ADABAS requires additional software for thesaurus structuring. Should ECDIN ever want to add a directory function with a built-in thesaurus, it would require a great deal of work. (9) ADABAS is not a text-handling system. ECDIN currently has no plans to incorporate a bibliographic file with character-string searching. Rather its bibliographic file is merely an adjunct to the data files. (10) If ECDIN grows rapidly, updating and searching of the file can become very cumbersome and complex due to the ADABAS pointer system.

Should future changes in ADABAS rectify some of the problems, it is likely that these new changes will be rapidly incorporated by the ECDIN facility. Until then, they are forced to accept redundancy, proliferation in the number of files, and the necessity for preparing numerous macros.

ECDIN'S ROLE IN THE EUROPEAN CHEMICAL INVENTORY

As part of the Council of the European Communities Sixth Ammendment to the 1967 Council Directive on the Classificaiton, Packaging, and Labeling of Dangerous Substances, it was determined that an inventory of chemical substances manufactured and used in the Member Countries was required as a means of defining those new chemicals that were subject to special reporting provisions. The ECDIN facility has been responsible for constructing the European Core Inventory (ECOIN), a compendium of about 35 000 chemicals now on the European market. This became official when it was printed in the *Official Journal of the European Communities* on March 31, 1982. Over the next 9 months, chemical companies are expected to submit data on an estimated 25 000–30 000 additional chemicals not included in ECOIN. The ECDIN staff will be examining these submissions and using them to prepare the final European Inventory of Existing Commercial Chemical Substances (EINECS). This will have the net result of increasing ECOIN's size to about 60 000 unique chemical substances.

REFERENCES AND NOTES

(1) Geiss, F.; Town, W. G. "Presentation at a Seminar Held on Environmental Chemicals Data and Information Network (ECDIN)"; Research Department of the Environment: London, England, July 6, 1977; Report No. 29.

(2) Boni, M.; Geiss, F.; Petri, J. H.; Town, W. G. "The Development of a Data Network on Chemicals and their Effects on the Environment. The Environmental Data and Information Network (ECDIN) of the European Community". In "Proceedings of Eurim II, A European Conference on the Application of Research in Information Services and Libraries"; RAI International Congrescentrum: Amsterdam, 1976.

(3) Bletchly, J. D. "A Report on the Apparent Needs of Possible Customers in the United Kingdom for an Operational ECDIN"; 1979.

(4) Fourneau, J. P. "Les Utilisateurs Potentiels de l'Environmental Chemicals Data and Information Network (ECDIN)"; Paris, 1979.

(5) Tagg, R. M. "A Study of Software Requirements for Scientific Data Bases"; Scicon: London, 1977; Part a.

(6) Tagg, R. M. "Evaluation of Alternative Software for Scientific Data Bases"; Scicon: London, 1977; Part b.

(7) Huitson, P. R. "ECDIN Input Format Manual"; Scicon: London, 1977.

(8) Town, W. G.; Powell, J.; Huitson, P. R. *Inf. Process. Manage.* **1980**, *16*, 91-108.

Conformation Specification of Chemical Structures in Computer Programs

ANDREA L. FELLA,* JAMES G. NOURSE, and DENNIS H. SMITH

Department of Chemistry, Stanford University, Stanford, California 94305

Received September 15, 1982

A computer implementation of conformation specification of chemical structures is described. The implementation produces a canonical name for the conformation given an arbitrary specification of the conformation of the structure. A compact or coded form of the conformation name for use in data bases is also described.

INTRODUCTION

Complete representation of a chemical structure or substructure within a computer program requires that constitution, configuration, and conformation all be specifiable. The DENDRAL group at Stanford University has implemented this representation in three successive stages. Constitutional representations have been dealt with by the GENOA[1] and CONGEN[2] programs which are capable of constrained generation of all constitutional isomers for any molecular formula. Inclusion of stereochemistry was accomplished by the STEREO[3] program which is capable of constrained generation of all possible stereoisomers. The structural representation within the programs is in the form of connection tables augmented with stereochemical information on the chiral centers.

Recently a computer program was written by our group which extends this representation to include conformation. While others have mentioned that their structural representations could be extended to include conformations,[4] a program that actually implements this is not, to our knowledge, available. Note that this program is not a generator of conformations, it expects that defined constitution and configuration are provided, and it also expects information about a specific conformation. From this information, the program produces a canonical (unique) representation of the conformation of the structure (or substructure).[5] This is not an explicit geometric representation, although given standard bond lengths and angles, along with the torsional angles around appropriate bonds, the approximate geometry of the structure could be determined. The uses of such a nongeometric representation include structure registry, search and retrieval systems, and data bases where the conformation of the structure is important, such as proton NMR data bases.

DESCRIPTION OF NAMING SCHEME FOR CONFORMATIONS

In order to determine a canonical name for the conformation of a structure, one must have a naming scheme.[6] The first aspect of the naming scheme deals with the torsional angle about a single bond. This continuous variable needs to be converted into a discrete variable for our purposes. For an example of this property, consider the six possible staggered and eclipsed positions around an sp^3–sp^3 bond (see Figure 1). The positions are described uniquely in terms of the atom numbers by always basing the conformation definition on the

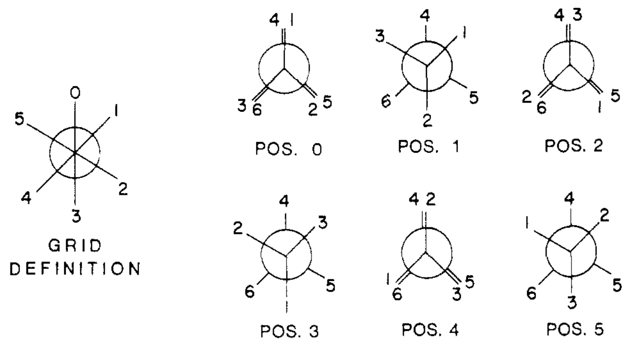


Figure 1. Eclipsed and Staggered rotamers for sp^3–sp^3 Bond with grid definition.

relationship between the two smallest numbered atoms connected to the atoms of the central bond.[4]

With the rear substituent fixed at position 0, the positions are given numbers by letting the rotational position be indicated by where the front substituent falls on the grid. The positions indicated in the diagram are all with respect to the two lowest substituents for the bond. For subsequent use, a priority is defined for the positions, with position 0 having a higher priority than position 1 etc. It is also possible to have a conformation for a bond defined to be either unspecified or freely rotating, with the unspecified one having the numeric value of −1 and the freely rotating one having a numeric value that equals the grid size for the bond.

Other types of bonds have different ranges (or grids) for the torsional positions. sp^3–sp^2 bonds have twelve positions, six staggered and six eclipsed, and sp^2–sp^2 bonds have four, two staggered and two eclipsed.[6]

The second aspect of the naming scheme deals with defining a priority ordering in which to consider the rotational bonds. This allows the rotational positions of the bonds to be considered in a specific order for each structure. sp^3–sp^3 bonds have the highest priority, followed by sp^3–sp^2 bonds, and then by sp^2–sp^2 bonds. This order is broken down into a more refined ordering scheme by having the ordering include the atom types of the bonds within each of the three bond types.

Note that the canonical name of the structure is based on the defined constitutional and configurational name of the structure. Strictly speaking it is not necessary to have these also be in canonical form before determining a lowest conformation name. That is, the canonicalization module will determine the conformation name based on any constitutional numbering and any configurational specification. However, in order for the *structure* to be considered in canonical form, the constitution and configuration need to be in canonical form first.

While it is necessary to convert the continuous variable of torsional angle into a discrete variable, it is desirable that the conformation specification of the structure be as descriptive of the actual structure as possible. The states for discrete torsional angles are allowed to become finer by doubling the grid size as many times as necessary. (In actuality, the program has a built in restriction of four levels of doubling, but this is not inherent in the code; it is an arbitrary restriction that is easily changed.) The doubled (or 12) grid for sp^3–sp^3 bonds has a similar clockwise definition of positions, with each position representing a 30° range of the torsional angle. The ability to double the grid introduces an ambiguity into the conformation position around a bond since position 1 on a 6 grid is not the same as position 1 on a 12 grid. Therefore, the grid size must be stored with the conformation position to remove this ambiguity.

There is also the additional criterion that any conformation defined on a finer grid be reducible to a position on the minimal grid. In order to do this, information must be available (i.e., through some form of input) that specifies whether the position on the finer grid is clockwise or counterclockwise of the defined grid position. This information is referred to as the "side" information.

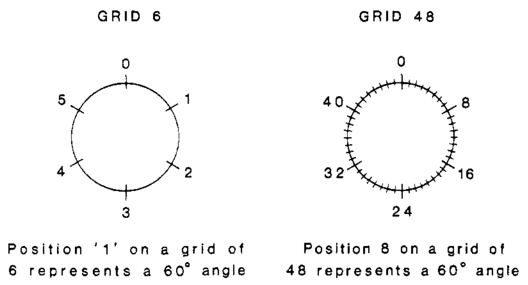


Figure 2. Demonstration of equivalence of grid positions.

CONFORMATION CANONICALIZATION PROGRAM CNFCAN

The CNFCAN program was created from two programs, one from pieces of the STEREO[7] program together with the new code written for the canonicalization of conformations and the other from an input program that creates data that the canonicalizer needs. The pieces of the STEREO program are responsible for setting up the environment to enable conformation canonicalization. While there are many data structures necessary to complete this environment, there are two of primary importance: (1) the canonical configuration of the structure,[7] (2) the symmetry group of the structure.[6]

Since these have been discussed extensively in earlier papers, we will not go into depth with these data structures but will instead discuss their relevancy to the conformation canonicalization algorithm where appropriate.

Program XYZCNV: Creation of Input for the Canonicalizer. Currently, the most convenient way to create input for the canonicalizer involves converting actual *x*, *y*, and *z* coordinates of a structure into the appropriate form for use by the conformation canonicalizer. The program XYZCNV requires coordinate information for each atom as well as connectivity and stereochemical information for the structure. XYZCNV steps through the connection table of the structure, creating an ordered list of bonds that have significant rotation. Then, for each bond in the list, the program uses an algorithm developed by Crippen[8] to determine the torsional angle around the bond based on the *x*, *y*, and *z* coordinates of any two substituents of the central bond and the coordinates of the atoms in the central bond. The angle determined is the clockwise angle (between 0° and 360°) from the rear substituent to the front substituent. The torsional angle is converted into the discrete conformation position by determining which position on the finest grid for the bond most closely matches the angle. Then the grid is reduced to be as coarse as possible. (For example, in Figure 2, position 8 on a grid of 48 is equivalent to a position of 1 on a grid of 6, so the coarser grid representation is chosen.) Information about which side the position tends toward is determined from where the torsional angle actually falls on the grid.

When the torsional angle and rotational position have been determined for each rotatable bond, a file is written that contains connection table and sterochemical information, as well as conformation position information for each bond. (This includes substituents, grid, side, and conformation position. The bond is implied by the position of the information within the ordered list.) This file will be read in by the conformation canonicalization program.

Another possible source of input is direct user definition of the conformation positions through a structure editor. (This capability has not yet been added to our own structure editor).

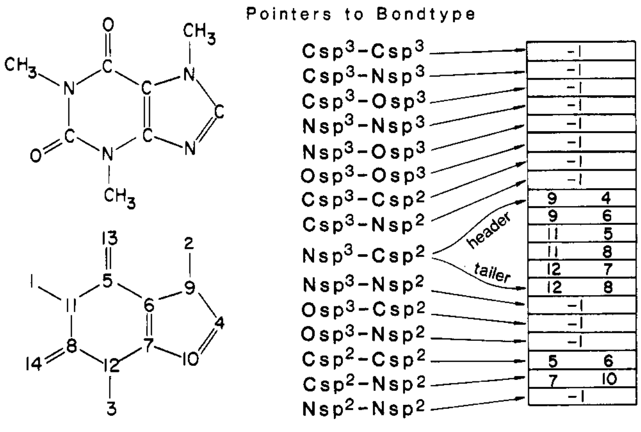


Figure 3. Structure of caffine with ordered list produced from subroutine ROTBND.

The canonicalizer only has the restriction of expecting a certain file format, and a program that writes such a file, given the necessary information, is easily written.

Canonicalization Module. There are three stages to determining the canonical conformation name of a structure. (1) Find the rotatable bonds in the structure (subroutine ROTBND). (2) Determine the properties of each bond necessary to determine the conformation (subroutine BNDPRP). (3) Determine the canonical conformation name of the structure (subroutine CANCON). Each of these will be discussed in sequence.

Subroutine ROTBND. This routine returns an ordered list of rotatable bonds. It steps through the connection table of the structure and checks each bond. Bonds not placed in the list are X–CH_3, X–NH_2, X–OH, and double bonds. The list is segmented into bond types by means of pointers which define the type of bond. The atom types are associated with the bond type. For example, one segment of the list contains all C-(sp^3)–N(sp^2) bonds. (The program currently is restricted to certain types of bonds, but it can be altered to include others.) The bond itself is represented as a single computer word with each of the bond in a half-word. Within each segment the list is simply placed in numerical order. (See Figure 3 for an example of a structure with its associated ROTBND list). Note that when the bond is between like atoms, the lower numbered of the two is placed in the left half-word, but when the bond is between unlike atoms, the order of the atoms in the word is taken from the order of the atom type in the segment pointer. Also note that empty segments (unrepresented bond/atom types) contain a –1.

The fact that a single computer word is used to represent the bond places a restriction on the number of atoms the program can handle. Our implementation is on a 36-bit DEC KI-10 computer, which restricts the molecule size to less than 2^{18} atoms, which is not a significant restriction. If, however, the program were run on a smaller (say 16 bit) machine, the restriction of the molecule size is to less than 256 atoms. The program code is flexible enough to easily be modified to fit the word size of the machine being used.

This data structure is of primary importance to the canonicalization procedure. First, it defines the ordering of bonds necessary for deriving a canonical name,[6] second, the data structures created in succeeding modules are based on this one, and third, it is necessary to recreate this list to associate the defined conformations with their correct bonds. The XYZCNV program, for example, uses this subroutine to create its ordered list of bonds. Upon entering the canonicalization program, the routine must be called so the appropriate conformations will be associated with the correct bonds as the information is read from the data file.

Subroutine BNDPRP. This routine goes through the ROTBND list and determines for each bond the information needed to determine the "lowest" representation for that bond. Conceptually, the information that is needed is a table that defines how many rotational positions on the grid (clockwise) are between the substituents of each atom and the lowest numbered substituent of that atom. In terms of data structures this information is broken into two pieces: (1) a table to order the substituents of each atom and (2) the "rotational" direction of each atom.

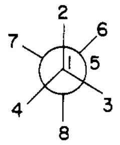


Figure 4. Example of rotational direction of atoms. Atom 1 in bond 1–5 has an increasing order of substituents (2, 3, 4) oriented in a clockwise fashion; rotational type = 1. Atom 5 in bond 1–5 has an increasing order of substituents (6, 7, 8) oriented in a counterclockwise fashion; rotational type = 1.

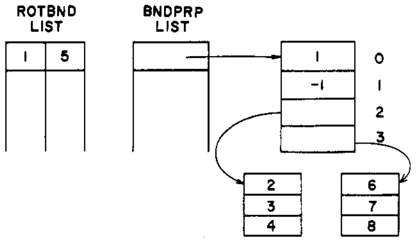


Figure 5. Partial list of bond properties returned from subroutine BNDPRP. Cell 0 represents the rotation type for atom 1. Cell 1 represents the rotation type for atom 5. Cell 2 represents the pointer to the order of substituents for atom 1. Cell 3 represents the pointer to the order of substituents for atom 5.

By rotational direction we mean whether the increasing order of substituent numbers (not involved in this bond) are ordered clockwise (rotational direction = 1) or counterclockwise (rotational direction = –1). The definition of this term also depends on the direction one "looks" down the bond (see Figure 4.) The atom in the left half-word of the ROTBND list is always considered to be the front atom, and the other is the rear. Obviously, the determination of 'rotational' direction about a stereocenter will involve the configuration of that stereocenter, and this is where the canonical configuration is used (see Figure 5 for a partial BNDPRP list).

Subroutine CANCON. The actual canonicalization routine utilizes the symmetry group of the structure.[6] For each rotational symmetry operation, the canonical conformation name of the structure is determined. As each new name is determined, it is compared with the previous name. If it is a lower name (i.e., if an element by element comparison, from the top of the list down, found that a particular conformation for a bond is lower), then the new conformation name replaces the old. The lowest conformation for a particular bond is the one whose grid position determines the smallest angle between the 0 position and the defined position.

If the symmetry operation is ignored for the moment, the method for determining the lowest conformation name of a particular bond is relatively simple. Available from subroutine BNDPRP are the number of clockwise positions that any substituent needs to rotate to be in the place of the lowest substituent. This number is determined for each of the substituents obtained from the entered conformation data. The difference (in modular arithmetic, using the grid size as the base) between the rotational number for the front atom's substituent and the rotational number for the rear atom's substituent effectively determines the number of clockwise grid positions the front substituent needs to rotate in order that the original front and rear substituents end up in the conformation of the lowest front and rear substituents. This difference

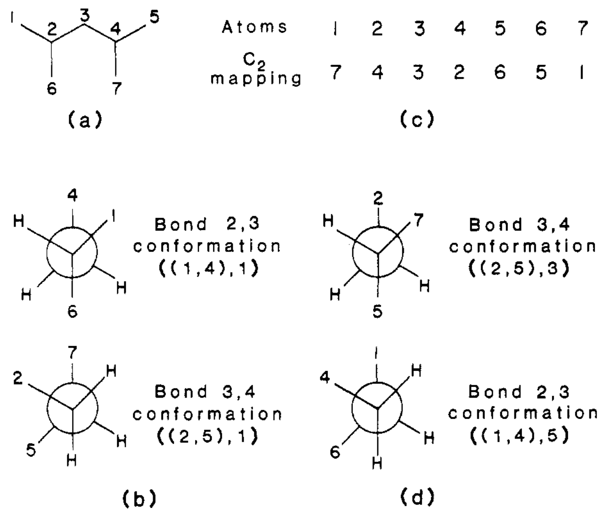

1 2 3 4 5
6 7
(a)
Atoms 1 2 3 4 5 6 7
C2 mapping 7 4 3 2 6 5 1
(c)
Bond 2,3 conformation ((1,4),1)
Bond 3,4 conformation ((2,5),1)
Bond 3,4 conformation ((2,5),3)
Bond 2,3 conformation ((1,4),5)
(b)
(d)

Figure 6. (a) Structure of 2,4-dimethylpentane. (b) Defined configurations and conformations for rotatable bonds. (c) Atomic mapping of rotational symmetry operation. (d) Effect of symmetry operation on defined conformations.

quantity is added (again modular, base grid) to the original entered conformation position to determine what the conformation is when defined by the two lowest substituents.

As an illustration of this subroutine, an example will be carried through. Consider the structure of 2,4-dimethylpentane (see Figure 6a) with configurations and conformations as defined in Figure 6b. Assume the "user"-specified conformation definition is as follows:

for bond 2–3,
atoms 6 and 4 are in conformation 3 [i.e., atoms 6 and 4 fall into grid position 3; compact notation ((6,4),3)]

for bond 3–4,
atoms 2 and 7 are in conformation 5 [i.e., atoms 2 and 7 fall into grid position 5; compact notation ((2,7),5)]

Following the calculations described above (refer to figure 6b), one obtains the following. (1) Atom 6 has four clockwise grid positions to rotate into the position of the lowest substituent. (2) Atom 4 is the lowest substituent; it has zero positions to rotate. (3) The subtraction $4 -_{(6)} 0 = 4$ indicates that atom 6 must rotate four positions in order that substituents 6 and 4 be in the conformation of the lowest substituents 1 and 4. (4) The originally defined conformation was 3, hence $3 +_{(6)} 4 = 1$, and a conformation of 1 is determined for the lowest two substituents of the bond.

A similar calculation can be made to determine that there is a conformation of 1 for substituents 2 and 5 of bond 3–4. So the conformation name for the structure using this symmetry operation is 1,1.

If the structure has no symmetry, this is all that needs to be done to determine the canonical conformation name of the structure. For structures with rotational symmetry, there are certain numberings of the structure that may produce different conformation names even though the conformations are identical. For this example, consider the atomic permutations of the symmetry group C_2, as shown in Figure 6c. Consider a defined conformation of

for bond 2–3, conformation ((1,4),2)

for bond 3–4, conformation ((2,5),3)

and an equivalent conformation by symmetry

for bond 3–4, conformation ((7,2),2)

for bond 2–3, conformation ((4,6),3)

Table I. Table of Coded Character Representations for Conformations for the First Shell Level for the Coded Canonical Conformation Name

grid 4 (sp^2–sp^2 bond)	grid 6 (sp^3–sp^3 bond)	grid 12 (sp^3–sp^2 bond)	
B - 0	F - 0	L - 0	R - 6
C - 1	G - 1	M - 1	S - 7
D - 2	H - 2	N - 2	T - 8
E - 3	I - 3	O - 3	U - 9
A - unspecified	J - 4	P - 4	V - 10
Z - freely rotating	K - 5	Q - 5	W - 11

The effect of symmetry operation is to map bonds and their substituents onto new bonds and substituents. This new mapping is shown in Figure 6d. The original conformation specifications change according to this new mapping

for bond 3–4, conformation ((2,5),3)

for bond 2–3, conformation ((1,4),5)

With this information, the calculations can be performed as above to determine what the lowest conformation is for this symmetry operation. In this case, the lowest conformation is specified without having to acutually carry out the calculations, and the conformation name for the structure with this symmetry operation is 5,3.

Now the two determined conformations are compared to find the lowest or canonical representation for the conformation. The order for the comparison (as defined by the ROTBND list) is bond 2–3 and then bond 3–4. The conformation of 1,1 is compared with the conformation of 5,3, and the former is chosen to represent the conformation of the structure and is the canonical conformation name of the structure.

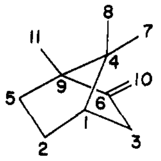
USE OF CANONICAL CONFORMATION IN DATA BASES

In order to make the canonical conformation name useful for database purposes, one needs a compact or coded form of the canonical name. The coded canonical conformation name must be used with a coded canonical representation of the entire structure. The form of the structure code including constitutional and configurational information that was used as a prototype has been described.[9,10]

First the coded canonical conformation name will be described apart from the structural code. Integration of this code into the overall structural code will be described later.

It is important that the coded canonical conformation name be compact, but there are several considerations that influence how compact the name can be. (1) It is useful to impose a shell structure on the coded canonical conformation name. The shell structure breaks the conformation code into "shells" of successively finer levels of conformation specificity. (2) The search and comparison functions should allow an unspecified conformation for a particular bond to match any conformation for that bond. (3) It is desirable to have a code that can be translated back into the original data structures.

The coded canonical conformation name takes the form of a string of ASCII characters, with the shell levels of the code separated by a reserved character; a period. The first shell of the coded canonical conformation name contains the coarsest conformation information for all rotatable bonds (i.e., what the conformation reduces to on the lowest grid for the bond). In the first shell level the conformation for each bond is represented by a single character. The character encodes both the grid size and the position on the grid (see Table I). A lower case character in the first shell indicates that more information is available for that bond (i.e., finer grid position has been defined for this bond). Successive shells contain the finer grid specification. The order in which bonds and



a.

ROTBND list	grid	confo
csp3-csp3		
1-2	24	5
1-3	24	19
1-4	6	3
2-5	6	0
4-9	6	6
5-9	48	6
csp3-csp2		
3-6	96	1
9-6	24	21

b.

```
gkIFKglv.gcl.9b.A
```

c.

```
0/3/224|gk.q9/27114)5)6|gkIFKg.q6.95.2/01|gkIFKglv.qcl.9b.A
```

d.

Figure 7. Integration of coded canonical conformation name into prototype code: (a) Structure of camphor as numbered by substructural coder (focus on hydrogen at atom 1); (b) ROTBND list for camphor with canonical conformation. (c) Conformation code for entire structure. (d) Conformation code integrated into substructure code.

characters are associated is determined by the ROTBND list.

Each of the following shell levels contains the information needed to determine the conformation position on a doubled grid size. Here the side information is needed to break the conformation specification into shells. Since four levels of doubling are allowed, there will be at most four conformation shell levels. In these shells, the information is packed tightly. The first packet of information is associated with the left-most character in the first shell that indicated more information was available. Each packet is three bits of information, with the left-most bit again indicating whether the bond has yet a finer conformation specification. The other two bits indicate whether to add 1, 0, or −1 to the doubled position indicated from the previous shell. Two three-bit packets are put together into a six-bit segment that is written out in base 64. (There is again a look-up table which translates numbers 0 to 63 into a single character.)

The shell structure imposed on the conformation has several advantages. Since each rotatable bond of the structure is represented by a single character, comparison at this shell is simple. It also allows for matching an unspecified conformation for a bond to any other conformation for that bond. Also note that given the ROTBND list, translation back to the original data structures can be accomplished provided the translation tables are available. The main advantage of having a shell-structure conformation code is that it allows matching to occur between structures that are conformationally close. A comparison function that attempts to match structures at successively finer levels of the conformation code has been written, and it returns the number of conformation shells, which match.

Figure 7 shows how the structure of camphor is treated by the conformation canonicalizer and the data base coder. Take bond 1–2 for an example. The grid is broken into 24 parts, and substituents 3 and 5 are in the fifth grid position. This represents a bond angle range of 67.5–82.5°. The grid indicates that information for this bond will be contained in the first three shells of the conformation code (see Figure 7c). The first "G" in the conformation code is the coarsest conformation description for this bond.

The prototype substructure code is also a shell-structured code,[9] with each shell representing approximately one bond radius further from the focus. (The focus is the atom around which the coding centers, usually a resonating nucleus.) At most five shells are coded. To produce a coded substructural name for a data base, it is necessary for the structure to be renumbered so the numbering will be canonical for each bond radius away from the focus.[9] The coded canonical name uses this new numbering in determining the list of rotatable bonds. To integrate the coded canonical conformation name into the prototype code, it is necessary to check for conformations at each substructural shell level and encode them if there are any rotatable bonds. For example, consider the second substructural shell level coded from the hydrogen on atom 1 for structure camphor (see Figure 7). The atoms included in this shell are 1, 2, 3, and 4. Two bonds can be coded for conformations, bond 1–2 and bond 1–3. Bond 1–4 cannot be coded because information necessary for defining which substituent will be used to code the conformaton is not available (i.e., more than one substituent is connected to atom 4, and since these substituents are not available in this shell, a conformation cannot be coded). At each substructural shell level, the entire conformation for the substructure to that shell must be coded. Adding a shell level (or increasing the bond radius from the focus by one) may alter the symmetry of the partial substructure being considered, and any symmetry change affects the conformation code for the entire substructure. The conformation code (if there is one for the shell) is placed at the end of each level's code, preceded by a delimiting character. Note that it is highly unlikely that there will be any conformation at the lowest (zero, first, and second) shell levels. See Figure 7d for an example of how the coded canonical conformation name is integrated into the prototype structural code. Each successive bond radius from the focus (in this case, the hydrogen on atom 1) is coded between the slashes, with each slash indicating a new shell level. The conformation information for the shell is put at the end of the shell, with a vertical bar indicating where the conformation part of the code begins. Configuration information is delimited by a plus sign. (In this example, the configuration of the structure is not explicitly represented in the code because the configuration at all stereocenters happens to be zero.[9]) The use of the delimiter for both configuration and conformation allows these pieces of the code to be ignored if so desired. This allows the user to control how specific the structural matches are when searching through a data base.

EXPERIMENTAL SECTION

These programs are implemented in the ALGOL-like BCPL programming language on a Digital Equipment Corp. KI-10 computer at the SUMEX-AIM facility at Stanford University. The programs are available to an outside community of users, by request to the authors, via an international computer network, to the limits of available resources.

ACKNOWLEDGMENT

We thank the National Institutes of Health (Grant No. RR-00612-12) for their generous financial support. Computer resources were provided by the SUMEX facility at Stanford University under National Institutes of Health Grant RR-0785.

REFERENCES AND NOTES

(1) Carhart, R. E.; Smith, D. H.; Brown, H.; Djerassi, C. *J. Am. Chem. Soc.* **1975**, *97*, 5755.
(2) Carhart, R. E.; Smith, D. H.; Gray, N. A. B.; Nourse, J. G.; Djerassi, C. *J. Org. Chem.*, **1981**, *46*, 1708.
(3) Nourse, J. G.; Smith, D. H.; Carhart, R. E.; Djerassi, C. *J. Am. Chem. Soc.* **1980**, *102*, 6289.
(4) Wipke, W. T.; Dyott, T. M. *J. Am. Chem. Soc.* **1974**, *96*, 4834.
(5) Unless otherwise specified, structure will mean structure and/or substructure.
(6) Nourse, J. G. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 168.
(7) Nourse, J. G.; Carhart, R. E.; Smith, D. H.; Djerassi, C. *J. Am. Chem. Soc.* **1979**, *101*, 1216.
(8) Crippen, G. M. "Distance Geometry and Conformation Calculations"; Research Studies Press: 1981.
(9) Gray, N. A. B.; Nourse, J. G.; Crandell, C. W.; Smith, D. H.; Djerassi, C. *Org. Magn. Reson.* **1981**, *15*, 375.
(10) Egli, H.; Smith, D. H.; Djerassi, C. *Helv. Chim. Acta*, in press.