A Model-Based Approach to the Teletype Printing of Chemical Structures[1]

RAYMOND E. CARHART

Computer Science Department, Stanford University, Stanford, California 94305

Received November 7, 1975

A Fortran program for drawing chemical structures on the teletype starting from a connection table is described. The program is guided in its task by an internally constructed model of the molecule and is thus freed from the limitations of template-based systems. An outline of the program logic is presented along with an example. Several samples of the program output are included to indicate its general performance level, and the availability of the program is discussed.

In the last several years, a number of computer programs have been developed for structure search and retrieval, for designing chemical syntheses, and for determining molecular structure from spectroscopic data.[2] These programs share a common problem; how does one display structural information to the user? It is always possible to describe a structure by a connection table or a list of spatial coordinates, but this requires a substantial amount of effort on the part of the user to translate from numerical data to a meaningful chemical drawing. The ideal solution to the problem calls upon sophisticated graphics-display terminals to plot the atoms and bonds, and several excellent systems for doing so exist.[3]

However, such graphics terminals are relatively expensive and are not widely available. Thus, for a remote network user of a program, or for a research project with limited funds for special terminals, these graphical display systems may not be useful. In such cases there is a need for a character-oriented display system which "types" the molecule, or at least a topological representation of it, using standard teletype symbols. Building upon concepts developed by Zimmerman,[4] Feldmann[5] has developed one such system which is particularly effective for drawing edge-fused polycyclic ring systems such as steroids. In order to handle the intricacies of atom placement within complex ring systems, this program uses a library of "templates" (standard drawings for individual rings of various sizes) along with a set of rules for combining these templates into complete ring systems. Although this approach can be quite efficient, it is limited in its generality by the size and nature of both the template set and the rule set, with the most difficult cases being bridged polycyclic systems.

Wipke[6] has briefly mentioned a template-free program for typing chemical structures based upon spatial coordinates, but this program makes no attempt to adjust atom positions to give an unambiguous layout of atom and bond symbols in the drawing. Its primary use is in the display of molecules for which the user has explicitly entered coordinates. The program described in this paper uses a method similar to Wipke's when other methods fail (see section IX).

In the design of the user-interface of CONGEN,[7] the constrained structure generator developed by the DENDRAL project at Stanford, it became apparent that a more general approach was necessary. CONGEN is capable of producing such a wide range of unusual cyclic structures, from small and tightly caged molecules to complex polycyclic natural products, that no reasonably small set of templates and rules can be expected to cover them all. Thus a template-free drawing program was developed for use with CONGEN and other DENDRAL programs. In this paper, the central concepts of this new approach are outlined.

I. FORMULATION OF THE PROBLEM

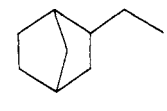
The initial information consists of a list of atom names (assumed for the moment to be single characters—see section VIII) and a list of the bonds interconnecting the atoms, with multiplicities. Typically, hydrogen atoms are omitted to simplify the problem. In the final drawing, one is restricted to a rectilinear grid,[8] each square of which represents one character position. The only directions which can be represented conveniently are integral multiples of 45°, and thus the separations between bonded atoms are restricted to one of these eight basic directions. No two atoms are allowed to occupy the same square, nor are they allowed to occupy squares which are adjacent, either vertically, horizontally, or diagonally (this is a necessary restriction for bonded atoms, because at least one bond symbol must occur between them; for nonbonded atoms, it is included for clarity—see section VII). To avoid confusion, no bond should pass directly over an atom position, though crossing bond symbols are allowed. The crossing of bonds implies that a given square can contain two or more bond symbols; section VIII discusses methods for typing such multiple symbols.

The basic problem, then, is to place the atom and bond symbols within the grid without violating the above restrictions.

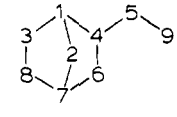
II. OVERVIEW OF PROGRAM OPERATION

The logic of the program was derived from my concept of my own mental processes in composing a chemical drawing in the rectilinear grid. The basic steps were conceived to be (1) form a three-dimensional image of the molecule and view it from its most planar side; (2) determine roughly what the bond distribution around each atom should look like when these bonds are restricted to vertical, horizontal, and diagonal orientations; (3) combine individual "atom pictures", arriving at a set of orientations for the bonds; and (4) attempt the drawing using these angles, adjusting bond lengths as necessary.

Whether or not these steps accurately reflect the human approach to structure drawing, they do form the basis for a successful computer program. The implementation of each step will be discussed in some detail below. A running example, the drawing of 2-ethyl[2.2.1]bicycloheptane (**1**), will be included to clarify the descriptions of the steps. Structure



1



2

2 shows the reference numbering which will be used for the atom positions. This is not the standard chemical numbering, but an internal numbering used by the program.

III. MODELING THE MOLECULE

In the initial stage of the program, a simple computational scheme is used to determine a reasonable three-dimensional

distribution for the atoms in the molecule. The approach used is not intended to mimic nature accurately, as do more sophisticated programs for molecular modeling. Rather, the purpose is to generate a rough, easily computed spatial model which provides a guide to the program when it is considering the layout of atom and bond symbols in the drawing. Starting from an arbitrary initial distribution of atoms, the program iteratively improves the atom positions; at each step, displacement vectors $\mathbf{d}(i)$ are computed for all atoms according to the following formula:

$$\mathbf{d}(i) = A\sum_j (\mathbf{X}_j - \mathbf{X}_i)/(r_{ij}^2 + B)^2 \quad \text{(summation over all } j \text{ not bonded to } i\text{)}$$

$$+ C\sum_k (\mathbf{X}_k - \mathbf{X}_i)[1 - 1/(r_{ik}^2 + B)] \quad \text{(summation over all } k \text{ bonded to } i\text{)}$$

where $\mathbf{X}_i = (X_i, Y_i, Z_i)$ is the position vector of atom i, r_{ij} is the distance from atom i to atom j, $A = -0.3$, $B = 10^{-6}$, and $C = 0.2$.

Contributions from the first summation lead to displacements which separate nonbonded atoms, while those from the second cause bonded atoms to approach a separation of unity. The variable B is included above to prevent the denominators of the fractions from approaching zero too closely, thus avoiding the computational problem of division by zero. The values for A, B, and C, as well as the general form of the expression, were determined through some experimentation with the program, though no systematic optimization of these was attempted.

If, in any iteration, the root-mean-square (rms) displacement exceeds a preset value (1.0), all displacement vectors are scaled so that the rms displacement equals that value. The displacements so calculated are added to the current atom positions and another cycle of refinement begins. The process terminates when the rms displacement drops below a preset value (0.001) or when 100 cycles have been completed, whichever comes first. The three-dimensional coordinates thus obtained are suitable for display using a graphics terminal, though because the model contains no terms related to bond angles, the resulting pictures lack some stereochemical features (e.g., when hydrogen atoms are omitted, methylene groups in acyclic chains are linear rather than bent).

For the purpose of teletype drawing the molecule must be "flattened" into a planar form before further processing. The program does this by first reorienting the molecule so that its most planar aspect (in the least-squares-plane sense) is parallel to the drawing plane (referred to as the x–y plane), and then carrying out a second stage of iterative coordinate refinement, now in two dimensions (x and y) rather than three. This allows for the readjustment of interatomic distances which are foreshortened in the projection of the three-dimensional molecule. Finally, the molecule is oriented in the plane so that its more linear aspect (in the sense of a least-squares line) extends along the x axis, taken as the horizontal axis in the drawing.

Figure 1 shows a projection of the example molecule (**1**) following the three-dimensional modeling, while Figure 2 shows the final, flattened model which will be used in constructing the teletype drawing.

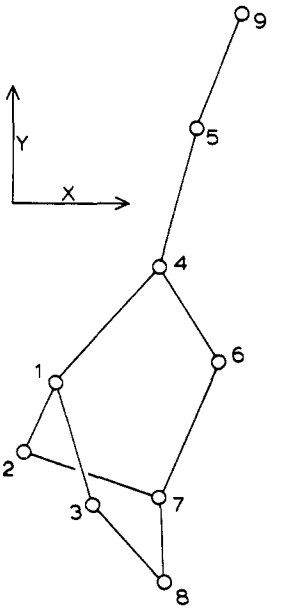


Figure 1. A projection of **1** after three-dimensional modeling. The vectors indicate unit distances.

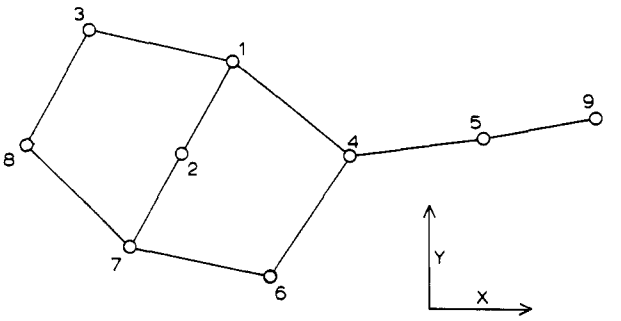
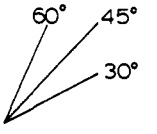


Figure 2. Structure **1** after two-dimensional modeling. Unit distances in the x and y directions are indicated.

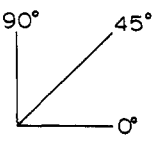
IV. DETERMINING ATOM STATES

Working from the planar x–y coordinates obtained from the modeling stage, the program next determines the possible "states" of each atom, i.e., the possible ways in which the actual bond directions about an atom can be "idealized", without coinciding, to the directions 0, 45, 90, ..., 315° which are available on the grid.[9] The determination of states is controlled by a variable called DELANG, which measures the maximum disagreement between the actual and idealized directions of each bond. DELANG is initially 35° so that, for example, a bond which extends from an atom at 97° may be idealized to any angle between 97° + 35° = 132° and 97° − 35° = 62°. Of course, the only ideal angle within this range is 90°.

The restriction that no two bonds may coincide means that no two bonds may emerge from the same atom in the same ideal direction. In some cases this can be a powerful restriction, because it can eliminate many possible ideal orientations for the bonds. For example, consider the bond distribution about the atom shown below:



The bonds marked 30 and 60° each have two ideal orientations, yet when coincidence is disallowed, the set of three bonds can only have the distribution:



Thus the ideal orientation of each bond is fixed.

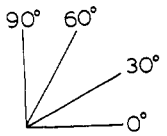
In some cases, ideal distributions for the bonds around one or more atoms cannot be found, for example, in the distrib-

Table I. The Atom States for **1**, Derived from the Planar Model Shown in Figure 2

Atom no.[a]	Possible atom states[a] 1	2	3	4
1				
2				
3				
4				
5				
6				
7				
8				
9				

[a] These atom numbers refer to structure **2**.

ution:



In such cases and under certain other conditions in which the program determines that it needs to consider more atom states (see below), DELANG is increased to 45° and new atom states are calculated. If this does not suffice, the molecule is considered to be "undrawable" (see section IX).

Table I shows the atom states derived from the planar drawing in Figure 2.

V. COMBINING ATOM STATES (ANGLE SELECTION)

In the next stage of analysis, the program seeks combinations of atom states, one for each atom, with the restriction that for every bond, the states of its endpoints must agree as to its orientation (i.e., the bond cannot "bend" in the middle). For example, suppose state 1 in Table I is selected for atom 1. Then only states 1 or 3 may be chosen for atom 3 because they are the only ones in which the 1–3 bond has a direction of 180°, as the first selection implies.

If no such selection is possible, then as above either DELANG is increased to 45° for another try, or if DELANG has already been increased, the molecule is considered to be undrawable.

In most cases, many different state combinations are possible, and the program can collect up to 50 of these, simply ignoring any additional solutions. Each selection determines, implicitly, a consistent (i.e., noncoincident) set of ideal directions to be used in drawing the bonds of the molecule.

Table II. Atom-State Combinations Obtained for **1** Based on the States Shown in Table I[a]

Combination index	States[b] of individual atoms[c] At-om 1	At-om 2	At-om 3	At-om 4	At-om 5	At-om 6	At-om 7	At-om 8	At-om 9
1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	2	1	1	1	2
3	1	1	3	1	1	1	1	2	1
4	1	1	3	1	2	1	1	2	2
5	1	3	1	1	1	1	2	1	1

[a] Only the first five combinations found by the program are given. [b] Atom-state numbers refer to those given in Table I. [c] Atom numbers refer to those in structure **2**.

Table III. The Drawing Strategy Obtained for **1**, Based on the First Atom-State Combination in Table II[a]

Strategy step	Endpoints[b] of bond From	To	Direction, deg	Dist[c]
1	1	4	315	2,3
2	4	6	225	2,3
3	6	7	180	3,2,4
4	7	8	135	2,3
5	8	3	45	2,3
6	3	1	0	(close ring)
7	7	2	45	2,3
8	2	1	45	(close ring)
9	4	5	0	3,2,4
10	5	9	0	2,3

[a] The starting atom is 1. [b] These atom numbers correspond to structure **2**. [c] Measured in grid steps in the indicated direction. The first distance listed is the "best guess", followed by poorer guesses.

In the example, there are over 50 solutions, the first five of which are given in Table II. In this case the first one will lead to a successful drawing.

VI. COMPUTING A DRAWING STRATEGY

When a consistent set of ideal angles has been found, a "strategy" is composed for laying out the drawing in the grid. Each step in the strategy represents one new edge in the drawing, and for bonds which do not close rings, one new atom. In coded form, each strategy element contains (1) the index *i* of the atom from which the new edge is to emanate; (2) the direction (determined from the node state of atom *i*) in which the new bond is to be drawn; (3) the index *j* of the atom at which the new bond is to terminate; and (4) two or three possible bond lengths to try, measured in grid steps in the indicated direction (omitted for ring closures). These steps, together with the index of the starting atom, determine the action of the "layout" segment of the program.

The trial bond lengths are calculated as follows. First, all coordinates in the two-dimensional model are scaled so that the shortest bond, when oriented in its ideal direction, has a length of 2.0 grid units, the center-to-center distance between two atoms separated by one bond symbol. This corresponds to an actual distance of 2.0 if this ideal direction is vertical or horizontal, and $2.0\sqrt{2}$ units if it is diagonal. Then, the "first try" distance for a bond is the integral number of grid units which comes closest to matching the scaled length of that bond when it is oriented in its ideal direction. Again, the length difference between diagonal and nondiagonal grid steps is accounted for. The "second try" and "third try" bond lengths are the second and third closest integral numbers, respectively (if either of these is less than two, it is ignored).

Each strategy step builds upon an atom which is positioned in a previous step. Through some experimentation with the

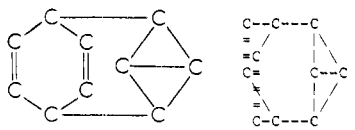


Figure 3. A teletype drawing which could be obtained if nonbonded atoms were allowed to occupy adjacent grid squares.

program, it was determined that in the drawing of complex ring systems the most frequent cause of failure was the inability of the program to close rings in the drawing. Thus, the bonds are listed in the strategy in an order which favors the rapid closure of rings so such failures can be detected early in the layout phase. Acyclic chains are positioned last.

Table III shows the strategy derived from the first atom-state combination given in Table II.

VII. LAYING OUT THE DRAWING

In this stage of the program, the first atom is positioned within an array which represents the grid, with one memory word per square. Each strategy step is considered in turn, with the "first try" distance being used to position each new atom until a conflict is found. The possible conflicts are as follows: (1) a bond or atom already occupies the square in which the new atom is to be placed; (2) an atom already occupies a square in which a new bond symbol is to be placed (note, however, that two or more bond symbols may occupy the same square); (3) there is an atom adjacent (either vertically, horizontally or diagonally) to the square in which the new atom is to be placed; or (4) for ring-closure bonds, there is no way to complete the ring in one of the eight ideal directions.

In case of a conflict, the program backtracks to the most recent "decision point" (choice of distance) in the strategy, erasing the bonds and atoms which have been placed since that step. It then tries the next best distance for that bond and proceeds with the subsequent strategy steps as before.

The restriction represented by (3) above is not necessary to the production of unambiguous drawings. The program always places at least one bond symbol between bonded atoms, so the appearance of two adjacent atoms would imply that they are not connected. However, such drawings are often confusing (e.g., Figure 3) so this restriction is included for clarity in the drawings.

In the case of ring-closure bonds, the program has no choice of location for either endpoint. It thus checks only whether a bond can be placed between the atoms without conflicts 2 or 4, above. The direction specified in the strategy is not necessarily the direction which will be used. Rather, this direction is simply a good "guess" which the program quickly checks before proceeding to a more general test.

This layout procedure can, in principle, explore every trial distance for every bond. Because each bond has two or three possible lengths, the amount of time spent in the layout is potentially an exponential in the number of edges, and can be excessive. To avoid this, the program counts the number of times it needs to "backtrack" to a previous strategy step, and when the count exceeds a preset value (100), then the case is considered to be too difficult to draw. When failure occurs, either because of this excessive backtracking or because all possible bond lengths have been explored, other strategies are tried based on the following scheme:

(1) If more atom-state combinations are available, the next is used.

(2) Otherwise, if DELANG has not yet been increased to 45°, and if there were ten or fewer atom-state combinations resulting from the 35° setting, DELANG is increased to 45° and the computation resumes at the point of atom-state calculation.

(3) Otherwise, the x–y coordinates are considered to be undrawable (see section IX).

Figure 4 shows the steps followed by the program in the drawing of **1**, following the strategy in Table III. The program has no success until it backtracks to the 6–7 bond; selecting a distance of two rather than three for this bond leads to a straightforward solution.

VIII. FORMATTING FOR OUTPUT

Obtaining a grid representation of a molecule is the key problem in teletype drawing, but even after a solution has been found there remain some problems related to formatting. In the grid representation, each atom occupies one square; however, for a flexible drawing system, provisions for at least two- and preferably three-character atom symbols should be included. Also, although the procedure outlined above prevents a given square from containing two bonds with the same orientation, there may be several (up to four) bonds, possibly of differing multiplicities, going in different directions. Provisions must be included for such cases also.

Two formats have been developed for output. One is a compact form which allows one character position for each grid square. For atom names of more than one character, the program searches for blank grid squares adjacent to the one containing the atom and places the extra characters there if possible. To indicate the directionality of single bonds, the standard ASCII characters - (minus), / (slash), ! (exclamation point[10]), and \ (backslash[11]) are used. All double bonds are given the symbol = (equal sign) regardless of direction, and the symbol # (pound sign) is used for triple bonds.[12] If a grid square contains more than one bond, the second and subsequent characters are typed by overprinting the same line (e.g., by using the carriage control "+" in standard Fortran IV). Some terminals are capable of backspacing, and this can also be used for printing several symbols in the same character position.

Figure 5 shows some of the ambiguities which can result from this format. In Figure 5a, one double bond hides another while in Figure 5b two atom names run together as a result of placement of the extra characters. Such cases are relatively rare in the current applications of the program, and the ambiguities can usually be resolved by the user, based on other knowledge about the molecule. The program makes no attempt to avoid these ambiguities.

A second format, which is cumbersome but always unambiguous, is used when multicharacter atom names cannot be placed in the drawing without overlapping each other or other features of the drawing. In this format, a three-by-three cell of character positions is assigned to each grid square. For cells containing atoms, the name is written on the center line. If the appropriate character positions around the periphery of the cell are blank, symbols are included for the bonds which emerge from the atom. For grid squares containing one bond, three bond characters are placed in a line, either vertically, horizontally, or diagonally, depending on the bond's direction. For grid squares containing several bond symbols, this same procedure is used, but the center square receives more than one symbol. These are currently all printed using the line-overprinting or backspace technique, but one could avoid these by simply choosing one of the symbols in the center square arbitrarily. With this format, directional symbols for single bonds are unnecessary, though they are retained to improve the legibility of the drawing. Figure 6 shows an example of this format.

IX. UNDRAWABLE CASES

When the program cannot find a grid representation for the x–y coordinates, it uses a third format in which the teletype

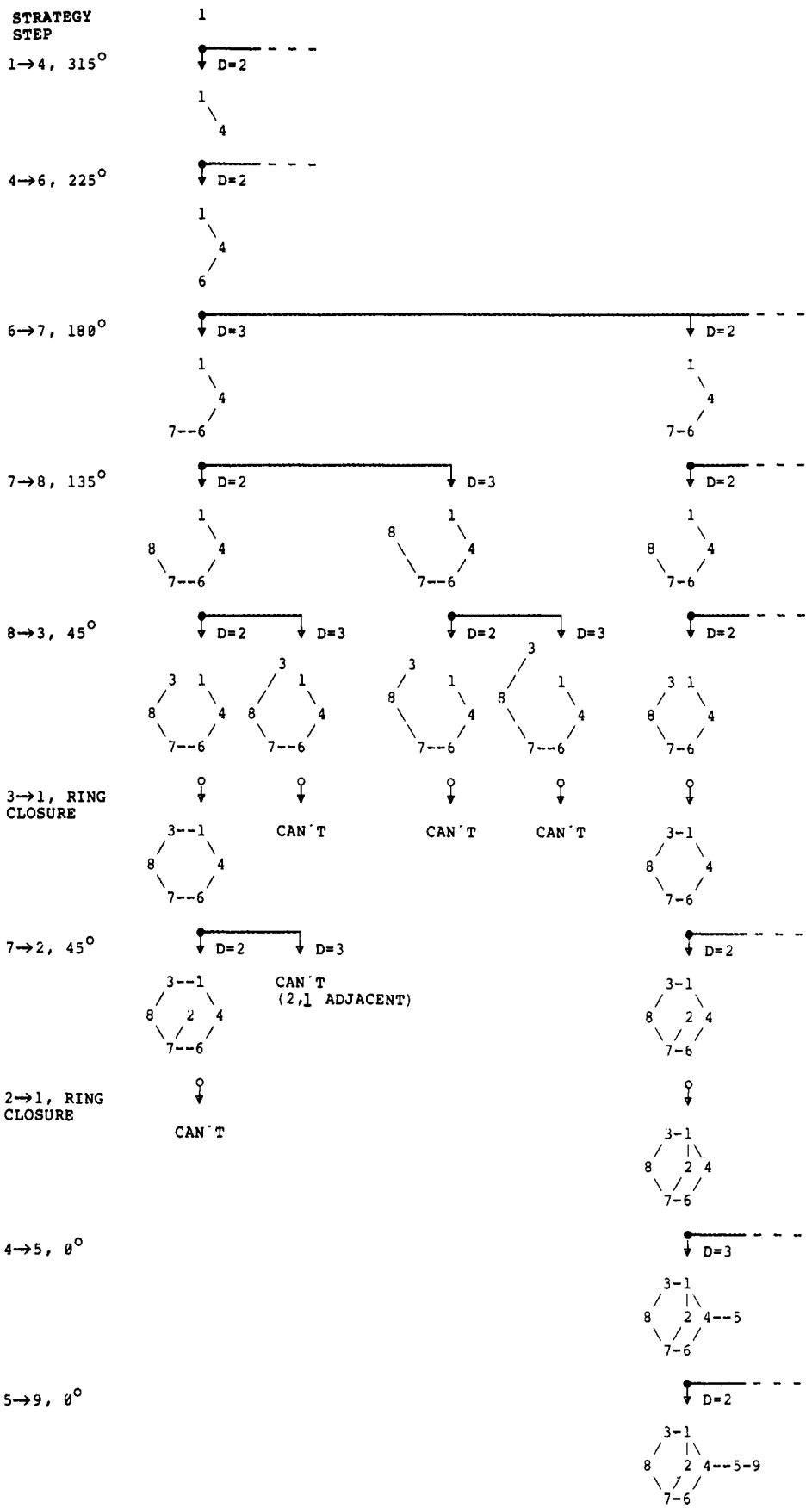


Figure 4. The steps in the final drawing of 1. The strategy is the one given in Table III. The flow of processing is depth–first from the top, with backtracking to the next highest decision point (solid circle) whenever a failure, indicated by the word "can't", occurs. The leftmost branch of each decision point is processed first. The dashed lines indicate unexplored branches of the tree, and the open circles denote ring-closure bonds.

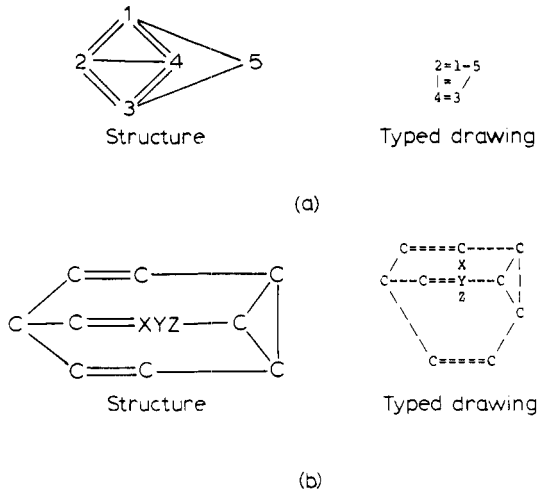


Figure 5. Some ambiguities created by the small output format: (a) one double bond hiding another; (b) atom names run together.

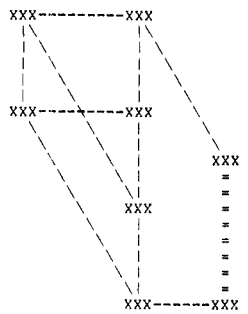


Figure 6. A sample of the large output format.

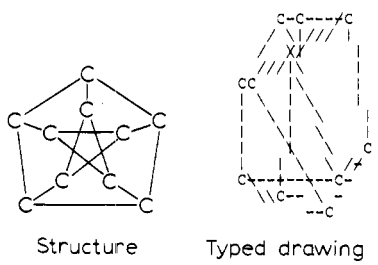


Figure 7. A sample of the "failure" format which is unintelligible.

is used as a "rough plotter" for drawing the actual coordinates. The x–y coordinates obtained from the planar modeling are scaled so that the overall drawing is of reasonable size and the atom symbols are placed in the grid positions which most nearly match their scaled coordinates. The bonds are typed as (perhaps irregular) chains of characters which approximate straight lines between connected atoms. Single bonds are typed using the directional symbols which most nearly match the actual direction. No attempt is made to avoid overlap of atom symbols, or overlap of bond symbols with other bonds or atom symbols. The procedure is similar to that described by Wipke[6] as part of the SECS system. Although these drawings are sometimes nearly useless (e.g., Figure 7), they are often clear enough to communicate the molecular structure (e.g., Figure 8).

X. CONCLUSIONS

Although there are many complex chemical structures for which the program cannot find a grid representation, the vast majority of the cases encountered in the day-to-day use of the program are successfully drawn. Figure 9 gives a representative sample of structures which can be routinely processed.

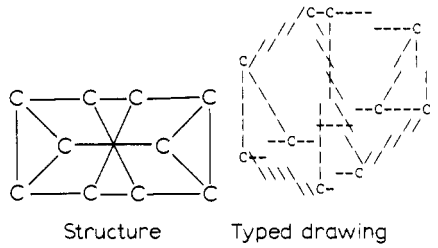


Figure 8. A sample of the "failure" format which is intelligible.

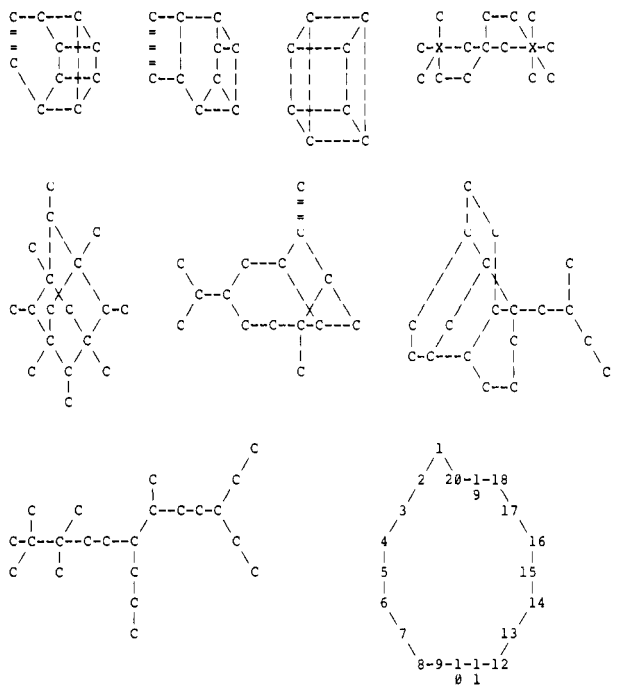


Figure 9. Samples of the program's output.

This work demonstrates that a program does not need a predefined template library to construct useful drawings of chemical structures containing complex ring systems. Rather, the information contained in a simple three-dimensional model (or the corresponding "flattened" two-dimensional one) can be used to guide the program to acceptable solutions. Although the program is somewhat larger and more time-consuming than might be desired, it is hoped that the basic ideas will form a foundation upon which other template-free systems can build.

XI. EXPERIMENTAL

The drawing program was written in Fortran IV on the SUMEX (Stanford University Medical EXperimental) computing facility, a Digital Equipment Corp. KI-10 computer operating under the TENEX[13] operating system. It requires about 60,000 36-bit words of core storage (including all arrays). The program can treat molecules of up to 50 atoms, with a maximum of six neighbors (counting double bonds as two, triple bonds as three, and so on) for each atom. The time required by the program is highly case-dependent, but averages about 0.7 s per structure (including I/O time) on the 217 topological isomers of benzene, and 4 s per structure on a set of 16-atom tetracyclic ring systems. In the benzene case, the program fails to find a grid representation for two of the structures.

A modified version of the program, which incorporates the drawing program described by Feldmann,[5] is being used routinely by several of the DENDRAL programs. The Feldmann program can handle acyclic molecules and certain edge-fused

polycyclic systems quite readily; other cases are passed to the more general but slower program described here. Both versions of the program are available for demonstration and experimentation over a nationwide computer network for those who wish to evaluate the program for their potential use.

Commented listings of the unmodified program (to which the above timing and core-requirement information pertains) are available from the author upon request. Special arrangements can be made for interested parties wishing to obtain a copy of the program in a more computer-accessible form. A version of the program adapted for the IBM 360/67 computer is also available.[11]

REFERENCES AND NOTES

(1) This work was supported by the National Institutes of Health, Grants RR00612-05A1 and RR00758-01A1; the latter in support of the Stanford University Medical Experimental Computer Facility, SUMEX.
(2) For an overview of recent work in these areas, see "Computer Representation and Manipulation of Chemical Information", W. T. Wipke, S. R. Heller, R. J. Feldmann, and E. Hyde, Ed., Wiley, New York, N.Y., 1974.
(3) See, e.g., Abstracts of Papers (COMP Division, Session on Computer Generated Graphics), 169th National Meeting of the American Chemical Society, Philadelphia, Pa., April 6–11, 1975, Port City Press, Baltimore, Md., 1975.
(4) B. L. Zimmerman, "Computer-Generated Chemical Structural Formulas with Standard Ring Orientations", Ph.D. Dissertation, University of Pennsylvania, Philadelphia, Pa., 1971.
(5) R. J. Feldmann, ref 2, pp 55–60.
(6) W. T. Wipke, ref 2, p 153.
(7) R. E. Carhart, D. H. Smith, H. Brown, and C. Djerassi, "Applications of Artificial Intelligence for Chemical Inference. XVII. An Approach to Computer-Assisted Elucidation of Molecular Structure", *J. Am. Chem. Soc.*, **97**, 5755 (1975).
(8) Although the typical spacing of characters in terminal printout is rectangular, we shall treat the grid as being composed of square elements to simplify the discussion. In the actual typed drawings, the "ideal" diagonal directions 45, 135, 225, and 315° are frequently closer to 60, 120, 240, and 300°, respectively. The atom states shown in Table I were composed using a computer terminal, and thus use the latter angles.
(9) Here, as throughout this paper, angles are measured counterclockwise from the positive x axis (the horizontal axis in the drawing plane).
(10) The vertical-bar symbol is available on many terminals and is preferable to the exclamation point. The typed drawings in this paper use the vertical bar.
(11) The "backslash" is peculiar to the ASCII character code, and there seems to be no suitable EBCDIC equivalent. The percent symbol (%) has been chosen arbitrarily for the version of the program adapted for IBM equipment (see section XI).
(12) With atoms of sufficiently high valence, CONGEN can generate structures with bond orders greater than three. The symbols * (asterisk), & (ampersand), and $ (dollar sign) are used for quadruple, quintuple, and hextuple bonds, respectively.
(13) D. G. Bobrow, J. D. Burchfiel, and R. S. Tomlinson, "TENEX, a Paged Timesharing System for the PDP-10", *Commun. ACM*, **15**, 135 (1972).

Selection of Descriptors According to Discrimination and Redundancy. Application to Chemical Structure Searching

LOUIS HODES

Department of Health, Education, and Welfare, National Institutes of Health, National Cancer Institute, Bethesda, Maryland 20014

Received November 14, 1975

Descriptors, for our purposes, will be fragment screens in a chemical search system. Given a file of compounds, the discrimination of a set of descriptors can be defined in terms of their incidence and mutual incidence in the file. A theory is developed which provides both a heuristic for selecting descriptors and a method for evaluating their marginal discrimination. These ideas have been used to generate an efficient screen code for a large file of chemical structures.[1]

1. INTRODUCTION AND BACKGROUND

Descriptors can be thought of as tags or labels applied to objects. They can be used to classify or retrieve subsets of the objects. It is in the nature of descriptors that they are usually used in looking for objects which contain them rather than for objects from which they are absent. In this way descriptors are philosophically different from ordinary binary variables where presence and absence generally carry equal weight.

We develop a theory of discrimination based on the incidence and joint incidence of descriptors, encompassing this nonsymmetrical property of descriptors. This theory is used to evaluate descriptors according to their marginal discrimination capability. This work is especially relevant to chemical structure searching, but it applies to taxonomy and information retrieval, i.e., wherever some selection of descriptors must take place.

This work is also related to feature selection in the area of pattern recognition.[2] However, features tend to occur in the form of variables rather than descriptors, this being sometimes also true in taxonomy work.[3] Descriptors in our sense of the term appear more frequently in document retrieval, but in that field the main problems are those of language, e.g., thesauri. Nevertheless, there have been some attempts to quantify the value of descriptors.[4,5] Closer in spirit to our work is that of Lee[6] and Kryspin and Norwich[7] on the use of information theory to select relevant variables.

Molecular structures supply a superabundance of good descriptors in the form of structure fragments,[8] and it is not easy to produce an appropriate subset. We develop here the information-theoretic concepts of discrimination, redundancy, and marginal discrimination of a new descriptor to supply the rationale for the construction of an efficient effective set of fragment descriptors used for screening a large file of molecular structures.[1]

Computers have facilitated the accumulation of large files of chemical structures, of the order of the hundreds of thousands or several million.[9,10] Because of these large numbers, searching the files according to structural characteristics has become a challenge.

It takes too much time to examine each structure in a file for a required substructure. The general problem is that of matching a graph to an arbitrary subgraph of another graph. It is almost certainly of exponential complexity. There are some good heuristics which take advantage of the nature of most chemical structure graphs, but even so, files quickly outgrew the size for which direct searching would be feasible.

Over years of experience, strategies for substructure searching of chemical files have evolved, some predating the use of computers. Many of these strategies are based on the use of fragments as descriptors, so that the chemical structures