. atom names longer than one character are surrounded by asterisks, for example *CL* for a chlorine atom, and
4. central bond molecules are prefixed with a special character – the value of the variable CENTRALBOND (currently an asterisk).

The output of the PREDICTOR is a list of number pairs, representing the mass-intensity pairs of the predicted spectrum. Peaks below mass 29 are omitted and intensities are adjusted to percent of the base peak (highest peak). Several options are available to the user to help him interpret the program's work. Section 4.5 lists those currently available.

Also, users familiar with the program can change its theory substantially by resetting parameters before calling DRAW. Section 4.2 indicates many of the parameter names and current values as well as brief descriptions of their effects. Since the functions using these global variables are so intertwined, it is impossible to describe the effects in all contexts. Thus it is generally helpful to look at several examples before and after changing parameters.

4.5. Options and how to use them

1. Print the spectrum as a bar graph instead of as a list of number pairs. Give a name of the spectrum to the function PSPEC, e.g.,

(PSPEC (DRAW (QUOTE C1N1C1C1C1C$)))

or (PSPEC (QUOTE ((15 . 20) (29 . 40) (30 . 22) . . .)))

or (PSPEC SPECNAME), where 'SPECNAME' is the name of a list of number pairs.

2. Print an analysis of the last predicted spectrum. This function prints all the number pairs of the spectrum, in order of descending mass units, with a short note explaining the source of the peak.

The synopsis printed by the function SCAN first indicates the structure of the molecule in DENDRAL dot notation, except with numbered atoms replacing the atom-types. For example Cl.=.01 02 C2.N1 for glycine, as given in the quasi-DENDRAL notation above. The mass-intensity pairs for the ions are indicated in order of decreasing masses. An indication of the source of the pair follows each pair:

(a) (MOL ION) following a mass-intensity pair indicates that this is the pair resulting from the unfragmented molecule.

(b) (*RR MOL) indicates that the pair resulted from some rearrangement of the molecular ion.

(c) (C4 1 C3) indicates that the pair resulted from breaking the single bond between atoms C4 and C3.

(d) (C4 1 C3:*RR C4) indicates that after the bond between C4 and C3 was broken, the fragment containing C4 underwent some rearrangement which resulted in the mass-intensity pair on this line.

To obtain this analysis, call the function SCAN after the spectrum has been calculated: (SCAN).

3. Calculate isotope peaks. After the mass-intensity pair calculations have been made for a fragment, the program can generate a cluster of peaks around the original one to account for isotopic variations of the fragment and addition of extra protons to the fragments. This feature is optional since the most significant peaks in spectra often do not include isotope peaks. Set the global variable IPEAKS to T (true) before calling DRAW: (SETQ IPEAKS T).

4. Print an on-line report of progress, including:

(a) the bond under consideration

(b) the rearrangement products

(c) the features considered and the numerical values associated with them during intensity calculation.

To monitor progress in this way, set the global variable SPEAK to T: (SETQ SPEAK T).

5. Include mass units below 29 in the spectrum. Use the function DRAW 100 instead of DRAW as the top-level function.

5. THE EVALUATION FUNCTION

After candidate structures have been generated by the STRUCTURE GENERATOR, the program needs some way to attach a degree of plausibility to each one. The PREDICTOR makes predictions for each one; the EVALUATION FUNCTION must now reflect the degree to which the predictions confirm or disconfirm each candidate hypothesis. Strictly numerical evaluation functions score predicted spectra on the basis of how much they 'cover' the peaks in the original spectrum without adding spurious peaks – perhaps weighting various kinds of failures. But all of these fail to account for the higher theoretical significance of some peaks over others, regardless of the numbers involved. After experimenting with such numerical evaluation functions, their inadequacies became obvious.

The current evaluation function is relatively untried, but its theoretical base is much sounder than that of previous functions. The PREDICTOR now marks various kinds of cleavages and rearrangements as being very significant from a theoretical point of view. For example, the results of alpha-cleavage in ketones, amines and ethers are put on a global list named SIGNIFICANT, together with the results of other theoretically significant peaks in the predicted spectrum. At the end of the PREDICTOR's run this global list remains set for use by the EVALUATION FUNCTION. Evaluation is a two-step process here: (A) reject any candidate whose predictions are inconsistent with the original data, and (B) rank the remaining candidates.

(A) For each candidate molecule the EVALUATION FUNCTION looks in the original spectrum for each member of this list of significant peaks. Either a significant predicted mass point is represented in the original spectrum or it is not. If there is a peak at this mass point, x, and its intensity level is higher than the expected intensity level from an isotope peak (1% of the intensity of the $x-1$ peak times the estimated maximum number of carbon atoms in the $x-1$ peak), then the next significant predicted peak is considered. When the evaluation routine decides that the original spectrum shows a significant peak only because this is an isotope peak, the candidate is rejected. Rejection of a candidate is accompanied by a message explaining which significant peaks were missing from the original spectrum or were present only in amounts expected from isotopic variations, as shown in the examples in the following section. If the significant peak is not present in the original spectrum and other masses in this region were recorded in the original spectrum, then

this candidate is rejected entirely. For example, if the predicted spectrum shows rearrangement peaks at the wrong mass points, it should not warrant further consideration since the theory is strongly violated by that candidate.

When only high mass peaks have been recorded in the original spectrum, as is frequently the case, and one of the significant peaks thus fails to appear in the spectrum, the EVALUATION FUNCTION notes this fact on the list LOWPKS. For example, a significant peak at mass 15 will not be found in a spectrum which starts at mass 40. No candidate is ruled out by the failure to match unrecorded peaks since a more complete spectrum may well include them. On the other hand, there is no assurance that these significant low mass peaks would, in fact, appear if low masses had been recorded.

(B) For each candidate, the list SIGNIFICANT is matched against the original spectrum. If the candidate is rejected, all of the missing significant peaks are printed as justification for rejecting it. The second step of this routine is to rank the remaining candidates, each of which accounts for some of the non-isotopic peaks in the recorded spectrum, but not necessarily all. The best candidate is taken to be the one which accounts for the most peaks, as one should expect. In case of ties, the preferred molecule is the one with the lower number of unrecorded low mass peaks in doubt (as saved on the list named LOWPKS). The rest of the peaks in the predicted and actual spectra are not used at all presently. However, we may want to resolve ties by a numerical scoring of the remaining (non-significant) peaks in the spectra.

Examples of the results of this EVALUATION FUNCTION are shown in the next section.

6. EXAMPLES, SUMMARY, AND CONCLUSIONS

The preceding sections have promised that section 6 would include examples showing the entire operation of the HEURISTIC DENDRAL program. Two simple examples are shown in Tables 9 and 10. These examples seem trivial until one considers the possible list of answers which could have been generated. The total number of structures for the composition C8H16O is about seven hundred 'chemically stable' structures. (Several thousand others were eliminated by BADLIST.) Forty of these are ketone structures, yet the particular spectra (S:09320 and S:09046) enable the program to reduce the output to one and three structures, respectively.

We have been using the program in another mode, namely to direct a search of chemical literature to determine which structural isomers of a given composition have previously been synthesized by chemists. In the case of threonine (composition C4H9NO3), we estimate that there are several thousand isomers (unrestricted by BADLIST). Approximately 750 of these are considered 'chemically stable' but only about sixty of these have been reported in standard chemical references. This disparity has great significance for chemists because of the number of potentially useful compounds that may be found among the 890 'new' structures.

```
(EXPLAIN (QUOTE C8H16O) S:09320 (QUOTE TEST2) (QUOTE JULY8))
*GOODLIST=(*N-PROPYL-KETONE3*)
*BADLIST=(*C-2-ALCOHOL* *PRIMARY-ALCOHOL* *ETHYL-ETHER2*
*METHYL-ETHER2* *ETHER2* *ALDEHYDE* *ALCOHOL* *ISO-PROPYL-
KETONE3* *ETHYL-KETONE3* *METHYL-KETONE3*)
----------

(JULY-4-1968 VERSION)
C*N-PROPYL-KETONE3*H8
MOLECULES      NO DOUBLE BOND EQUIVS
   1.   C=.. O C3H7 CH2.C3H7,
   2.   C=.. O C3H7 CH2.CH. . CH3 CH3.
DONE
----------

(SCORE (QUOTE TEST2) S:09320)
JULY-8-1968
1.) c211oc1c1cc1c1c1c$
((43 . 100) (57 . 88) (58 . 22) (71 . 100) (85 . 88) (86 . 22)
(128 . 14))
2.) c211oc1c1cc1c11cc$
((43 . 87) (57 . 100) (58 . 8) (71 . 87) (85 . 100) (86 . 4)
(100 . 4) (128 . 16))
*THIS CANDIDATE IS REJECTED BECAUSE OF (100).
----------

*LIST OF RANKED MOLECULES:
1. #1.
  S=6.
  P=(57 71 43 85 86 58)
  U=NIL
*1. #N MEANS THE FIRST RANKED MOLECULE IS THE NTH IN THE
ORIGINAL NUMBERED LIST ABOVE. S=THE SCORE (HIGHEST=BEST)
BASED ON THE NUMBER OF SIGNIFICANT PREDICTED PEAKS IN THE
ORIGINAL SPECTRUM. P=THE LIST OF SIGNIFICANT PREDICTED
PEAKS. U=THE LIST OF POSSIBLY SIGNIFICANT UNRECORDED PEAKS
USED IN RESOLVING SCORING TIES (THE FEWER IN DOUBT THE BETTER).
DONE
```

Table 9. An example of HEURISTIC DENDRAL output: 4-Octanone

```
(EXPLAIN (QUOTE C8H16O) S:09046 (QUOTE TEST1) (QUOTE JULY8))
*GOODLIST=(*ETHYL-KETONE3*)
*BADLIST=(*C-2-ALCOHOL* *PRIMARY-ALCOHOL* *ETHYL-ETHER2*
*METHYL-ETHER2* *ETHER2* *ALDEHYDE* *ALCOHOL* *ISO-PROPYL-
KETONE3* *N-PROPYL-KETONE3* *METHYL-KETONE3*)
----------

(JULY-4-1968 VERSION)
C2*ETHYL-KETONE3*H8
MOLECULES      NO DOUBLE BOND EQUIVS
   1.   CH2. . CH2.C3H7 C=.O C2H5,
   2.   CH2. . CH. . CH3 C2H5 C=.O C2H5,
   3.   CH2. . CH2.CH. . CH3 CH3 C=.O C2H5.
DONE
----------
```

Table 10. An example of HEURISTIC DENDRAL output: 3-Octanone

```
(SCORE (QUOTE TEST1) S:09046)
JULY-8-1968
1.) c1lc1c1c1cc2loc1c$
((29 . 100)(57 . 100) (71 . 70) (85 . 40) (99 . 70) (128 . 13))
2.) c1lc1lcc1cc2loc1c$
((29 . 100) (57 . 100) (71 . 100) (72 . 4) (99 . 100) (128 . 19))
3.) c1lc1c1lccc2loc1c$
((29 . 100) (57 . 100) (71 . 87) (99 . 87) (128 . 16))
```

```
*LIST OF RANKED MOLECULES:
1. #2                          3. #3
   S=5.                           S= 4.
   P=(29 99 57 71 72)             P=(29 99 57 71)
   U=(29)                         U=(29)
2. #1                          *1. # N MEANS THE FIRST . . .
   S=4.                               (see Table 9)
   P=(29 99 57 71)
   U=(29)
```

Table 10 (contd.). An example of HEURISTIC DENDRAL output: 3-Octanone

---

We realize that the internal structure of HEURISTIC DENDRAL has not been presented in much detail. No very unusual programming has been employed, however; but we have taken full advantage of the facilities of LISP 1.5. What we have tried to present in this paper is the global strategy of the program. Between the global strategy of a program and its coded functions there are many levels of complexity. We have tried to keep an eye on both extremes and to stay roughly mid-way between them in order to show how some of the heuristics of the program work, how the various subroutines are tied together, and how we plan to expand the program to cover cyclic structures and more classes of acyclic structures.

Recently we have had some ideas of how to rewrite HEURISTIC DENDRAL to separate more completely the model of chemistry from the graph manipulating processes. This will be our next big programming effort. Hopefully the revised program will handle rings without making them special cases. The program's poor handling of ringed structures is now its major deficiency.

In limited areas, the current program performs its two major tasks with a fair measure of success.

(1) Using Lederberg's DENDRAL algorithm plus a theory of chemical stability, the STRUCTURE GENERATOR can construct all acyclic isomers (structural variants) of a given composition, either including or rejecting unstable structures.

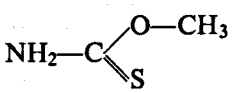
(2) With the more interesting task of explaining the data from a mass spectrometer by finding molecular structures which best account for the data, the program approaches the chemists' level of sophistication only for a few select classes of molecules. Expanding the program to cover more classes depends upon much interaction with chemists, but no new programming strategies are anticipated.

7. A SUMMARY OF THE DENDRAL ALGORITHM

DENDRAL is a system of topological ordering of organic molecules as tree structures. Proper DENDRAL includes precise rules to maintain the uniqueness and the non-ambiguity of its representations of chemical structures. Each structure has an ordered place, regardless of its notation; the emphasis is upon topological uniqueness and efficient representation of molecular structures. The principal distinction of DENDRAL is its algorithmic character. DENDRAL aims (1) to establish a unique (i.e., canonical) description of a given structure; (2) to arrive at the canonical form though mechanistic rules, minimizing repetitive searches and geometric intuition; and (3) to facilitate the ordering of the isomers at any point in the scan, thus also facilitating the enumeration of all of the isomers.

The DENDRAL representation of a structure is made up of operators and operands. The operators are valence bonds issuing from an atom. Each bond looks for a single complete operand. An operand is (recursively) defined as an unbonded atom, or an atom whose following bonds are all satisfied in turn by operands. Hydrogen atoms are usually omitted, but are assumed to complete the valence requirements of each atom in the structure. If the structure has unsaturations (one unsaturation for each pair of hydrogen atoms by which the structure falls short of saturation), these are indicated by locations of double and triple bond operators. Single, double, and triple bonds are represented by . , : , and ; respectively. The operator : may be represented by = and the operator ; by $.

As an example, the molecule



has one unsaturation and may be written in many ways, including:

(1) C.O.C.:NS
(2) C.O.C:.SN
(3) O..CC.:NS
(4) O..C.:NSC
(5) C..:O.CNS
(6) C.:.O.CSN
(7) C.:.NSO.C (canonical)
(8) C:..SNO.C
(9) S:C..O.CN
(10) N.C.:O.CS

Each of these ten notations is a non-ambiguous representation of the molecule. However, proper DENDRAL also specifies that the representation be unique. The key to obtaining the unique or 'canonical' representation is the

recognition of the unique center of any tree structure and the subsequent ordering of successive branches of the tree.

The centroid of a tree-type chemical structure is the bond or atom that most evenly divides the tree. A molecule will fall into just *one* of the following categories, tested in sequence. Let V be the count of non-hydrogen atoms in the molecule. Then either

A. *Two central radicals* of equal count are either (1) united by a leading bond (V is even) or (2) sister branches from an apical node (V is odd); or

B. *Three or more central radicals*, each counting less than $V/2$, stem from a single apical node.

In the first case, the centroid is a bond, and the canonical representation is an operator followed by two operands. In the other two cases the centroid is an atom, and the canonical representation is an operand in the form of an atom followed by two or more bonds and operands. In every case where two or more bonds follow an atom, the operands must be listed in ascending DENDRAL order.

DENDRAL order (or simply 'weight') is a function of the composition and arrangement of a structure and finds its primary use when comparing two operands (radicals). The weight of a radical is evaluated by the following criteria (in descending significance): count, composition, unsaturation, next node, attached substructures.

Count is the number of skeletal (non-hydrogen) atoms. Of two radicals, the one with the higher count is of higher weight.

Composition refers to the atoms contained in the radical. An arbitrary ordering of the atoms makes carbon less than nitrogen less than oxygen less than phosphorus less than sulfur, C<N<O<P<S. (This ordering is alphabetical as well as by atomic number.) When comparing two radicals of the same count, the one with the fewer number of carbons has lesser weight. If carbons are equal, the one with the fewer nitrogens is of lesser weight. And so forth.

Unsaturation counts the number of extra bonds (1 for a double bond, 2 for a triple bond) in the radical, including those (if any) in the afferent link (the bond leading into the radical). Of two radicals, the one with the greater number of unsaturations has the greater weight.

The *next node* or *apical node* refers to the first atom in the radical (the one connected to the afferent link). When comparing two apical nodes, the following three criteria are evaluated (in order of decreasing significance):

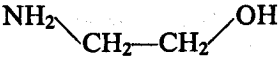
Degree is the number of afferent (attached) radicals. The apical node with the most radicals attached to it has the greater weight.

Composition refers to the type of atom. A carbon atom is the lowest apical node, while a sulfur atom is the highest.

Afferent link refers to the bond leading to the apical node. A single bond afferent link is the lowest, a triple bond is the highest.

If the above criteria fail to determine which of two radicals has the greater weight, then the radicals appendant on the two apical nodes must be arranged in increasing order and compared in pairs. The first inequality in weight of appendant radicals determines the relative weight of the original radicals.

The canonical representation for the molecule in the example given earlier is notation #7. It must be a central atom molecule since its count (ignoring hydrogen atoms) is 5; and the non-terminal carbon atom is the only atom which has all its appendant radicals with counts less than 5/2. Of the three appendant radicals, the one containing two atoms has the highest count and thus is the heaviest. Of the two radicals containing a single atom each, the one with the double bond is the heavier because it has more unsaturations.

Even-count molecules may have a bond for center, if the count of the molecule is evenly divided by cutting that bond. Thus, the canonical form for  is .C.NC.O, a leading bond, the first dot, calling for two operands.

The collection of rules and conventions described above provides a unique and non-ambiguous representation for any non-ringed chemical structure. In addition, the rules also allow us to construct the 'lowest' structure which can be made from a composition (collection of atoms). Once this lowest structure has been made, it may be transformed by a process of rearranging its atoms and unsaturations into the 'next to lowest' structure. This 'incrementing' process may be continued until the 'highest' structure has been made.

REFERENCES AND READING

Lederberg, J. (1964) DENDRAL-64, *A System for Computer Construction, Enumeration and Notation of Organic Molecules as Tree Structures and Cyclic Graphs*, Parts I–V. Interim Report to the National Aeronautics and Space Administration.

Lederberg, J. & Feigenbaum, E.A. (1967) *Mechanization of Inductive Inference in Organic Chemistry*. Stanford Artificial Intelligence Project, Memo No. 54.

Sutherland, G. (1967) DENDRAL–*A Computer Program for Generating and Filtering Chemical Structures*. Stanford Artificial Intelligence Project, Memo No. 49.