Exhaustive Generation of Organic Isomers. 2. Cyclic Structures: New Compact Molecular Code

M. L. Contreras,* R. Valdivia, and R. Rozas

Chemistry Department, University of Santiago, Casilla 5659, Santiago 2, Chile

Received March 17, 1992

A selective generation and enumeration program for cyclic isomers, CAMGEC, is presented. The only input to this system is the molecular formula which allows for calculation of isomer characteristics such as number of rings or multiple bonds or a combination of the two. Generation could be done on one of these options or on all of them. Exhaustive generation process is based on graph theory and on a new concept of tuple notation. A filtration process, based on a decomposition–recomposition principle, for discarding repetitive structures is applied over trees specially defined for cyclic compounds that include multiple bonds and heteroatoms with multiple valencies. Selective generation could be done over one or over all of the following parameters: (a) isomer characteristics, (b) one or more selected patterns, and (c) level or degree value of the root vertex. CAMGEC's performance shows its potential applicability in areas such as molecular design, organic synthesis, and structure elucidation.

INTRODUCTION

Many algorithms and ring sets have been devised to represent and handle chemical structures.[1–6] Graph theory has proved to be very helpful in the study of these problems.[7–15]

One important application of graph theory to the enumeration and generation of acyclic systems consists of the *N*-tuple concept developed by Knop et al. in 1981.[16] Later, Randić et al., in 1988,[17] extended that concept to the generation of a compact molecular code for cyclic systems. That work was oriented to benzenoid structures, represented by polyhex graphs.[18–20] The principal strategy was to transform the structure from cyclic to acyclic by excising selected vertices in a polycyclic graph. A spanning subtree results for which an *N*-tuple is adopted. Then such incomplete code is augmented by listing of adjacencies for the vertices, which represent ring closures. These authors[17] claim that molecular codes are applicable to all kinds of structures, but there is no mention in their work about the corresponding treatment for the presence of heteroatoms or multiple bonds in the molecule.

Since the importance of benzenoid compounds for human health is high,[21–23] many efforts have also been done for their algorithmic and chemical graph treatment.[18–20,24] Balaban, in 1971, defined the first molecular boundary code for them.[25] Later, a related code was developed[18] as an extension of the *N*-tuple code. The boundary code is of exclusive application for benzenoids, and it is not extendable to other systems. The boundary code lists the directions as one moves clockwise along the perimeter of the polyhex starting at the left-most edge. By convention the ring is oriented in such a way that the left-most edge is vertical. The resulting code becomes lexicographically maximized.

Molecular codes[17,25,26] are invaluable in computer systems for chemical documentation.[27,28] For the purpose of retrieval, data for individual structures can be stored and manipulated in machine-readable files.[29] Also that could be extended for chemical reaction types,[11,12] for computer-assisted synthesis design,[9] for structure–property and structure–activity studies, for structure elucidation,[30] for enumeration and generation[31,32] of isomers, and so forth.

Cyclic systems, however, have not been described yet in terms of tuple arrays in spite of the advantages the tuple notation can offer.[26]

In this paper a computer assisted molecular generation and counting system, CAMGEC, for cyclic isomers is presented. The method is based on a new concept of compact molecular code for polycyclic systems which is submitted to the necessary filtering steps with the aim of efficient avoidance of duplication during the generation process. Special trees are defined for representing cyclic graphs, and care is taken about the occurrence of both heteroatoms of multiple valencies and multiple bonds. Each step of the generation process is dynamically controlled and addressed to the formation of structures consistent with the input information and pattern requirements. The system allows for a selective exhaustive and irredundant generation process. Examples that can be used in molecular design, organic synthesis, and structure elucidation are presented.

DESCRIPTION OF THE CYCLIC ISOMER TUPLE NOTATION

Cyclic isomers are considered to be formed by atoms belonging at least to one cycle. Atoms that join two cycles like the central C atom in dicyclopentylmethane also could form part of these isomers. No branches are included. For substituent consideration, special atoms are defined as having a consistent valence. For instance, a carbon having a nitro group or any other functional group as a substituent could be defined as a Cx atom having three possible available valences for participating in the formation of any cycle. In this way the valence occupied by the substituent is taken in consideration. In this work, cyclic systems will be considered isolated from acyclic systems.[33] Stereoisomerism will be not considered.

Molecular cyclic structures can be represented by graphs with as many nodes as there are atoms in the molecular formula, as is normal in organic chemistry.[34–36] Hydrogen atoms are excluded. Graphs can be represented by trees, and trees can be represented by tuples. So, finally, graphs can be represented by tuples.

Each node of a graph, of a cyclic isomer, has been assigned an identification number. This number coincides with a corresponding node number in the tree that represents the graph (see Figure 1). The branch '1,2,3,4,1' represents the four-member cycle and the branch '1,2,5,1' represents the three-

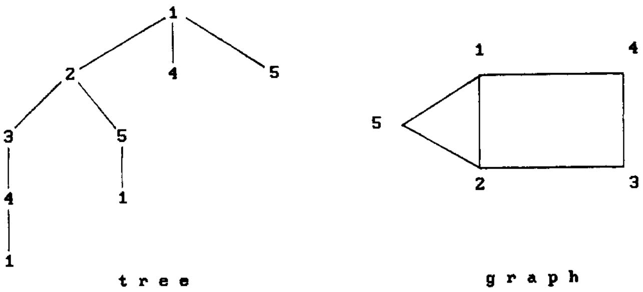

Figure 1. Scheme of a tree and its corresponding graph.

member cycle as it will be explained later. Once the tree is constructed, a particular tuple is used for the mathematical description or representation of the tree.

A tuple is defined[16] as an array of integers in such a way that a K-tuple $(a_1, a_2, ..., a_K)$ of integers is lexicographically smaller than an L-tuple $(b_1, b_2, ..., b_L)$ if there exists an index j with $1 \leq j \leq L$, so that $a_1 = b_i$ for $1 \leq i \leq j$ and either $j = K+1$ or $a_j < b_j$. Each of the components of the tuple represents a node of a tree.

Many different tuples can be defined for describing a single graph or a tree in the function of the analyzed tree; tree definition depends on the chosen root vertex of the tree, the kind of nodes, and the nature of the links between adjacent nodes (as it will be seen later). There is only one tree for a particular graph that can be represented by the lexicographically largest N-tuple. This tuple corresponds to the canonical representation, and it is used in the whole generation process.

Trees have a root and a certain number of branches that depend of the degree value of the root node. For instance, a root node with degree 3 will have three branches joined to it. The root is selected as having the largest degree, and it is said that it is the father of each of the nodes directly linked to it in the tree. Each one of these nodes are fathers of the next adjacent components in the tree, and so on. Trees so constructed describe the existent sequences of nodes in the graph.

When a whole cycle sequence has been completed, the corresponding branch in the tree gets its end at a leaf-node (node without any son), whose number will be the same that one already used in another part of the tree, normally at the beginning of that branch. Here a reciprocal relationship is established between each leaf-node and its father, because in another part of the tree a branch will end in a leaf-node whose relationship with its father will be just the inverse of the described situation. The sense of these relationships is for taking into account the fact that one node in a cycle could be father or son of its neighbor depending on the sequence direction, i.e., clockwise or anticlockwise. As a consequence two leaf-nodes of a tree are needed to describe one cycle of the graph. One of them will occupy the final point of a branch, it will be numbered as **u** and it will keep a relationship with its father which will be numbered as **v**. At the same time, in another part of the tree, a node **u** will be the father of a leaf-node **v** and that will indicate formation of a cycle. This is depicted in Figure 1, where four leaf-nodes are present in the tree, and consequently, there are two cycles in the graph. Nodes 4 and 1, and also 5 and 1, evidently indicate cycle presence.

Then, a typical tuple description of that tree is done.[16,26] In this case the tuple is of the type (3,2,1,1,0,1,0,0,0). As it was expected,[7,16,26] the tuple has nine components (one for each node of the tree), and the sum of all of them is 9 − 1 = 8.

So, tuple notation for cyclic isomers as it was developed in this work differs from the one for acyclic isomers,[26] at least in the following related points:

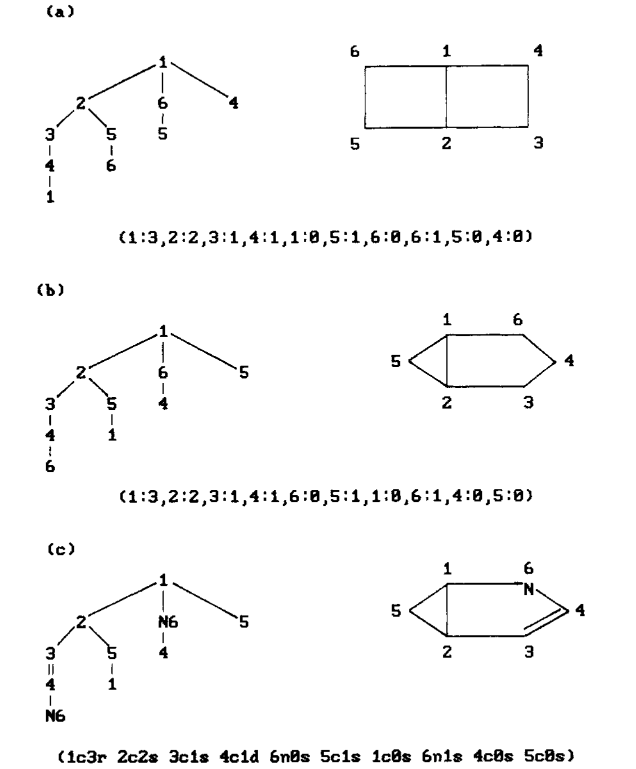

Figure 2. Extended N-tuple notation. Cases a and b use a simplified notation. Case c shows the N-tuple notation of a bicyclic graph including a heteroatom and a multiple bond.

(a) For cyclic isomers the number of nodes of the tree is larger than the number of vertices of the graph. For acyclic ones these numbers are coincident.

(b) Trees for cyclic isomers are numbered according to related numbers of the corresponding graph. Trees for acyclic isomers have no need for an identification number.

(c) For both types of isomers, trees are constituted by a root and some branches that end in leaf-nodes. In the case of cyclic isomers, leaf-nodes are used to describe cycle formation. For acyclic isomers, leaf-nodes are vertices that just determine the end of a branch or the end of a chain of the graph.

An additional parameter, the number assigned to each of the tree nodes, is included, in agreement with the graph numbers, for defining each of the tuple components. In this way the information that each tuple component carries is ordered and used as in the following sequence:

(i) node number
(ii) atom symbol
(iii) degree
(iv) type of link with its father

In Figure 2 some schematic examples are presented. Cases a and b use a simplified notation because they do not contain hetereoatoms or multiple bonds, meanwhile case c has an N atom and a double bond.

At this point, it is also important to define the number of cycles of a graph. It corresponds to the number of single cycles excluding the cycles that could be defined by the union of two or more previously described cycles. The union of two cycles is understood here as the process that allows them to share one or more edges with the aim of forming a new cycle (condensed or fused cycles).

For instance, in Figure 3 a graph having four cycles is

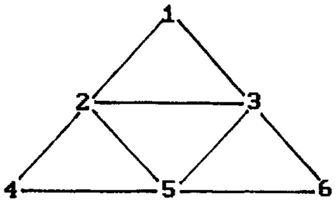

Figure 3. Graph with four cycles.

represented. Any additional one could be described by the union of two or more of the single cycles already defined (there could be a total of 11). If the four described cycles are (1,2,3,1,), (2,4,5,2,), (2,3,5,2), and (3,5,6,3), then the cycle (1,2,5,3,1) should be discarded since it could be constructed by the union of cycles (1,2,3,1) and (2,3,5,2).

GENERATION AND ENUMERATION OF CYCLIC ISOMERS

The generation process of cyclic isomers is based on graph theory and tree representation. Two principal aspects have to be taken in consideration: (i) the generation of irredundant cyclic isomers should be an exhaustive process and (ii) a filtration procedure should be able to efficiently eliminate duplicate structures. Both processes should be applied in a sequential way over a single structure in dynamic memory. A suitable notation for correctly representing structural particularities such as the presence of heteroatoms and multiple bonds in the cyclic structure was used as described above. The generation process consists basically of three main steps:

(1) cyclic skeleton generation
(2) heteroatom incorporation
(3) multiple bond incorporation

As a previous step, the system determines the isomer characteristics (IC) from the molecular formula. Hydrogen atoms are taken into account here. For that, the system uses the following equation deduced from graph theory:[26]

$$\mathrm{IC} = [\sum n_i(v_i - 2) + 2]/2 \qquad (1)$$

where n_i is the number of atoms of valence v_i present in the molecular formula. In this step the system constructs an IC options menu. The IC consists of the number of double or triple bonds, the number of cycles, or an adequate combination of them. For instance, the IC value for isomers of C_6H_{10} equals 2, and correspondingly the following characteristics are deduced by the system:

(a) 2 double bonds
(b) 2 cycles
(c) 1 double bond and 1 cycle
(d) 1 triple bond

Users can choose one or all of these options. In that way, the generation process can be selective. In addition, the molecular formula can contain one or more types of heteroatoms having different valences in the same structure as for example with thiosulfonic acid in which one S atom has a valence of 2 and the other a valence of 6.

Step 1 also contains a filtering process, necessary for elimination of redundant structures, and a pattern matching process for making a more selective isomer generation.

Steps 2 and 3 are done after the cyclic skeleton is established in step 1, if the molecular formula contains heteroatoms and if the user has selected multiple bonds as one of the IC options. At the end of each of these steps, filtering and pattern matching processes are done sequentially. That is necessary to ensure a general exhaustive and irredundant isomer generation process. Incorporation of heteroatoms and multiple bonds is done, taking care of the valences of the different atoms. These values are defined before starting to run the program. An installed module allows for atom valence assignment.

Use of the program is facilitated with the implementation of specially designed menus. An interactive initialization step allows for the input of the complete series of parameters needed for running the program such as:

molecular formula (this is the only compulsory input data)
IC option
patterns
level or root degree

The system has two possibilities: storing the generated isomers into an archive or just counting the number of generated isomers without storing their structures. In every case an archive is created to take account of the examples already studied. When the process is finished, results are automatically archived at the end of a particular defined file which contains for each case of the register name, the molecular formula, the used isomer characteristics, patterns, levels, or root degrees, and the number of generated isomers.

(1) Cyclic Skeleton Generation. Skeleton generation algorithms use the *N*-tuple notation[26] extended in this work to cyclic isomers. For that, trees of a size *M* are generated first, using

$$M = n + 2c \qquad (2)$$

where *n* is the number of atoms in the molecular formula excluding hydrogen atoms, and *c* is the number of rings determined as IC from the molecular formula. The number *c* is related to the number of leaf-nodes. As it was explained under cyclic isomer tuple notation, these nodes represent terminal points of the tree, i.e., components that have no sons. The number of leaf-nodes in these trees is 2*c*. These nodes do not describe significant atoms of the graph. To be consistent with the employed notation, however, a special program assigns to them the same characteristics (heteroatoms or multiple bonds) of the corresponding node that was used to describe the atoms.

Structures generated in this way are then transformed by an algorithm that allows for ring construction. For instance, the tree (3,2,1,1,0,1,0,1,0,0) can represent or produce the following graphs:

(1:3,2:2,3:1,4:1,1:0,5:1,6:0,6:1,5:0,4:0)

(1:3,2:2,3:1,4:1,6:0,5:1,1:0,6:1,4:0,5:0)

Here, components are separated by a comma. The first number of each component of a tuple defines the node and allows for cycle definition. The second one describes the tree (gives the number of sons each node has). These numbers correspond to parts i and iii described above under tuple notation. (See Figure 2a and b.)

The next step is to determine the representative tuple according to the maximum reverse lexicographic order. In Figure 4 some of the trees that can represent the graph of Figure 2b are shown. They all have four leaf-nodes, and that means the considered structure has two cycles. Nevertheless,

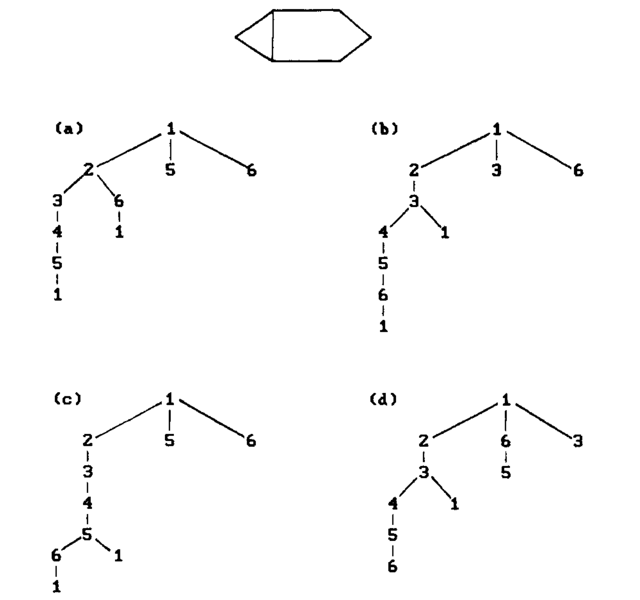

Figure 4. Some of the trees that represent a same graph.

the trees of Figure 4 are equivalent, and the filtering processing of redundant structures is necessary.

Filtering processes for cyclic isomers have to find the tuple with the maximum lexicographic value. In this case, rotations around imaginary axes[26] are not adequate, and a different method was developed here. We call it the **fragmentation method**. In this method, the tree is first decomposed into each of its principal nodes (corresponding graph nodes), keeping connectivity with their neighbors. Then, the tree is recomposed trying to get an N-tuple which is lexicographically bigger. If such a bigger tuple is achieved, then the tree being studied is discarded; otherwise the process continues. For instance, the N-tuple of tree b of Figure 4

(1:3,2:1,3:2,4:1,5:1,6:1,1:0,1:0,3:0,6:0)

becomes the following N-tuple by applying this procedure (see Figure 5):

(1:3,3:2,4:1,5:1,6:1,1:0,2:1,1:0,2:0,6:0)

which is indeed lexicographically bigger than the initial one. That is deduced of course by comparing the sequence of the second number of each tuple component: there was a change from (3121110000) to (3211101000). So, that requires that the initial N-tuple of tree b of Figure 4 be discarded. The fragmentation method can be observed graphically in Figure 5 where tree b of Figure 4, its fragments, and the new tree are shown.

There is another point to be considered. The tree (1:3,2:2,3:2,4:2,5:1,6:1,2:0,1:0,1:0,6:0,3:0,4:0) and the tree (1:3,2:2,3:2,4:2,5:1,6:1,1:0,2:0,1:0,4:0,3:0,6:0) produce the same cyclic graph, as shown in Figure 6. The program has to discard one of them. This is done as follows:

(i) The program starts analyzing tree a and then decomposes it in their six components.

(ii) Then using these fragments and trying to get a larger tuple, the program build again a new tree. Points i and ii are part of the fragmentation process.

(iii) Changes are made here to get the standard increasing numeration of the components of the tree. In this way, the root which was numbered 2 is renumbered 1 and the same is done consistently in the rest of the tree structure for these numbers.

(iv) In this case the procedure followed affords a modified tree (a′), which is identical with tree b. At this point the program that has validated both trees discards the one whose first leaf-node has the smaller number, and so tree b is finally discarded.

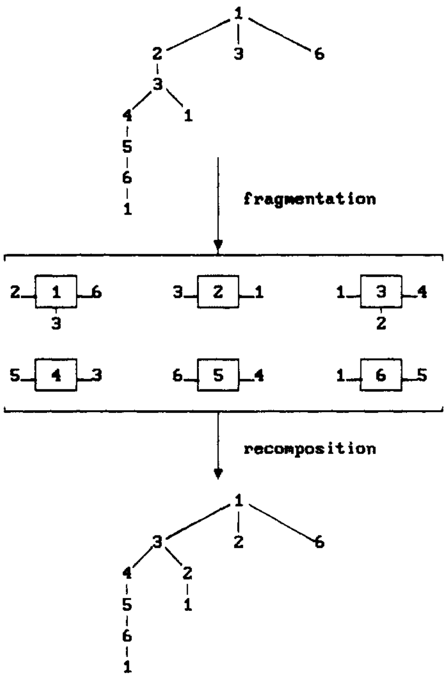

Figure 5. Scheme of the fragmentation method. A tree, its fragments, and the resulting tree whose N-tuple is lexicographically bigger (see text).

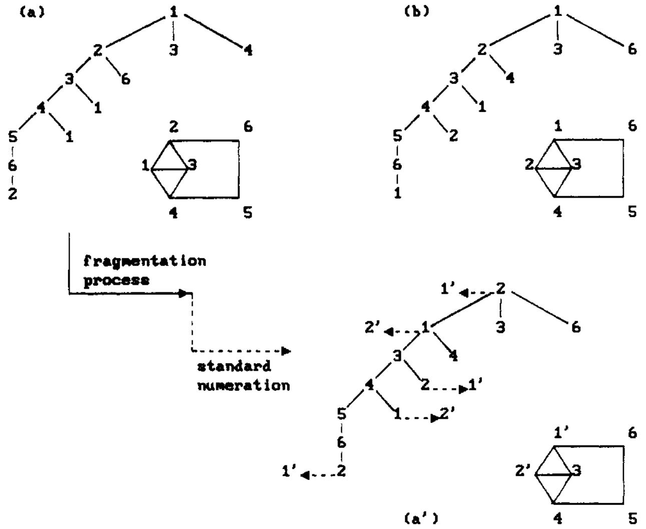

Figure 6. Scheme of the discarding procedure applied to validate trees that produce the same cyclic graph. a and b are validated trees. a′ is an intermediate tree. After the process, b is discarded.

(2) Heteroatoms Incorporation. When skeletons of the isomers are built, i.e., after filtering processes and pattern matching are done, heteroatoms are incorporated to the structure, taking care of the defined valences each atom has and according to the molecular structure composition. Heteroatoms could have different valences in the molecule.

Many times, as a result of the discarding of invalid structures through a fragmentation process (disconnecting a tree and reconnecting its fragments until the canonical tuple is found), similar trees remain. Tuples of these trees differ only in the order of appearance of the atoms. So, a precedence rule was established for carrying on the filtering process. Carbon has the highest priority, followed by elements in groups 5 and 3 of the periodic table, in this order. In each case, the priority is higher for smaller atomic numbers. Next are atoms in

Table I. Compact Molecular Code for Generated Isomers of C_5H_7Cl Having Two Cycles

isomer[a]	compact molecular code								
1	1c4r	2c1s	3c1s	1c0s	4c1s	5cx1s	1c0s	3c0s	5cx0s
2	1c3r	2cx2s	3c1s	4c1s	1c0s	5c1s	1c0s	4c0s	5c0s
3	1c3r	2c2s	3c1s	4cx1s	1c0s	5c1s	1c0s	4cx0s	5c0s
4	1c3r	2c2s	3c1s	4c1s	1c0s	5cx1s	1c0s	4c0s	5cx0s
5	1c3r	2c1s	3cx2s	4c1s	1c0s	5c1s	1c0s	4c0s	5c0s
6	1c3r	2c1s	3c2s	4c1s	1c0s	5cx1s	1c0s	4c0s	5cx0s

[a] See Figure 8 for its corresponding graphic representation.

groups 6 and 2 with the same decreasing priority with increasing atomic numbers. Last are atoms in groups 7 and 1 of the periodic table in this order, in each case ordered according to decreasing priority with increasing atomic number. The following rule was used for the particular examples presented here:

$$C > N > P > O > S > F > Cl > Br > I$$

This rule was applied to discard lexicographically smaller tuples. However, global results of the entire generation process are not affected by the chosen precedence rule. Use of another precedence rule will just provoke a different order of appearance of the isomers in the final coded archive.

Finally, a selective generation can be done through the use of a pattern (or a series of different patterns) which is input by the user at the beginning of the generation process. In this way the number of final-generated structures is decreased. This filtering is done after each one of the three main steps is completed.

(3) Multiple Bond Incorporation. This step is accomplished by the program after the cyclic skeleton is established and after the heteroatoms (if any) are incorporated into the structure. For that, the program takes in consideration the IC option selected by the user at the beginning of the process. Also, valences of each atom are considered for the incorporation of multiple bonds. One or many double or triple bonds can be used. The program is able to generate all the topological isomers corresponding to the used molecular formula without repetitions. For that, once multiple bonds are incorporated, the filtration processes operate accordingly. Here similar trees can result. Tuples of these trees differ only in the order in which atoms involving multiple bonds are written. A solution to this problem is achieved by using the following precedence rule:

$$r > s > d > t$$

where r denotes the root vertex, and s, d, and t represent single, double, and triple bonds, respectively. The use of this rule allows establishment of the canonical tuple. It is convenient to have in mind that for other defined precedence rules, e.g., 'r > t > d > s', the generation process would give the same number of isomers and also the same structures. The only difference would be the order of appearance of the isomers in the final archive.

As it was mentioned above, the generation process could be done under restriction of a designed pattern or a series of patterns that are provided to the program at the beginning. The patterns may or may not contain heteroatoms and multiple bonds. These restrictions are applied now over the canonical tuple, and the final results are found.

In Figure 7 an schematic view of the generation process including filtering steps is presented.

(4) Decodification Module. Generated structures are stored in an archive. Every topological isomer occupies only one line of the archive, thanks to the compact and unique code developed here. In Table I an archive is presented for isomers of C_5H_7Cl having two cycles. The codified archive occupies in this case 0.286 kB, meanwhile the corresponding decodified one occupies, into a Prolog format, 6.780 kB. In Figure 8 the corresponding isomers are graphically represented.

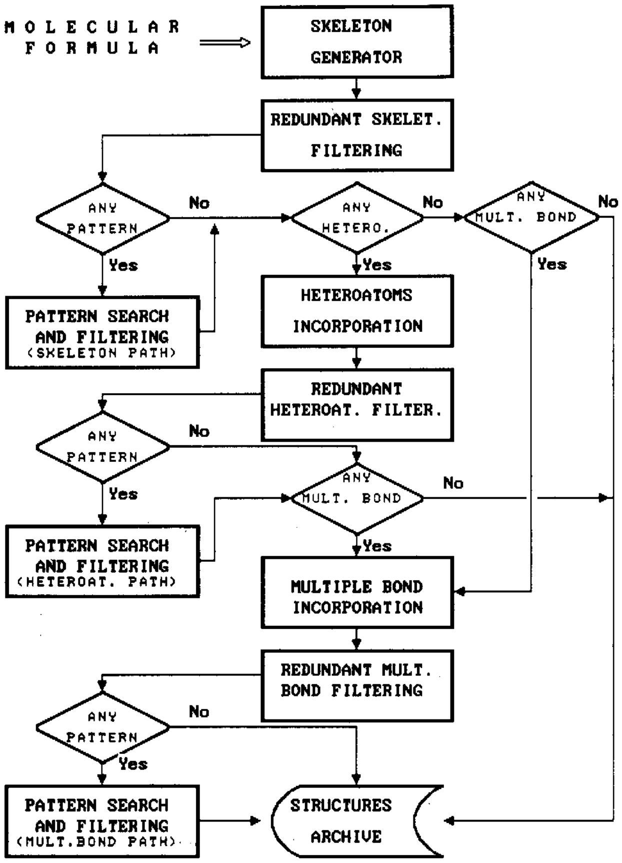
MOLECULAR FORMULA
SKELETON GENERATOR
REDUNDANT SKELET. FILTERING
ANY PATTERN
No
Yes
PATTERN SEARCH AND FILTERING (SKELETON PATH)
ANY HETERO.
No
Yes
ANY MULT. BOND
No
Yes
HETEROATOMS INCORPORATION
REDUNDANT HETEROAT. FILTER.
ANY PATTERN
No
Yes
PATTERN SEARCH AND FILTERING (HETEROAT. PATH)
ANY MULT. BOND
No
Yes
MULTIPLE BOND INCORPORATION
REDUNDANT MULT. BOND FILTERING
ANY PATTERN
No
Yes
PATTERN SEARCH AND FILTERING (MULT.BOND PATH)
STRUCTURES ARCHIVE

Figure 7. General block diagram for the selective generation of cyclic isomers including heteroatoms and multiple bonds.

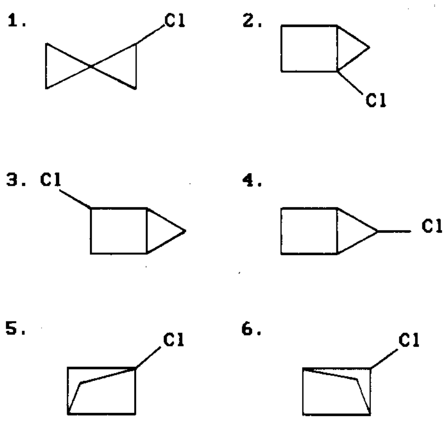

Figure 8. Structures of the isomers whose corresponding compact molecular codes are given in Table I.

In fact, for each line of the N-tuple archive, a decodified archive, for graphic display, is created corresponding to each of the isomers.

Table II. Saturated Cyclic Isomers of Different Families Having 5, 7, and 10 C Atoms[a]

formula	no. of C atoms			no. of cycles
	5	7	10	
C_nH_{2n+2}	3	9	75	0
C_nH_{2n}	1	1	1	1
C_nH_{2n-2}	3	8	21	2
C_nH_{2n-4}	3	27	248	3
$C_nH_{2n+1}X$	8	39	507	0
$C_nH_{2n-1}X$	1	1	1	1
$C_nH_{2n-3}X$	6	28	113	2
$C_nH_{2n-5}X$	17	112	1724	3
$C_nH_{2n+2}O$	14	72	989	0
$C_nH_{2n}O$	1	1	1	1
$C_nH_{2n-2}O$	10	36	131	2
$C_nH_{2n-4}O$	17	202	2424	3
$C_nH_{2n+3}N$	17	89	1238	0
$C_nH_{2n+1}N$	1	1	1	1
$C_nH_{2n-1}N$	14	48	163	2
$C_nH_{2n-3}N$	33	321	3442	3
$C_nH_{2n+2}O_2$	69	463	8697	0
$C_nH_{2n}O_2$	3	4	6	1
$C_nH_{2n-2}O_2$	41	164	722	2
$C_nH_{2n-4}O_2$	94	1121	15865	3
$C_nH_{2n+4}N_2$	97	686	13550	0
$C_nH_{2n+2}N_2$	3	4	6	1
$C_nH_{2n}N_2$	73	264	1046	2
$C_nH_{2n-2}N_2$	271	2593	30272	3

[a] Effect of the number of cycles on the isomer distribution. X represents a substituent.

Table III. Monocyclic Isomers of Different Families Having 5, 7, and 10 C atoms[a]

formula	no. of C atoms			no. of double bonds
	5	7	10	
C_nH_{2n}	1	1	1	0
C_nH_{2n-2}	1	1	1	1
C_nH_{2n-4}	2	3	5	2
C_nH_{2n-6}	2	4	8	3
$C_nH_{2n-1}X$	1	1	1	0
$C_nH_{2n-3}X$	3	4	5	1
$C_nH_{2n-5}X$	5	11	24	2
$C_nH_{2n-7}X$	4	16	56	3
$C_nH_{2n}O$	1	1	1	0
$C_nH_{2n-2}O$	2	3	5	1
$C_nH_{2n-4}O$	4	9	20	2
$C_nH_{2n-6}O$	2	10	44	3
$C_nH_{2n+1}N$	1	1	1	0
$C_nH_{2n-1}N$	3	4	6	1
$C_nH_{2n-3}N$	8	15	29	2
$C_nH_{2n-5}N$	8	25	80	3
$C_nH_{2n}O_2$	3	4	6	0
$C_nH_{2n-2}O_2$	6	12	25	1
$C_nH_{2n-4}O_2$	8	27	94	2
$C_nH_{2n-6}O_2$	4	28	176	3
$C_nH_{2n+2}N_2$	3	4	6	0
$C_nH_{2n}N_2$	12	20	36	1
$C_nH_{2n-2}N_2$	33	76	195	2
$C_nH_{2n-4}N_2$	42	148	565	3

[a] Effect of the insaturation on the isomer distribution.

Input to this module is composed of the name of the *N*-tuple archive, the number of the line from where the decodification is wanted, and the number of the line where to stop decodification. That allows for taking in consideration a certain number of structures from the whole archive. This has proven to be a very practical facility that allows for a better memory use. If decodification is not selectively required, the system also can decodify all of the generated structures.

RESULTS AND DISCUSSION

The system CAMGEC for cyclic structures developed and presented above is based on a new concept of molecular code. Generation of isomers is done in an exhaustive and irredundant way according to any molecular formula and with many selection possibilities. The program also finds out all isomers in agreement with the molecular formula under certain imposed restrictions such as the number of rings and multiple bonds. Imposed restrictions may also include a linear or cyclic substructure as a pattern. In this way even the size of rings of generated isomers can be controlled.

In Table II the number of saturated topological (nonbranched) cyclic isomers of different compound families with 5, 7, and 10 C atoms having one to three cycles in their structure are presented. For comparison, the number of corresponding acyclic isomers is also given. Depending on the formula, these isomers could include alcohols, ethers, amines, peroxides, diamines, etc.

Isomer families considered in Table II and the following are hydrocarbons, monosubstituted hydrocarbons, and compounds with one or two oxygen or nitrogen atoms. As can be seen, only one monocyclic isomer is generated for molecules with zero or one heteroatom independent of the number of C atoms in the molecule. In this way generation of monocyclic isomers can be considered as having the size of the ring as an additional restriction. In Table II the size of the rings for the monocyclic isomers generated for the first two studied families is equal to the number of C atoms: 5, 7, and 10 members. For the following two families having one O or one N atom, the monocyclic ring sizes are 6, 8, and 11 members, respectively (between these isomers are counted some hydrogenated derivatives of pyridine, azocine, and others); meanwhile for the compounds having two O or two N atoms in the molecule, the corresponding monocyclic ring sizes are 7, 9, and 12 members (perhydrodiazepines, perhydrodiazonines, and others). In these cases more than one monocyclic compound is generated to account for the different relative positions of heteroatoms, such as in the 1,2-, 1,3-, and 1,4-perhydrodiazepines.

Symbol X in Table II represents a substituent (Cl, OH, NH_2, AcO, NO_2, CH_3, etc.). This situation is considered in the molecular formula by incorporating a 'special' C atom, Cx, with a residual free valence of 3, i.e., for chlorocyclopentane or for nitrocyclopentane, the input molecular formula is given as C_4H_9Cx. The Cx atoms could also represent a carbocation center.

Data in Table II show that the number of isomers increases both with the number of C atoms and with the number of cycles in the molecule. Also that number increases with the number of heteroatoms and with their valence. In this way for the saturated isomers containing three cycles in their structure, there are 248 isomers of $C_{10}H_{16}$, 2424 isomers of $C_{10}H_{16}O$, and 3442 isomers of $C_{10}H_{17}N$. On the other hand, there are 15865 isomers of $C_{10}H_{16}O_2$ and 30272 isomers of $C_{10}H_{18}N_2$.

One interesting thing for the organic families analyzed and for the relatively small *n* values considered is that the number of acyclic isomers in most of these cases is larger than the number of isomers having two rings, but it is smaller than the one corresponding to isomers having three rings in their structure.

The presence of double bonds instead of rings in the structure produces a sharp decrease in the total number of isomers. This is observed by comparing data from Tables II and III where the number of monocyclic compounds of the families already mentioned in Table II having one, two, and three

Table IV. Selective Generation of Isomers of Different Families Having Three Cycles and 10 C Atoms[a]

	no. of cyclic isomers					
formula	without pattern	pattern *a*	pattern *b*	pattern *c*	patterns *a* and *b*	pattern *d*
$C_{10}H_{16}$	248	148	161	96	82	11
$C_{10}H_{16}O$	2424	1226	1198	734	449	43
$C_{10}H_{17}N$	3442	1620	1540	889	533	49
$C_{10}H_{16}O_2$	15865	6525	5882	3597	1589	133
$C_{10}H_{18}N_2$	30272	10739	9141	5017	2085	161

[a] Effect of the pattern requirements on the isomer distribution. Patterns: *a* corresponds to cyclopropane; *b*, to cyclobutane; *c*, to [1,1,0]-bicyclobutane; and *d*, to [2,2,0]-bicyclohexane.

double bonds are presented. For instance, there are only 195 isomers of $C_{10}H_{18}N_2$ having one cycle and two double bonds instead of the 30272 $C_{10}H_{18}N_2$ isomers having three rings. The same is valid for hydrocarbons: there are five isomers of $C_{10}H_{16}$ having one ring and two double bonds instead of the 248 isomers of $C_{10}H_{16}$ having three rings.

The same tendency observed for saturated cyclic compounds is found for unsaturated isomers. So, there is an increase in the number of isomers with both the number of C atoms and the number of heteroatoms and their valence. In addition, there is an increase in the number of monocyclic isomers with the number of double bonds; for instance, there are five isomers of $C_{10}H_{18}O$ having one double bond, there are 20 isomers of $C_{10}H_{16}O$ having two double bonds, and there are 44 isomers of $C_{10}H_{14}O$ having three double bonds.

Selective generation developed in this work is of great potential utility as can be deduced from Table IV. Here, the number of saturated isomers having 10 C atoms and three cycles in their structure is presented. Generation has been done according to one or two patterns. In all the cases, the use of a pattern allows for the generation of a smaller number of isomers in comparison to the number of isomers generated without such a restriction. For instance, as shown in the last row of Table IV, from the total of 30272 isomers of $C_{10}H_{18}N_2$, only 161 have the pattern *d* present in their structure.

It is interesting to note that in general the more atoms in the pattern, the more selective is the process. For instance, the second row of Table IV shows that there are 1226 isomers of $C_{10}H_{16}O$ with a cyclopropane ring (pattern *a*) in their structure; 1198 isomers having as a part of their structure a cyclobutane ring (pattern *b*); there are only 43 isomers that have as a substructure two condensed cyclobutane rings (pattern *d*).

Condensed cycles are patterns more restrictive than isolated rings. That can be deduced by comparing for any of the examples in Table IV, the number of isomers that have pattern *b*, four atoms in a cyclobutane ring, and the number of corresponding isomers that have pattern *c*, four atoms in two condensed cyclopropane rings. For isomers of $C_{10}H_{16}$, these numbers are respectively 161 and 96. On the other hand, pattern *d*, two condensed cyclobutane rings having six atoms as a total, reveals to be more restrictive than using a sequence of two pattern summing up seven atoms as in the case of patterns *a* and *b*. That can be illustrated for instance, by the isomers of $C_{10}H_{16}O_2$ whose corresponding numbers are respectively 133 and 1589.

The use of more than one pattern, such as in the case of pattern *a* and one pattern *b*, makes the generation process more selective in comparison with the process done with any of the patterns *a* or *b* alone. From Table IV, there are 1620 isomers of $C_{10}H_{17}N$ having pattern *a*, 1540 isomers having pattern *b*, and only 533 isomers having both patterns *a* and *b*.

In Table V some representative results are illustrated obtained with CAMGEC. For each run, the IC used are specified (number of cycles and double bonds present in the isomer structure) as is the use of a pattern or a sequence of patterns. Five different generation processes have been accomplished for each run: one for each root degree (2, 3, 4, and 5) and one for all of them together. For each process, the number of cyclic isomers, the CPU time per isomer, and the total CPU time are given.

Data from Table V confirm tendencies already analyzed and illustrate new features CAMGEC can achieve. The new features refer to the capacity of working with heteroatoms having multiple valences in the same molecule and to the ability of generating isomers whose corresponding root degree values are kept fixed for any process. The number of isomers generated in this way for the different root degree values and for a particular formula will sum up the total number of isomers. This option is very useful in the case of large generation processes such as those with a large number of heavy atoms in the molecular formula or those with complex patterns. In these cases, partial processes can be run separately for each root degree value. The global CPU time for each run is of the same order of magnitude as that of the sum of the individual CPU time for each root vertex degree process, but the chronological connection time is significantly reduced by working with separated partial processes.

In addition, examples are shown in Table V concerning the use of the different possibilities in CAMGEC for making the isomer generation process selective. In this way, in a single process like the one applied in runs 1–3, the following elements of selection and control are used: (a) molecular formula, (b) isomer characteristics, (c) use of one or two patterns, (d) root degree, and (e) identity of the process—it is not possible to run a process having a duplicate archive name even if it is a repetition of one already done. This point helps to avoid fortuitous errors.

Isomers in runs 5 and 7 in Table V have in their molecular formula a P atom with a valence of 5. This single fact can dramatically increase the number of generated isomers from 30272 isomers of $C_{10}H_{18}P_2$ (run 4) to 645269 isomers of $C_{10}H_{18}P_2$, having one P atom with a valence of 5 (run 5). Both of these sets of isomers have three cycles in their structures, but the first set (run 4) has no double bond while the second set (run 5) has one double bond. This fact could partially explain the bigger number of cyclic isomers found in run 5. The bigger valence of the P atom could be a more significant explanation for the observed values. A similar example is shown in runs 6 and 7 where an increase from 10144 to 101259 isomers is observed. Here the related numbers are smaller than the ones from runs 4 and 5 because of the smaller number of cycles: 2 vs 3.

On the other hand, use of unsaturated cyclic patterns with multiple heteroatoms is illustrated in runs 2 and 3 of Table

Table V. Global Performance and Selectivity of Generation System for Cyclic Isomers of 10 C atoms, in Function of Number of Cycles and Double Bonds, Presence of Patterns, Presence of One P Atom with a Valence of 3 and an Other with a Valence of 5 in Same Molecule, and Root Degree Value

run	formula	no. of cycles/ double bonds	pattern[b]	data[b]	root degree 2	root degree 3	root degree 4	root degree 5	global
1	$C_{10}H_{18}N_2$	3/0	a and b	*I*	0	931	1154		2085
				t		7.573	4.298		8.005
				T	4722.51	7051.00	4960.57		16690.52
2	$C_{10}H_{16}N_2$	2/2	e	*I*	0	148	6		154
				t		4.025	62.806		8.855
				T	398.28	595.70	376.84		1363.72
3	$C_{10}H_{16}N_2$	2/2	f	*I*	0	174	6		180
				t		3.405	62.491		7.570
				T	397.62	592.63	374.95		1362.67
4	$C_{10}H_{18}P_2$	3/0		*I*	0	16156	14116		30272
				t		0.462	0.370		0.582
				T	4709.67	7468.76	5223.08		17619.12
5	$C_{10}H_{18}P_2$[a]	3/1		*I*	0	371563	270628	3078	645269
				t		0.079	0.078	1.436	0.097
				T	8274.16	29715.84	21258.24	4420.48	63080.96
6	$C_{10}H_{18}P_2$	2/1		*I*	0	9394	750	0	10144
				t		0.090	0.526		0.161
				T	400.46	848.15	394.78	281.75	1641.11
7	$C_{10}H_{18}P_2$[a]	2/2		*I*	0	94781	6478	0	101259
				t		0.037	0.106		0.050
				T	625.14	3552.80	692.31	282.08	5135.36

[a] One P atom with a valence of 5. [b] Patterns: a and b correspond to cyclopropane and cyclobutane; e corresponds to dihydroimidazole; and f corresponds to dihydropyrazole. [c] *I* is the number of cyclic isomers; *t* is the CPU time per isomer (s/i); and *T*is the total CPU time (s).

V for isomers of $C_{10}H_{16}N_2$ having two cycles and two double bonds. Pattern *e*, a dihydroimidazole ring, is more selective than pattern *f*, a dihydropyrazole ring, probably for reasons of symmetry. Here it is worth noting that the isomers generated under root degree 4 represent the isomers that have a spiro system (they have a cyclic atom joined to four other non-hydrogen atoms). That was easily proved both by displaying the generated structures at a graphic interface and by manually decoding the generated archive.

The CPU time was determined in a Data General-Aviion instrument under DG/UX operating system, working in a multiuser mode. The program was written in C and is running also in a MicroVax II under Ultrix.

Finally, the IC change of a cycle for a double bond both decreases the number of generated structures as it was explained before and also produces a more efficient global process as it is observed by comparing runs 4 and 6 in Table V with global CPU time of 0.582 s per isomer (s/i) and 0.161 s/i, respectively. For runs 5 and 7 the global CPU time per isomer corresponds to 0.097 and 0.050 s/i, respectively.

Data from runs 5 and 7 show a larger number of generated isomers (64 5269 and 10 1259, respectively) and the more efficient processes both at a global level and also at the level of each root degree process.

The generated isomers not only are enumerated as was envisaged at the beginning of the problem of enumeration of isomers,[37] but they are kept in an archive ready to be displayed or input to programs that calculate molecular parameters such as molecular volume, topological indexes,[26,38-40] and others (AM1, MM2, Gaussian methods).

In relation to the number of isomers generated for benzenoids, it was not possible to compare directly our results with others previously reported[16,18,19] because of different software orientation. Our method generates all the isomers that have a particular molecular formula, and the other systems generate all the isomers that have a certain number of cycles. For example,[18,20] between the 22 benzenoid species containing a total of five benzene rings, there are 12 isomers with 22 C atoms in their structure, six isomers with 21 C atoms, three isomers with 20 C atoms, and one with 19 C atoms.

One important thing to be noted is the concept of compact molecular code introduced here as it was explained above under tuple notation. The code is simple, unique, brief, pronounceable, and easily comprehensible, and it does not depend on chemical intuition or properties of chemicals as is required for other molecular codes.[17,41]

Our molecular code does not change the molecule, but it codifies the cyclic structure as it is. This is one of the principal differences with methods that turn the cyclic molecule into an acyclic one and then codify the acyclic spanning subtree resulted from this transformation.[17] Besides this difficulty, they have to number this fragment and add to the code the corresponding number of positions that represent ring closures. As is obvious, their code cannot be directly visualized because it represents an acyclic structure. Neither can the number of cycles be known by inspecting the code. The only direct information it gives is the number of carbon atoms of the molecule and the number of excized vertices.

In addition, that coding method has more opportunity for errors which can pass undetected, even by specialized users. For example, the molecular code proposed[17] for triphenylene will incorrectly produce by decoding a cycle with seven C atoms instead of one of six. This error comes from incorrectly erasing a vertex in one of the first steps of codification to produce a wrongly labeled spanning subtree and, therefore, an incorrect code. Our method avoids such mistakes because it codifies the cyclic structure directly and in addition automatically validates codification of molecules or patterns. Also notation of the label part of the code[17] is troublesome for writing or typing. If vertex valencies and position numbers are mixed, then errors like that of run 17 of Table III reported for a benzenoid of 30 C atoms and seven cycles can be produced. The sequence '2,2,12,5,3' appears. It is evident that there is a mistake because the first number 2 indicates that a vertex of valence 2 has to be joined to both positions signed by the following two numbers: 2 and 12. The following number, 5, should be the valence of the next vertex excised whose valence is not correct for C atoms. Only after decoding the *N*-tuple part of the code, then labeling the nodes, and drawing down the rest of the structure is it possible to deduce the probably correct sequence '2,21,25,3'.

Another point that should be mentioned refers to molecular code length, which should be the same for structures of the same size and similar complexity.[17] In fact, the molecular code proposed in this work consists in an N-tuple with a fixed number of components $M = n + 2c$ as was established above in eq 2, and it is the base of the generation method developed here. For instance, for triphenylene, a branched cata-fused benzenoid, which is a molecule with 18 C atoms and four cycles, our compact code has 26 ciphers coinciding with the reported one.[17] However, for chrysene, a nonlinear nonbranched cata-fused benzenoid, also with 18 C atoms and four cycles, and with a compact code of 26 ciphers length, the reported value[17] is 24. This behavior is found also for other examples. The difference is produced because the reported molecular code[17] depends on the number and type of prunned vertex.

Summing up, the molecular code proposed here is more general and easier for visualizing and decoding than those in the literature.[17,18] In fact, just by looking at the N-tuple and counting up the leaf-nodes appearing as '0', is it possible to infer the number of cycles the molecule has and also the number and type of both heavy atoms and multiple bonds.

In conclusion, a computational method of exhaustive and irredundant generation and enumeration of cyclic isomers has been presented. The method has been written in C (Unix), and it is based on graph theory. Several complementary tools have been developed. Special trees have been defined for cyclic structures, and a new compact molecular code for cyclic isomers has been proposed. Decoding subroutines have been developed, and also others for validation of the different interactive phases of the system have been implemented.

The generation method described here has the possibility of working in a selective way. A pattern or a sequence of them could be used for one process. Also the system considers the presence of multiple bonds and of heteroatoms which can have different valencies in the same molecule. The generated isomers can be manually decoded or can be directly displayed with available graphic capacities.

A variety of examples of the use of this program have been presented. From them it is possible to realize that the system could solve problems of molecular design, organic synthesis, and structure elucidation. Also, the designed new concept of molecular code will be of great help in chemical documentation for storing and retrieval of molecular information.

ACKNOWLEDGMENT

We gratefully acknowledge the University of Santiago through their Direction of Research and Development, for financial support. Also we are very grateful to Prof. A. T. Balaban for many useful criticisms and suggestions to this work.

REFERENCES AND NOTES

(1) Downs, G. M.; Gillet, V. J.; Holliday, J. D.; Lynch, M. F. Review of Ring Perception Algorithms for Chemical Graphs. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 172–187.

(2) Corey, E. J.; Peterson, G. A. An Algorithm for Machine Perception of Synthetically Significant Rings in Complex Cyclic Organic Structures. *J. Am. Chem. Soc.* **1972**, *94*, 460–465.

(3) Wipke, W. T.; Dyott, T. Use of Ring Assemblies in a Ring Perception Algorithm. *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 140–144.

(4) Zamora, A. An Algorithm for Finding the Smallest Set of Smallest Rings. *J. Chem. Inf. Comput. Sci.* **1976**, *16*, 40–43.

(5) Gasteiger, J.; Jochum, C. An Algorithm for the Perception of Synthetically Important Rings. *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 43–48.

(6) Roos-Kozel, B. L.; Jorgensen, W. L. Computer-Assisted Mechanistic Evaluation of Organic Reactions. 2. Perception of Rings, Aromaticity, and Tautomers. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 101–111.

(7) Harary, F. *Graph Theory*; Addison-Wesley: Reading, MA, 1972.

(8) Lunnon, W. F. *Graph Theory and Computing*; Read, R. C., Ed.; Academic: New York, 1972.

(9) Masinter, L. M.; Sridharan, J.; Lederberg, J.; Smith, D. H. Application of Artificial Intelligence for Chemical Inference. XII. Exhaustive Generation of Cyclic and Acyclic Isomers. *J. Am. Chem. Soc.* **1974**, *96*, 7702–7714.

(10) Balaban, A. T.; Filip, P.; Balaban, T. S. Computer Program for Finding all Possible Cycles in Graphs. *J. Comput. Chem.* **1985**, *6*, 316–329.

(11) Fujita, S. Description of Organic Reactions Based on Imaginary Transition Structures. 1. Introduction of New Concepts. *J. Chem. Inf. Comput. Sci.* **1986**, *26*, 205–212.

(12) Fujita, S. Description of Organic Reactions Based on Imaginary Transition Structures. 2. Classification of One-String Reactions Having an Even-Membered Cyclic Reaction Graph. *J. Chem. Inf. Comput. Sci.* **1986**, *26*, 212–223.

(13) Downs, G. M.; Gillet, V. J.; Holliday, J. D.; Lynch, M. F. Theoretical Aspects of Ring Perception and Development of the Extended Set of Smallest Rings Concept. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 187–206.

(14) Akutsu, T. A New Method of Computer Representation of Stereochemistry Transforming a Stereochemical Structure into a Graph. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 414–421.

(15) Rucker, G.; Rucker, C. Isocodal and Isospectral Points, Edges, and Pairs in Graphs and How To Cope with Them in Computerized Symmetry Recognition. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 422–427.

(16) Knop, J. V.; Muller, W. R.; Jericević, Z.; Trinajstić, N. Computer Enumeration and Generation of Trees and Rooted Trees. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 91–99.

(17) Randić, M.; Nikolić, S.; Trinajstić, N. Compact Molecular Codes for Polycyclic Systems. *J. Mol. Struct. (Theochem)* **1988**, *165*, 213–228.

(18) Knop, J. V.; Szymanski, K.; Jericević, Z.; Trinajstić, N. Computer Enumeration and Generation of Benzenoid Hydrocarbons and Identification of Bay Regions. *J. Comput. Chem.* **1983**, *4*, 23–32.

(19) Knop, J. V.; Szymanski, K.; Jericević, Z.; Trinajstić, N. Computer Generation and Identification of Carcinogenic Bay Regions in Benzenoids Hydrocarbons. *Int. J. Quantum Chem.* **1983**, *23*, 713–722.

(20) Knop, J. V.; Muller, W. R.; Jericević, Z.; Trinajstić, N. *Computer Generation of Certain Classes of Molecules*; SKTH: Zagreb, 1985; Chapters 3 and 4.

(21) Hites, R. A.; Simonsick, W. J., Jr. *Calculated Molecular Properties of Polycyclic Aromatic Hydrocarbons*; Elsevier: Amsterdam, 1987.

(22) Dias, J. R. *Handbook of Polycyclic Hydrocarbons*; Elsevier: Amsterdam, 1987.

(23) Gelboin, H. V.; Ts'o, P. O. P. *Polycyclic Hydrocarbons and Cancer*; Academic: New York, 1978–1981; Vols. I–III.

(24) Knop, J. V.; Szymanski, K.; Klasing, L.; Trinajstić, N. Computer Enumeration of Substituted Polyhexes. *Comput. Chem.* **1984**, *8*, 107–115.

(25) Balaban, A. T. Chemical Graphs. Part 12. Configurations of Annulenes. *Tetrahedron* **1971**, *27*, 6115–6131.

(26) Contreras, M. L.; Valdivia, R.; Rozas, R. Exhaustive Generation of Organic Isomers. 1. Acyclic Structures. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 323–330.

(27) Contreras, M. L.; Deliz, M.; Rozas, R. Personal Microcomputer Based System of Chemical Information with Topological Structure Data Elaboration. *J. Chem. Inf. Comput. Sci.* **1987**, *27*, 163–167.

(28) Contreras, M. L.; Allendes, C.; Alvarez, L. T.; Rozas, R. Perception and Recognition of Digitized Molecular Structures. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 302–307.

(29) Ash, J. E.; Chubb, P. A.; Ward, S. E.; Welford, S. M.; Willet, P. *Communication, Storage, and Retrieval of Chemical Information*; Ellis Horwood: Chichester, UK, 1985; Chapter 8.

(30) Funatsu, K.; Miyabaiyashi, N.; Sasaki, S. Further Development of Structure Generation in the Automated Structure Elucidation System CHEMICS. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 18–28.

(31) Bangov, I. P. Computer-Assisted Structure Generation from a Gross Formula. 3. Alleviation of the Combinatorial Problem. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 277–289.

(32) Hendrickson, J. B.; Parks, C. A. Generation and Enumeration of carbon skeletons. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 101–107.

(33) Contreras, M. L.; Valdivia, R.; Rozas, R. Exhaustive Generation of Organic Isomers. 3. Mixed Ring-Chain Structures, in preparation.

(34) Pine, S. H.; Hendrickson, J. B.; Cram, D. J.; Hammond, G. S. *Organic Chemistry*; McGraw-Hill: New York, 1980.

(35) Fessenden, R. J.; Fessenden, J. S. *Organic Chemistry*; Willard Grant: Boston, 1979.

(36) March, J. *Advanced Organic Chemistry*; Wiley: New York, 1985.

(37) Polya, G. *Acta Math.* **1937**, *68*, 145.

(38) Kier, L. B.; Hall, L. H. *Molecular Connectivity in Structure-Activity Analysis*; Research Studies Press: Letchworth, England, 1986.

(39) Rouvray, D. H. The Modeling of Chemical Phenomena Using Topological Indices. *J. Comput. Chem.* **1987**, *8*, 470–480.

(40) Diudea, M. V.; Minailiuc, O.; Balaban, A. T. Molecular Topology. IV. Regressive Vertex Degrees (New Graph Invariants) and Derived Topological Indices. *J. Comput. Chem.* **1991**, *12*, 527–535.

(41) Polya, G.; Read, R. C. *Combinatorial Enumeration of Graphs, Groups, and Chemical Compounds*; Springer-Verlag: New York, 1987.