

## COMPUTER GENERATION OF ACYCLIC GRAPHS BASED ON LOCAL VERTEX INVARIANTS AND TOPOLOGICAL INDICES. DERIVED CANONICAL LABELLING AND CODING OF TREES AND ALKANES\*

Teodor Silviu BALABAN\* and Petru A. FILIP

*Centre of Organic Chemistry of the Roumanian Academy, Splaiul Independentei 202 B,  
71141 Bucharest, Roumania*

Ovidiu IVANCIUC

*Department of Organic Chemistry, Polytechnic Institute, Splaiul Independentei 313,  
77206 Bucharest, Roumania*

### Abstract

A new procedure (GENLOIS) is presented for generating trees or certain classes of trees such as 4-trees (graphs representing alkanes), identity trees, homeomorphical irreducible trees, rooted trees, trees labelled on a certain vertex (primary, secondary, tertiary, etc.). The present method differs from previous procedures by differentiating among the vertices of a given parent graph by means of local vertex invariants (LOVIs). New graphs are efficiently generated by adding points and/or edges only to nonequivalent vertices of the parent graph. Redundant generation of graphs is minimized and checked by means of highly discriminating, recently devised topological indices based either on LOVIs or on the information content of LOVIs. All trees on  $N + 1$  ( $N + 1 < 17$ ) points could thus be generated from the complete set of trees on  $N$  points. A unique cooperative labelling for trees results as a consequence of the generation scheme. This labelling can be translated into a code for which canonical rules were recently stated by A.T. Balaban. This coding appears to be one of the best procedures for encoding, retrieving or ordering the molecular structure of trees (or alkanes).

### 1. Introduction

The problem of enumerating (counting) and generating (displaying) structures (isomers) has been preoccupying mathematicians and chemists for over a century. The large number of papers [1–11], reviews [12, 13] and books [14–18] devoted to this subject emphasizes its importance both for chemistry and for graph theory.

We are aware that what we are about to describe here for acyclic structures has already been done elegantly and efficiently by other groups [4, 7, 10, 11], so that the numerical results presented in the tables are not new. However, our method and the derived algorithm are new and have a generality which makes our approach

\*Dedicated to Professor Alexandru T. Balaban on the occasion of his 60th anniversary.

\*Present address: Structural Chemistry Department, Ruhr University Bochum, 4630 Bochum 1, Germany.

attractive for new applications. The recently described generation of carbon skeletons by Hendrickson and Parks [11], although much broader in scope, encouraged us to present our approach to this problem, which we believe to be more efficient for chemically relevant species with a somewhat limited number of atoms. The theoretical basis developed by Kvasnička and Pospichal [10] for the cooperative indexing of structures could be put to use in our algorithm. Actually in the same paper, the latter authors have addressed the problem of reconstructing graphs from topological indices. We present here an extension to these previous studies.

We started this work some years ago in connection with topological indices (TIs) and the study of their degeneracy, i.e. which are the smallest nonequivalent graphs for which the same value is obtained for a certain TI. For this purpose, we needed a data base of structures on which to run routinely calculations for various TIs.

The procedures described at that time in the literature [4–8] were not adaptable to our computing facilities. We classify the existing procedures for generating structures into *budding* algorithms which grow new structures by appending vertices and/or edges to previously generated structures, and *coding* algorithms which generate computer codes representing structures.

We decided to devise a “budding” program (written in BASIC and which could be run on a Z-8080, 64 KB RAM) adapted to our very narrow scope. The basic idea of this program is the same as that of the SKEL\_GROW program of Hendrickson and Parks [11] or that of Trinajstić’s generating algorithm of polyhexes [9], namely to generate structures on  $N + 1$  vertices (hexes) from the previously generated structures of  $N$  vertices (hexes). The disadvantage pointed out by Hendrickson and Parks for this approach is that it requires large computer times and storage, derived from the need to compare a newly generated structure with all the previously generated ones. These disadvantages can be partially avoided by storing the previously generated structures in the computer memory as a single number representing the TI whose degeneracy needs to be tested. Since the total number of structures on  $N + 1$  vertices of a given type (alkanes, primary alcohols, etc.) is known [14, 19], we generated these from the previously corrected generated structures on  $N$  vertices. If the numbers obtained by our approach matched the known ones, then two conditions must have been simultaneously fulfilled:

- (i) all the TIs for the newly generated structures must have been distinct, i.e. the respective TI is not degenerate for graphs with  $N + 1$  vertices; and
- (ii) at the local, vertex level, for all the structures on  $N$  vertices, the nonequivalent vertices must have been correctly distinguished so that into each nonequivalent position, a new bond could be added in order to link the  $(N + 1)$ th vertex of a newly generated structure.

If the numbers of structures did not match, then the generation scheme must have been at fault, a degeneracy must have been found and this could be due to the non-fulfillment of the first, or second, or both of the above conditions. By retracing the generation scheme, one could easily find the missing isomer and one could

verify which of these conditions was not fulfilled. In a previous paper [20], we termed the degeneracy contradicting the first condition as “accidental” degeneracy, or *operational degeneracy*, i.e. the operations of the “global” TI were not sufficiently complex as to discriminate all the structures on  $N$  vertices. The degeneracy contradicting the second condition was termed *assignment degeneracy*, meaning that to nonequivalent vertices, equal local vertex invariants (LOVIs) were assigned.

The success in discriminating structures of some of our newly devised topological indices (based either on the information on distances [21], or on LOVIs [20]), as well as the ease with which nonequivalent vertices are distinguished by means of the local invariant set (LOIS) of LOVIs, makes from the blending of these two features an efficient graph generator which is described in the present paper. To ensure self-consistency, the LOVIs and TIs on which the generation algorithm is based will be briefly reviewed in the third section of this paper. Our program (which we named GENLOIS) is described in the next section, together with the computing times and the numerical results for acyclic structures.

The general applicability, however, is limited by the fulfillment of the two criteria mentioned above. For chemical relevant species for which all congeners need to be known, or accessed, and where  $N$ , the number of atoms, cannot be too large, this scheme functions well, as we will try to demonstrate.

Another point needs to be stressed and it stems from our more heuristical approach to chemical related graph-theoretical problems. In the generation of 4-trees (alkane hydrogen-depleted graphs), since our procedure starts from methane and produces higher alkanes, these are produced in a given order, which is specific to the generation algorithm, i.e. it is not imposed by any exterior conditions. What is more interesting is that a labelling of vertices results for each isomer. This labelling resembles the Morgan “orbit” numerotation schemes [19], as well as the system of coding trees due to Neville [22]; also, it is cooperative in the sense defined by Kvasnička and Pospichal [10]. Again, this is done without exterior rules. This labelling is easily translated into an  $N - 3$  tuple which may be used for coding alkanes, as A.T. Balaban has recently shown [23]. This code, which we will term LINKCODE, will be discussed in section 5 of this paper.

As a consequence of the cooperative indexing, GENLOIS generates the structures in the order of increasing codes (lexicographic increasing order). The astonishing fact is that these codes also represent the minimum code for a given isomer and are thus canonical. A theorem is given to prove that this statement is valid as long as the TIs and LOVIs are discriminating correctly structures and vertices, respectively.

An alternative procedure to the “budding” scheme that is used in our approach is the “coding” procedure which led to the most efficient program for generating acyclic structures, due to Trinajstić and co-workers [7]. Also, the program SKEL\_GEN of Hendrickson and Parks [11], which generates carbon skeletons, uses a “coding” approach which is actually more efficient than the one used in the SKEL\_GROW program. This procedure consists in a combinatorial generation of codes ( $N$ -tuples [7], upper [11], or lower [10] triangles of the adjacency matrix) and ordering these

codes into a “reverse or direct lexicographic order”. If a produced code does not represent a connected graph or after permutation of the labels, it does not represent a minimum (maximum) code, that structure is rejected as either not relevant or as already in the list of generated structures. Since each code is generated only once (forward generation) and the comparison of a new structure with the previously generated ones is avoided, this scheme is faster and is less memory consuming. Various schemes have also been devised to drastically reduce the permutation of the codes in order to eliminate irrelevant ones. The efficiency of the Trinajstić algorithm resides in the generation of exactly the set of minimum  $N$ -tuples, and this has allowed the generation of structures with up to 80 vertices [7, 17].

Having to hand a generation algorithm which apparently functions well and which is based on the budding sequence, not on the approach of coding the structures, we extract a code from the generated structures. This code, applied to alkanes, was disclosed earlier by A.T. Balaban [23] and it resulted from the computer representation of the generated structures. In addition, the generation algorithm produces structures in a strict order. We do not argue as this being *the* ordering (of alkanes, for instance) but we compare it, in section 6, to other existing ordering procedures.

The ordering of alkanes is somewhat similar to that devised by Trinajstić and co-workers [7] (the inverse lexicographic order) in the respect that the most branched alkane (or the star tree) is generated first, while the linear  $N$ -alkane (tree) is generated last. We will point out the differences between these two ordering procedures, differences which increase as  $N$  increases. The coding of alkanes and their ordering according to the increase of our canonical code is actually independent of the generation algorithm and may be regarded as a hierarchical ordering scheme for all alkanes according to an intrinsic property that is related to their branching.

The generation algorithm is shown to be inferior in speed to the Trinajstić generation algorithm [7], but much faster than the programs SKEL\_GEN and SKEL\_GROW of Hendrickson and Parks [11] used for generating carbon skeletons.

An interesting feature of the GENLOIS algorithm is that it connects three important and much debated problems of graph theory: structure generation, canonical coding, and ordering of structures by means of a unique criterion. The aim of the present paper is to show that for chemical relevant species, all these problems can be efficiently solved with the aid of TIs.

For graph-theoretical terms and their chemical equivalents, the reader is referred to available reviews of chemical applications of graph theory [12–16, 24–28].

## **2. Generation of acyclic chemical structures with the aid of local vertex invariants and topological indices**

The direct enumerations of graphs are methods of counting graphs simultaneously with their generation. In the following description, a direct “budding” enumeration procedure is presented which runs efficiently even on personal computers. We limit

our present discussion to acyclic structures; the results for cyclic structures, which can also be accessed with the present algorithm, will be given separately.

Our approach for generating structures is very similar to the “by hand” or intuitive way of producing new isomers from a given set of inferior homologous isomers. Therefore, it is instructive to explicit (see fig. 1) the generation from the

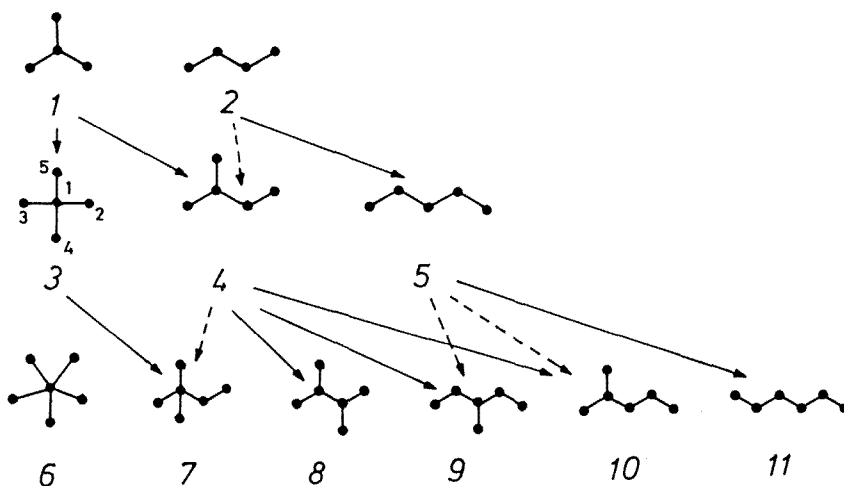


Fig. 1. A “budding” sequence for generating acyclic structures.

two butanes **1** and **2** of the three pentanes **3–5**, and from these of the five hexane isomers **7–11**. This is a special case of the generation of 4-trees in which the degree of vertices, i.e. the valency of a carbon atom, cannot be larger than four. We should mention that addition of a new edge to the vertex of degree four in **3** (quaternary carbon labelled **1**) gives rise to a 5-tree (**6**) and is permitted in a generation algorithm only if general trees are desired.

The set of structures on  $N$  vertices is processed sequentially. Each such graph is denoted as a parent (predecessor) graph (e.g. butane and pentane isomers in fig. 1) and generates a subset of “daughter” (successor) graphs with  $N + 1$  vertices (e.g. pentanes and hexanes, respectively, in fig. 1) by connecting (or budding) a new vertex to each of (or from) the vertices in the parent graph.

The butanes **1** and **2** can be budded to eight structures with five vertices which represent the three pentanes **3–5**. In turn, from the latter, fifteen newly generated structures are obtained, representing the six isomers of hexane. The differences between the existing numbers of isomers and the generated ones is due to the fact that the same daughter graphs can be accessed from different parent graphs or simply, because from the same parent graph, topologically equivalent vertices were budded. Therefore, the successor generation contains redundant structures and their number increases factorially as  $N$  increases.

In order to generate new isomers more efficiently, two rules can be stated for adding a new vertex. Firstly, in order to generate acyclic structures, a new edge should be added only to non-equivalent vertices of the parent graph. Thus, from **3** only two new structures (**6** and **7**) should be generated; vertices labelled 2, 3, 4 and 5 are equivalent and so their budding leads to the same hexane isomer, namely **7**. Taking into account the equivalence classes of vertices in **3–5**, only eight new structures (corresponding to the eight arrows in fig. 1) need now to be generated. Secondly, one observes that hexanes **7**, **9** and **10** can be generated, each from two parent pentanes. This duplicate generation could be avoided, or at least minimized, if only the “forward” generation is allowed. By “forward” generation, we imply a pre-existent order of the three parent pentanes (as in fig. 1) which gives rise to an imposed order of the newly generated hexanes. Thus, only five new structures need now be generated (this “forward” generation is indicated by full arrows in fig. 1), if the “backward” generation (dotted arrows) is prohibited.

We shall first illustrate how, by means of local vertex invariants, one can distinguish the equivalence classes of vertices in a parent graph, and then how to code and order trees and thus avoid as much as possible the “backward” redundant generation of structures. The problem of storing newly generated structures for chemically relevant species can be done in the main memory by means of a highly discriminant topological index  $V$  [21], or by means of TIs based on local vertex invariants [20], which we discuss in section 3.

### 3. Local vertex invariants (LOVIs) and topological indices (TIs)

We have recently presented an approach [20] which permits an easy way to assign to vertices of a given graph local vertex invariants (LOVIs). This approach is similar to the one proposed by Golender et al. [29] for flow graphs. It has the advantage of a very low *assignment degeneracy*, i.e. distinct LOVIs are obtained for nonequivalent vertices and different sets of ordered LOVIs are usually obtained for nonisomorphic graphs. Only some special cases of polycyclic graphs, although nonisomorphic, give rise to the same set of ordered LOVIs.

On the other hand, we have found no example of an acyclic graph with less than seventeen vertices that, for nonequivalent vertices, has the same LOVIs. Thus, these vertex invariants are well suited for our present purpose, namely to discriminate the equivalence classes of vertices in a given parent graph.

For a graph of  $N$  vertices, a local invariant set (LOIS)  $X$  of LOVIs,  $x_i$  ( $i = 1, 2, \dots, N$ ), is obtained as the solution set of a system of linear equations:

$$\mathbf{Q} \cdot \mathbf{X} = \mathbf{R}, \quad (1)$$

where  $\mathbf{Q}$  is a square  $N \times N$  matrix obtained from the adjacency matrix  $\mathbf{A}$  or distance matrix  $\mathbf{D}$  by substituting its diagonal elements  $a_{ii}$ , and  $\mathbf{R}$  is a column vector with  $N$  components. Several types of LOIS were given in the initial paper [20] and others

have been calculated. The type of a LOIS can be designated by a triad **APR**, where **P** is the column vector of components  $p_i$  which replace the diagonal elements  $a_{ii}$  of matrix **A**, thus forming matrix **Q**. System (1) becomes

$$(\mathbf{A}^\alpha + \mathbf{1} \cdot \mathbf{P}) \cdot \mathbf{X} = \mathbf{R}, \quad (2)$$

where  $\alpha$  is a power of the adjacency matrix (or distance matrix) and **1** is the unit matrix.

The column vectors **P** and **R** reflect certain chosen properties which can be *topological*, such as the vertex degrees **V**, the distance sums **S**, the number of vertices in the graph **N** (i.e. a constant) or *chemical*, such as the atomic number **Z**, atomic mass **M** (to distinguish among isotopic species), ionic or atomic radii, ionization potential, electronegativity of the respective atoms, etc.

For our present purpose, we used two such LOIS, namely **AZV** and **D'ZV**, obtained by replacing the diagonal elements of the adjacency matrix and reciprocal distance matrix [30], respectively, with the atomic number **Z**. The free term vector **R** in (1) is represented by **V**, namely the set of vertex degrees (the valency set of the respective atoms). In the present case, since only vertices of one type are present (we will be dealing with unlabelled trees, or with trees with vertices of the same colour), **Z** will have the same constant for all  $N$  vertices, i.e. 6 (the atomic number of carbon) as an extension from alkane graphs to trees. It has been shown [20] that heteroatom substitution, i.e. a different  $Z_i$  for a given vertex  $i$ , affects markedly the value of respective LOVI and those of neighbouring vertices, while remote vertices remain unaffected. In this way, heteroatoms (or labelling or "vertex colouring") can be easily taken into account.

Figures 2 and 3 present the procedure for obtaining the **AZV** and **D'ZV** LOIS, respectively, for the pentane isomers **1–3**, the numbering of vertices being unimportant at this stage.

We used the reciprocal distance matrix **D'** (instead of **D**, which has the same information content) because the numerical methods we employed for solving linear systems of equations converge rapidly to steady solutions if the diagonal terms are much larger than all other terms. The **D'** matrix [30] is obtained by replacing all the distances  $d_{ij}$  between vertices  $i$  and  $j$  in the distance matrix **D**, with their reciprocals, i.e.

$$d_{ij}^r = 1/d_{ij} \quad (i \neq j). \quad (3)$$

One notes that due to the mode of obtaining the LOVIs, equal LOVIs are necessarily assigned to equivalent vertices. On the other hand, it is not readily apparent whether nonequivalent vertices will be assigned distinct LOVIs. This brings us to the question which actually started the computational work described in the present paper: how well are nonequivalent vertices differentiated in the LOIS?

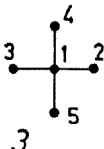
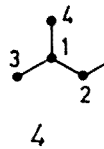
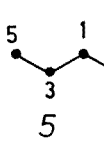
	$Q = A + Z \cdot 1$		LOVI
	$\begin{pmatrix} 6 & 1 & 1 & 1 & 1 \\ 1 & 6 & 0 & 0 & 0 \\ 1 & 0 & 6 & 0 & 0 \\ 1 & 0 & 0 & 6 & 0 \\ 1 & 0 & 0 & 0 & 6 \end{pmatrix}$	$\begin{cases} 6x_1 + x_2 + x_3 + x_4 + x_5 = 4 \\ x_1 + 6x_2 = 1 \\ x_1 + 6x_3 = 1 \\ x_1 + 6x_4 = 1 \\ x_1 + 6x_5 = 1 \end{cases}$	$\begin{cases} x_1 = 0.6250 \\ x_2 = 0.0625 \\ x_3 = 0.0625 \\ x_4 = 0.0625 \\ x_5 = 0.0625 \end{cases}$
	$\begin{pmatrix} 6 & 1 & 1 & 1 & 0 \\ 1 & 6 & 0 & 0 & 1 \\ 1 & 0 & 6 & 0 & 0 \\ 1 & 0 & 0 & 6 & 0 \\ 0 & 1 & 0 & 0 & 6 \end{pmatrix}$	$\begin{cases} 6x_1 + x_2 + x_3 + x_4 = 3 \\ x_1 + 6x_2 + x_5 = 2 \\ x_1 + 6x_3 = 1 \\ x_1 + 6x_4 = 1 \\ x_1 + x_2 + 6x_5 = 1 \end{cases}$	$\begin{cases} x_1 = 0.4281 \\ x_2 = 0.2409 \\ x_3 = 0.0953 \\ x_4 = 0.0953 \\ x_5 = 0.1265 \end{cases}$
	$\begin{pmatrix} 6 & 1 & 1 & 0 & 0 \\ 1 & 6 & 0 & 1 & 0 \\ 1 & 0 & 6 & 0 & 1 \\ 0 & 1 & 0 & 6 & 0 \\ 0 & 0 & 1 & 0 & 6 \end{pmatrix}$	$\begin{cases} 6x_1 + x_2 + x_3 = 2 \\ x_1 + 6x_2 + x_4 = 2 \\ x_1 + 6x_3 + x_5 = 2 \\ x_2 + 6x_4 = 1 \\ x_3 + 6x_5 = 1 \end{cases}$	$\begin{cases} x_1 = 0.2424 \\ x_2 = 0.2727 \\ x_3 = 0.2727 \\ x_4 = 0.1212 \\ x_5 = 0.1212 \end{cases}$

Fig. 2. Obtaining the AZV-LOVIs for pentanes 3–5.

### 3.1. DISCRIMINATION OF TOPOLOGICAL NONEQUIVALENT VERTICES WITH LOCAL VERTEX INVARIANTS (LOVIs)

The discrimination of nonequivalent vertices is important for many computational applications of chemical graph theory. To list a few:

(i) calculation of the expected number of signals in the  $^{13}\text{C}$ -NMR-spectrum for a given compound;

(ii) assessing the “topological” influence of a substitution at a given position with a labelled atom, a heteroatom, or a substituent, at various other positions in a molecule;

(iii) different enumeration problems such as trees, identity trees, rooted trees, rooted trees on a specified vertex, identity trees (i.e. trees in which all vertices are topologically distinct), homeomorphically irreducible trees (i.e. trees with no vertex of degree two), or any desired combination. An example of such a “desired” combination might be the search for identity 4-trees (alkanes) with at least one quaternary and two tertiary vertices.

Application (i) has recently received attention in the literature [31] and a correlation between the value of the chemical shift and the magnitude of the LOVI



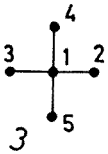
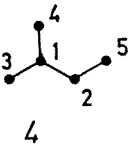
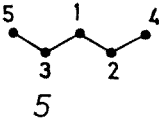
	$Q = D^r + Z \cdot 1$		LOVI
	$\begin{vmatrix} 6 & 1 & 1 & 1 & 1 \\ 1 & 6 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ 1 & \frac{1}{2} & 6 & \frac{1}{2} & \frac{1}{2} \\ 1 & \frac{1}{2} & \frac{1}{2} & 6 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 6 \end{vmatrix}$	$\begin{cases} 6x_1 + x_2 + x_3 + x_4 + x_5 = 4 \\ x_1 + 6x_2 + \frac{1}{2}x_3 + \frac{1}{2}x_4 + \frac{1}{2}x_5 = 1 \\ x_1 + \frac{1}{2}x_2 + 6x_3 + \frac{1}{2}x_4 + \frac{1}{2}x_5 = 1 \\ x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_3 + 6x_4 + \frac{1}{2}x_5 = 1 \\ x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_3 + \frac{1}{2}x_4 + 6x_5 = 1 \end{cases}$	$x_1 = 0.6341$ $x_2 = 0.0488$ $x_3 = 0.0488$ $x_4 = 0.0488$ $x_5 = 0.0488$
	$\begin{vmatrix} 6 & 1 & 1 & 1 & \frac{1}{2} \\ 1 & 6 & \frac{1}{2} & \frac{1}{2} & 1 \\ 1 & \frac{1}{2} & 6 & \frac{1}{2} & \frac{1}{3} \\ 1 & \frac{1}{2} & \frac{1}{2} & 6 & \frac{1}{3} \\ \frac{1}{2} & 1 & \frac{1}{3} & \frac{1}{3} & 6 \end{vmatrix}$	$\begin{cases} 6x_1 + x_2 + x_3 + x_4 + \frac{1}{2}x_5 = 3 \\ x_1 + 6x_2 + \frac{1}{2}x_3 + \frac{1}{2}x_4 + x_5 = 2 \\ x_1 + \frac{1}{2}x_2 + 6x_3 + \frac{1}{2}x_4 + \frac{1}{3}x_5 = 1 \\ x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_3 + 6x_4 + \frac{1}{3}x_5 = 1 \\ \frac{1}{2}x_1 + x_2 + \frac{1}{3}x_3 + \frac{1}{3}x_4 + 6x_5 = 1 \end{cases}$	$x_1 = 0.4319$ $x_2 = 0.2365$ $x_3 = 0.0649$ $x_4 = 0.0649$ $x_5 = 0.0840$
	$\begin{vmatrix} 6 & 1 & 1 & \frac{1}{2} & \frac{1}{2} \\ 1 & 6 & \frac{1}{2} & 1 & \frac{1}{3} \\ 1 & \frac{1}{2} & 6 & \frac{1}{3} & 1 \\ \frac{1}{2} & 1 & \frac{1}{3} & 6 & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & 1 & \frac{1}{4} & 6 \end{vmatrix}$	$\begin{cases} 6x_1 + x_2 + x_3 + \frac{1}{2}x_4 + \frac{1}{2}x_5 = 2 \\ x_1 + 6x_2 + \frac{1}{2}x_3 + x_4 + \frac{1}{3}x_5 = 2 \\ x_1 + \frac{1}{2}x_2 + 6x_3 + \frac{1}{3}x_4 + x_5 = 2 \\ \frac{1}{2}x_1 + x_2 + \frac{1}{3}x_3 + 6x_4 + \frac{1}{4}x_5 = 1 \\ \frac{1}{2}x_1 + x_2 + \frac{1}{3}x_3 + \frac{1}{3}x_4 + 6x_5 = 1 \end{cases}$	$x_1 = 0.2342$ $x_2 = 0.2538$ $x_3 = 0.2538$ $x_4 = 0.0871$ $x_5 = 0.0871$

Fig. 3. Obtaining the  $D^rZV$ -LOVIs for pentanes 3–5.

is to be expected. In other words, a special LOIS could be tailored so as to give a good correlation for a certain class of compounds. This semi-empirical LOIS remains to be devised.

To illustrate application (ii), the AZV-LOVIs for the smallest identity alkane graph (the heptane **12**) are given in fig. 4 together with those of the corresponding aza- and bora-analogues. An inspection of the numerical values of the AZV-LOVIs in fig. 4 shows that a large deviation from the values in the parent (unsubstituted) compound **12** occurs in the heterocyclic analogues only if the vertices represent the point of substitution. A somewhat smaller deviation is encountered for the adjacent vertices, while remote vertices have practically the same LOVIs as the parent graph **12**.

Ammonium cations or borate anions (if tetracoordinated heteroatoms are allowed) are also differentiated from the carbon analogues, as shown in fig. 5 for the undecane **13**.

If isotopic labelling of alkanes is to be considered, a different LOIS can be constructed so that the mass differences between isotopes are taken into account. Such a LOIS is the analogous AMV-LOIS, where  $M$  stands for the atomic mass which replaces the diagonal elements in the adjacency matrix.



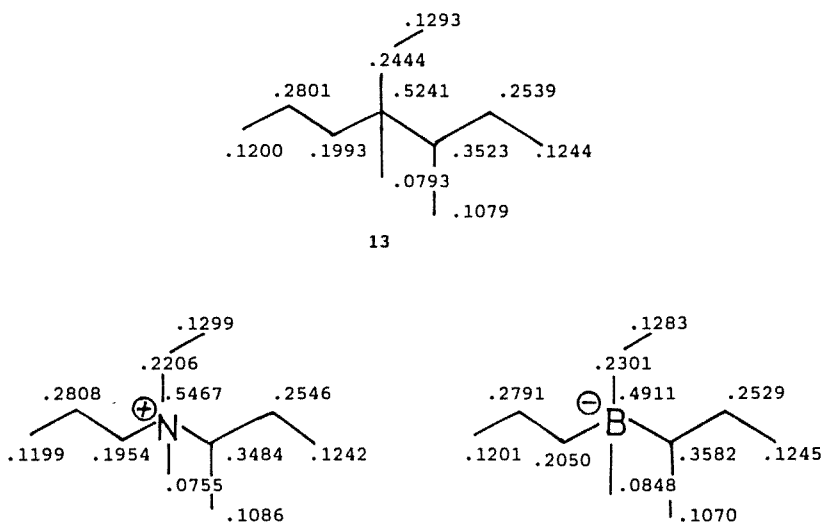


Fig. 5. AZV-LOVIs for undecane 13 and the corresponding ammonium cation and borate anion.

Regarding application (iii) related to different enumeration problems, we shall present in greater detail the results obtained with AZV and  $D^rZV$  LOISs, on which the graph generator GENLOIS is based.

### 3.2. DISCRIMINATION OF STRUCTURES WITH TOPOLOGICAL INDICES (TIs)

After differentiating the vertices by means of LOVIs, these values can be combined into a global index with a mathematical operation. This was done in a previous paper [20] with operations such as the simple addition of LOVIs, or more complex ones which took into account the molecular topology. One such example is the Randić formula [32], in which the summation of the reciprocal geometrical means of vertex degrees is done for all pairs of adjacent vertices.

For discriminating structures in the present algorithm (GENLOIS), we used the TI named  $V$ , which was recently devised by A.T. Balaban [21]. It appears that no degeneracy is encountered for  $V$  for alkanes with less than 17 vertices. This index is calculated by applying the Randić formula

$$V = \frac{q}{\mu + 1} \sum_{i \neq j} (v_i v_j)^{-1/2} \quad (4)$$

to local vertex invariants  $v_i$  which represent the local information on the magnitude of distances of vertex  $i$  to all other vertices in the graph. By analogy to the A.T. Balaban index  $J$  [33], in eq. (4)  $q$  represents the number of edges and  $\mu$  is the cyclomatic number of the graph. For trees, evidently  $q = N - 1$  and  $\mu = 0$ . The invariants  $v_i$  are calculated by

$$v_i = s_i \log_2 s_i - u_i, \quad (5)$$

where  $s_i$  are the distance sums (the sums over the rows of the **D** matrix),  $\log_2$  is the binary logarithm, and  $u_i$  represent the mean local information on the magnitude of distances (in bits). The latter invariants  $u_i$  are calculated using Shannon's formula

$$u_i = - \sum_j \frac{j}{s_i} \log_2(j/s_i). \quad (6)$$

Bonchev and Trinajstić [34] have used Shannon's formula extensively for devising TIs. Information theoretic indices were reviewed in a book by Bonchev [28].

GENLOIS, using index  $V$ , correctly generated all alkanes with up to seventeen vertices and all trees with up to fifteen vertices. The algorithm calculates for each structure the global TI ( $V$  in our case) and searches the list of TIs previously stored in an INDEX FILE. If for that structure  $V$  is not encountered in the INDEX FILE, then that structure was not previously found and its TI is appended to the list (the TI in INDEX FILE while its structure is stored by means of a code in a STRUCT FILE). The fact that only one number (albeit in double precision) is stored and compared makes the present approach quite efficient. However, we do not yet know if this index  $V$  still works for higher  $N$  values than those presented in the tables. By a constructive method, we have found graphs with eighteen vertices which present degeneracy for any TI that is calculated by means of the distance vector [35], as is the case for  $V$ . Accordingly, the octadecanes will not be correctly generated. This might be circumvented by calculating two indexes (LOIS based and  $V$ , for example), followed by comparing and storing both numbers.

#### 4. Description of the program GENLOIS and results

Figure 6 presents the block diagram of the program and a brief description of these blocks follows.

The program starts by asking in *Block 1* if there exist previously generated structures with  $N$  vertices stored by means of their LINKCODES (vide infra) in a STRUCT FILE. If this is not the case,  $N$  is set to three and propane is assigned as the only existing graph. If a STRUCT FILE exists, its name and the number of structures on  $N$  vertices listed in that file have to be specified. Also, the program asks if trees are desired or the search is limited to 4-trees (alkanes).

In *Block 2*, the following counters are initialized:

- NT – number of trees/alkanes
- NR – number of rooted trees/alkanes
- NR<sub>1</sub> – number of trees/alkanes with a primary root
- NR<sub>2</sub> – number of trees/alkanes with a secondary root
- NR<sub>3</sub> – number of trees/alkanes with a tertiary root

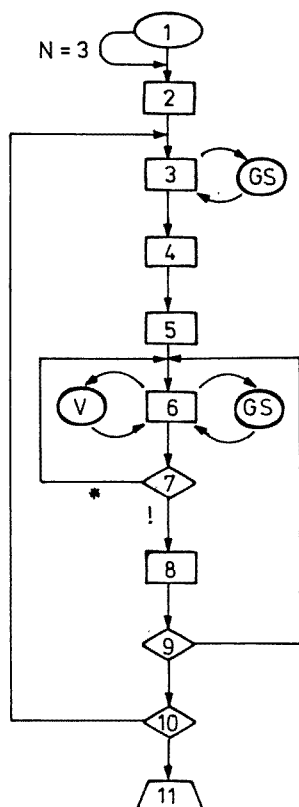


Fig. 6. Block diagram of GENLOIS.

$NR_4$  – number of trees/alkanes with a quaternary root

$NR_5$  – number of trees/alkanes with a pentenary root

$NR_6$  – number of trees/alkanes with a sextenary root

$NID$  – number of identity trees

$NIH$  – number of homeomorphically irreducible trees.

In *Block 3*, the program starts to process sequentially each of the stored graphs on  $N$  points from the STRUCT\_N FILE. Each structure is read, and its serial number and its LINKCODE are printed. Then the adjacency, distance or reciprocal distance matrices are calculated. The desired LOVIs ( $AZV$  or  $D^T ZV$ ) are then computed (in double precision) with the aid of a Gauss–Seidel subroutine (GS) which solves iteratively systems of linear equations.

In *Block 4*, the distinct LOVIs are identified, counted ( $a$ ) and sorted according to the vertex degrees ( $a_1$  on primary vertices,  $a_2$  on secondary, etc.) in order to increase the counters as follows:

$$NR = NR + a; \quad NR_1 = NR_1 + a_1; \quad NR_2 = NR_2 + a_2 \dots$$

If the investigated parent graph has  $a = N$ , then it is an identity tree and its serial number is stored in an IDENT FILE. If it has no vertices of degree two ( $a_2 = 0$ ), then the serial number of that graph is stored in a HIG FILE as being a homeo-morphically irreducible parent graph.

*Block 5* eliminates from the list of topological distinct vertices those of degree higher than three ( $a_4, \dots$ ), if only alkanes are desired.

In *Block 6*, into each position for which a distinct LOVI exists, a new vertex is added and the desired TIs (either  $V$  and/or LOIS-based TIs) are now calculated with the aid of the subroutine INDEXV and/or the same subroutine GS (but for a structure on  $N + 1$  vertices).

In *Block 7*, the TI obtained above is compared to previously calculated values for the structures on  $N + 1$  vertices stored in an INDEX FILE. If an equal value is found (i.e. that structure was generated previously), an asterisk "\*" is printed, and a new vertex is added in *Block 6*, in the next nonequivalent position.

If the TI is distinct from all previous values, *Block 8* prints a stike "!", stores the TI at the end of the INDEX FILE and the LINKCODE corresponding to its structure, at the end of STRUCT\_  $N + 1$  FILE, and increments NT.

In *Block 9*, if not all the nonequivalent ( $a$ ) positions were inspected for the parent graph currently under investigation, the next position is budded in *Block 6*. If all positions were inspected, the next parent structure in the STRUCT\_  $N$  FILE at *Block 3* is accessed. *Block 10* ensures that all the parent graphs in the STRUCT\_  $N$  FILE are investigated, after which *Block 11* outputs the counters, the CPU time, and the names of the newly generated STRUCT\_  $N + 1$  FILE and INDEX FILE, thus ending the program.

Table 1 presents the number of alkanes and labelled alkanes on certain vertices obtained with the program GENLOIS, as well as the computing times for a personal 286-AT computer equipped with a 16 MHz clock. Table 2 similarly presents the computing times and the number of trees obtained with the same program by allowing vertices of degree larger than three to be budded. The numbers fully agree with those found by other generation and/or enumeration schemes [14,17], and show that this approach functions well for alkanes with less than 17 atoms and for trees with less than 16 vertices.

The number of redundant structures generated (i.e. of "\*"), as seen in the tables, is very low when compared to other recent approaches [11]. As  $N$  increases, the number of "\*" decreases steadily from about 25% of all structures generated from pentanes, is about 11% for hexadecanes, and 13% for trees with fifteen vertices. The reason for this low redundancy lies in miminizing the "backward" generation by exploiting the LINKCODES of the structures.

Figure 7 presents the alkanes  $C_3$ – $C_9$  with their LINKCODES, while fig. 8 presents the trees with seven vertices, all structures being listed in the order of their generation. It is evident from these figures that this ordering coincides with that of







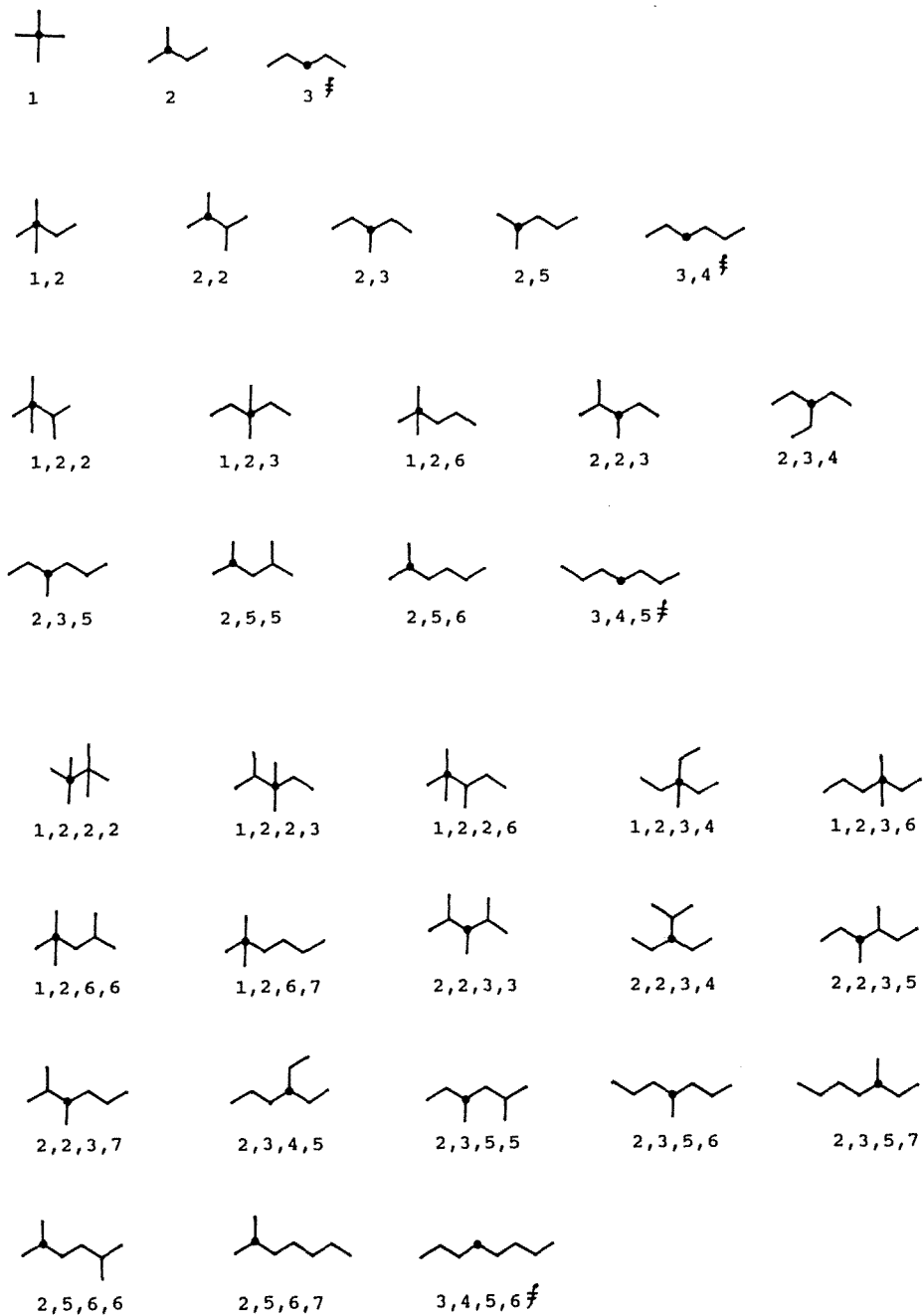


Fig. 7. Caption on next page.

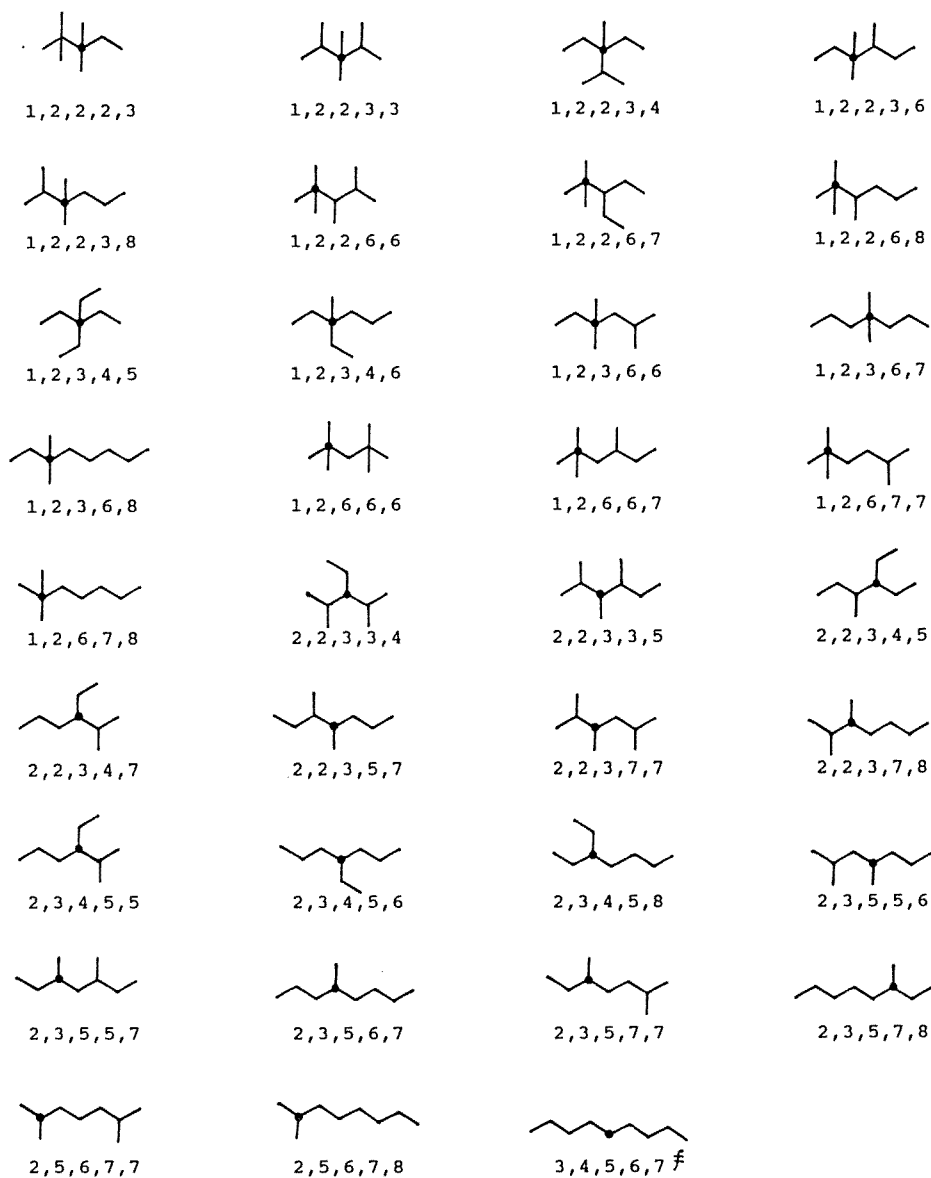


Fig. 7. Alkanes C<sub>5</sub>–C<sub>9</sub> with their abridged LINKCODES. Vertices indicated by a dot are labelled 1. ‡ For these alkanes, the first digits in the LINKCODE to be omitted were 1,1 and 2, not 1,1 and 1, as for the rest.

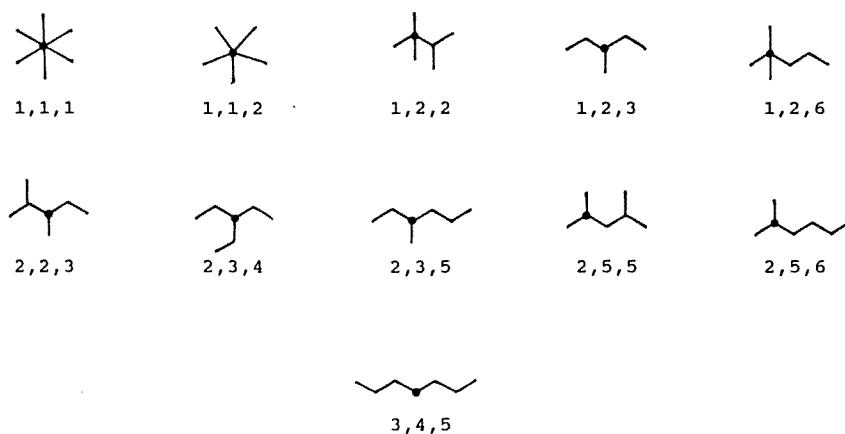


Fig. 8. Trees with seven vertices with their abridged LINKCODES.

the increase of the LINKCODES. Before returning to this ordering of structures by GENLOIS, a discussion of the LINKCODE will be made in section 5.

## 5. Cooperative labelling of vertices (indexing) and canonical coding of structures by GENLOIS with the LINKCODE

GENLOIS is a *budding* algorithm and as can be seen from the tables, it produces correctly acyclic structures. The alternative, *coding* type of algorithm used for the direct enumeration of structures produces computer codes which are translated into structures. We succeeded in reversing this translation process, namely, we extracted from the generated structures with GENLOIS their computer representation which can be viewed as a molecular code. We termed this code LINKCODE and A.T. Balaban has recently given canonical rules for it [23]. For alkanes, the LINKCODE fulfills all the requirements proposed by Read [36] so it qualifies as a “good” code. Trinajstić appreciates in his recent review on the role of graph theory in chemistry [13] that this LINKCODE is “the shortest available molecular code”. We will detail here its derivation from the GENLOIS algorithm. Kvasnička and Pospichal [10] have elaborated the formalism for “cooperative indexing” (or labelling) of graphs and we shall make use of some of their theorems.

Methane can be regarded as the initial parent graph of all structures accessed by GENLOIS. Accordingly, let us assume that the label 1 is attached to the unique carbon atom in methane. Ethane, its daughter graph, will be labelled 1-2, while propane will be cooperatively indexed as 3-1-2 (or 2-1-3). In cooperatively indexed graphs, if a vertex is assigned the label 1 and its degree is  $v_1$ , then its neighbouring vertices receive the labels 2, 3,  $\dots$ ,  $v_1 + 1$ . If the vertex labelled 2 has degree  $v_2$ , then its unlabelled adjacent vertices receive the labels  $v_1 + 2, \dots, v_1 + v_2$ , and so on until all vertices in the graph have been labelled. It has been demonstrated [10]

that if a cooperatively indexed graph with  $N$  vertices is budded, then the daughter graph is also cooperatively indexed, the label of the new vertex being  $N + 1$ . This implies that all graphs accessed by GENLOIS can be viewed as cooperatively indexed graphs. Figure 9 presents three nonanes (which are also identity trees) 14–16 with their vertex labelling as resulting from their GENLOIS predecessors, by retracing the generation scheme.

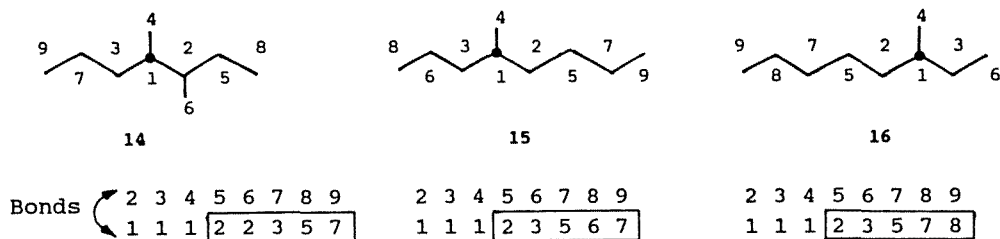


Fig. 9. Obtaining the LINKCODES and abridged LINKCODES (boxed) for the three identity nonanes 14–16.

In the list of bonds (also given in fig. 9), the LINKCODE is represented by the string of vertex labels in the lower row. For a tree with  $N$  vertices, this string of  $N - 1$  vertex labels can easily be translated into the adjacency matrix: if the  $j$ th entry in the LINKCODE is label  $i$ , then vertex  $i$  is adjacent to the vertex labelled  $j + 1$ . Since the first two bonds are always labelled 2-1-3, A.T. Balaban, in a previous paper [23], deleted (without loss of information) the first two labels 1 from the string of labels in the LINKCODE, thus using only  $N - 3$  labels to uniquely describe a tree on  $N$  vertices. If two digit labels are to be avoided (in order to optimize the disk storage space), the same author proposed a “DIFFERCODE” [23] in which each digit represents the difference between two consecutive digits in the LINKCODE. Since LINKCODES are easily reconstructed from the DIFFERCODES (and vice versa), we shall restrict the present discussion to the latter and we shall delete the first three ones in the string of  $N - 1$  labels. For the linear  $N$ -alkane (which has no vertex of degree 3), the first three digits we omitted from the LINKCODE were 1, 1 and 2.

We define the LINKCODE as being canonical, i.e. it is the minimal string (in the lexicographical sense) of all the possible cooperative labellings of a given graph. It was also demonstrated previously [10] that two graphs  $G$  and  $G'$  are isomorphic iff their canonical (LINK) codes are identical and that if a graph is canonically indexed, then the indexing is cooperative.

We were astonished by the fact that GENLOIS produces only LINKCODES (i.e. canonical labelled trees) and that the trees are generated in the strict increasing order of their LINKCODES. We can demonstrate why and when this happens by means of a theorem based on the fact that in a LINKCODE the labels are in

increasing order. If this is not true, it can be demonstrated that either the respective tree is not cooperatively indexed or that a smaller string exists which must then be the LINKCODE.

#### THEOREM

In an algorithm that generates from the complete set of trees on  $N$  vertices (parent set), which are ordered in the increasing order of their LINKCODEs, the complete set of trees on  $N + 1$  vertices (daughter set), the latter set will also be generated in the order of increasing LINKCODEs iff the procedures for discriminating nonequivalent vertices in the trees of the parent set and for discriminating the structures in the daughter set are faultless (i.e. free of assignment and operational degeneracies, respectively).

*Demonstration of the if part.* From a certain parent graph with a certain minimum code (i.e. a LINKCODE), a daughter graph is generated by adding the  $N + 1$  vertex to a certain position. The daughter LINKCODE will thus be formed by adding to the parent LINKCODE the label of the vertex to which the additional vertex is linked. Let us assume that the parent graph has the LINKCODE  $b_1 b_2 \dots b_N$  and that one of its daughter graphs receives the code  $b_1 b_2 \dots b_N b_{N+1}$ . We also assume that the generation procedure is faultless. Let us suppose that  $b_1 b_2 \dots b_N b_{N+1}$  is not the LINKCODE of that daughter graph (i.e. it is not the minimum code) and that it was not found, i.e. its TI was not in the TI list of previously generated trees on  $N + 1$  vertices. Accordingly, a different (cooperative) labelling of vertices will produce a new, smaller code  $a_1 a_2 \dots a_N a_{N+1}$  which must be the LINKCODE of that daughter graph. In other words, there must exist an index  $k < N + 1$  for which  $b_i = a_i$  ( $i = 1, k$ ) and  $b_i > a_i$  ( $i = k + 1, N + 1$ ). This implies that there must exist a parent tree having the LINKCODE  $a_1 a_2 \dots a_N$  which must precede in the list the parent graph  $b_1 b_2 \dots b_N$ , since the parent set is ordered. Then by budding the latter tree at the vertex labelled  $a_N$ , the same daughter tree should have been obtained earlier and accordingly its TI must have been in the list of daughter graphs. This contradicts either the assumption that the parent set is ordered according to the increasing LINKCODEs, or that the distinction of the vertices in the trees of the parent set, and/or the distinction of the daughter trees was faultless. A fault could happen if vertex  $a_N$  was not seen as distinct from other vertices in the parent tree and was not budded to give the daughter tree, i.e. the LOVIs must have been degenerate (assignment degeneracy). The other possibility for a faulty generation is that the global TI is degenerate so that the daughter graph's TI was found in the list, albeit it was not generated before, i.e. an operational degeneracy is encountered.

*Demonstration of the only if part.* It is evident that if operational or assignment degeneracies are encountered, an incomplete daughter set will result.

The generation of trees with  $N + 1$  vertices from the set of previously generated trees or  $N$  vertices with the program GENLOIS in fact represents the generation of

rooted trees on  $N$  vertices, followed by the inspection of the roots which can be budded to produce the next generation of trees. We thought that we could improve the speed of the program if the backward generation could be avoided, and thus bud only the roots of the parent trees which led to previously not found trees on  $N + 1$  vertices.

The above theorem enabled us to minimize the redundant graph generation by allowing only those vertices of the parent graph that have labels larger than the largest label encountered in the LINKCODE of the parent graph to be budded. In other words, if all vertices of the parent graph are allowed to be budded, the list of "stars and strikes" which allows a retracing of the generation scheme, has all the stars (\*) in front (i.e. graphs which were generated previously and their TIs are already in the list) and the strikes (!), if any, at the end (i.e. new graphs whose TIs are appended to the list). It is obvious that if a vertex of the parent graph with a small label is budded, the newly formed daughter graph will have a LINKCODE which can be derived from another parent graph (with lower ranking and which must have been budded previously) so that the TI of that daughter graph is already in the list and, accordingly, a "\*" is printed. By starting to bud only vertices with labels larger than the last label in the parent graph's LINKCODE, one cuts drastically the number of "\*"s generated, while the computing times are shortened by a factor of about three.

An alternative approach to reduce the redundant generation of graphs was tried, but it failed for alkanes with more than 13 vertices. This was done by adding the new vertex to the highest ranking vertices in the parent graph first, and storing all the new structures derived from a given parent graph. The string of ! and \* is now produced backwards (! if any in front and the \* at the end). If a structure is found to have been generated before (i.e. the first \* is encountered), the inspection of that parent graph is abandoned since only more branched trees will now be generated and these also have to be in the list. The structures which were found from that parent graph (i.e. !, in any) are then stored in the reverse order of their generation in the file STRUCT, while their indices are stored in the file INDEX, and the next parent is inspected (restarting at Block 3 in fig. 3). The computing times are somewhat larger than those listed in the tables since the number of redundant graphs now generated is close to the number of parent graphs inspected. This simple and crude scheme fails for  $N > 14$ , since minimum codes (LINKCODEs) are no longer produced for the trees. Two 13-alkanes and five 14-alkanes are encountered where the string of \* ! alternate. Thus, a tetradecane with a smaller LINKCODE that was previously generated was found from a parent tridecane with a larger parent LINKCODE. These tridecanes are presented on the left-hand side of fig. 10, together with their abridged LINKCODEs as generated by the GENLOIS program and the labels of their vertices.

In all cases, one notes that by budding the vertices with larger labels first (in decreasing order of labels), one comes to bud the vertex indicated by a square in fig. 10. The new tetradecanes that are now produced are indicated on the right-hand

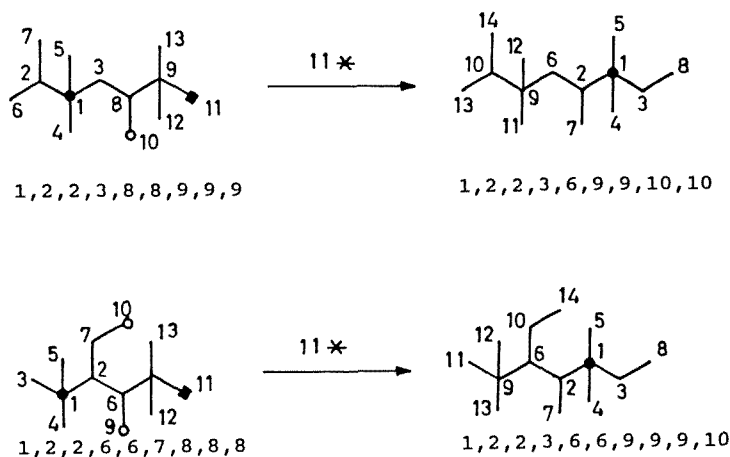


Fig. 10. Two parent tridecane from which not all daughter tetradecane are found if inspection is abandoned after encountering the first \*.

side of fig. 10, together with their minimum codes and the labelling of vertices derived therefrom. One notes that a shift of the label for vertex 1 has occurred and the new alkanes receive smaller codes than could have been derived from the parent on the left-hand side of fig. 10. Consequently, they must have been generated previously and their TI should be in the list. Accordingly, a \* is printed and that parent graph is not inspected further, although new tetradecanes, not previously generated (i.e. with LINKCODES) could have been generated by budding the parent graphs at the vertices indicated by open circles.

The result is that not all the tetradecanes are generated (one isomer is lost) and their codes are not minimal (i.e. canonical LINKCODES), so that they cannot be used for the further generation of alkanes  $C_{15}$ .

However, all  $C_{15}$  isomers are produced if another modification is made, namely that the parent graph is abandoned for further inspection only after encountering two consecutive \*\*. No theoretical basis exists for this modification and the computing times are now much larger than with GENLOIS. This scheme will probably fail when more than one shift of the centre of gravity (the label of vertex 1) can occur when passing from the parent graph to daughter graphs.

The danger of a heuristic approach to programming is that one gets carried away by its simplicity and tries to optimize one's programs even after the initial goal has been attained. The fact that nature is always a little more complicated than initially pictured is part of its beauty. In adding these remarks, we try to follow some of the advice beautifully advocated by Hoffmann [37] regarding the presentation of chemical research with more personal involvement, together with the not so "positive" results.

## 6. Structure ordering by GENLOIS by means of the LINKCODE

The ordering of structures has been a much debated problem of classical graph theory and heuristical [38], empirical [39] and mathematical [40] criteria have been proposed. The concepts of branching and molecular complexity have been defined on the basis of these ordering schemes.

The ordering of trees according to their increasing LINKCODEs, i.e. the order in which they are generated by the GENLOIS program, is different to the previous orderings. It is similar to the ordering of Trinajstić [7,17] in the respect that the star graph is generated first (it has the smallest LINKCODE  $1, 1, \dots, 1$ ) and the linear alkane is generated last (as having the largest LINKCODE  $1, 2, \dots, N - 2$ ). However, differences exist, starting with the ordering of hexanes, when 3-methylpentane (LINKCODE  $1, 1, 1, 2, 3$ ) is ordered before 2-methylpentane (LINKCODE  $1, 1, 1, 2, 5$ ), whereas the  $N$ -tuple ordering [7] places 2-methylpentane before 3-methylpentane. It is debatable whether branching is reflected in either of these ordering schemes if heuristical approaches to molecular branching are considered. The differences in ordering are amplified as  $N$  increases. The largest differences are encountered for the heptanes, for 3-ethylpentane which is ordered as fifth by the LINKCODE but only as eighth by the  $N$ -tuple representation; for the octanes, for 3-ethylhexane ranked as twelfth by the LINKCODE and as seventeenth by the  $N$ -tuple representation; and for the thirty-five nonanes, 3-ethyl-3-methylhexane is ordered as tenth by GENLOIS, while by the Trinajstić algorithm it is ranked as sixteenth.

A much closer ordering similarity to the LINKCODE ordering was encountered quite unexpectedly for the ordering advocated by Bertz [40], which is calculated by means of iterated line graphs (or graph derivatives). Bertz's procedure has the advantage of a mathematical clothing and is intuitively appealing. However, we do not agree that this ordering must reflect the "branching" of trees. Perhaps the term molecular complexity is better suited. The ordering of all structures up to and including the 18 octanes (or the 23 trees with 8 vertices) is with one exception the same, both in the order of increasing LINKCODEs or in the order of the decreasing number of graph derivatives of a sufficiently high order (larger than four). The notable exception is encountered for 3-methylhexane (abridged LINKCODE  $2, 3, 5$ ), which is ordered before 2,4-dimethylpentane (abridged LINKCODE  $2, 5, 5$ ) by GENLOIS. When calculating the graph derivatives, these heptanes receive inversed rankings. For nonanes, only five inversions (out of 47 trees) were found, the largest ranking difference being encountered for 2,2,4,4-tetramethylpentane, which was ranked as fourteenth by the LINKCODE and as tenth by Bertz's procedure. For decanes, the differences increase both in number and in rankings. However, this ordering similarity reflects some of the underlying connections between the two entirely different graph-theoretical approaches, and is worth further study.

The ordering of structures by index  $V$ , detailed in a previous paper [21], parallels to some extent the LINKCODE ordering, the smallest  $V$  value being encountered for the most "branched" structure. This similar ordering can be exploited



to avoid the comparison with a very large number of TIs ( $V$ -values) that have to be stored on disk, thus increasing considerably the computing times due to the large disk-access times. Portions of a "chopped" INDEX FILE can be stored in the main memory of the computer, and comparison of structures can be made only with this "chopped" file and still generate correctly all structures. The 10359 hexadecanes (in table 1) were generated by memorizing only 5000 TIs. After the first 5000 new hexadecanes have been generated, thus filling the batch file, a "chopping" of the first 200 indices is done and the new indices that are found are appended to this file until it is again filled, where a new chopping occurs. No attempts have been made to optimize this "chopping" procedure, but it appears to be sufficient if about half of the number of structures on  $N + 1$  vertices are memorized in the continuously chopped file. In other words, for the generation to proceed correctly two structures should not be ranked with a greater difference by index  $V$  and by the LINKCODES than the dimension of this chopped file.

## 7. Conclusions

Although the numerical results presented were known, the GENLOIS approach for generating acyclic structures has some advantages which stem from its ability to distinguish a certain type of vertex by means of its LOVI and its degree. "Customized" searches for a certain combination of vertices can be easily effected if this combination is known beforehand and appropriate counters are added to the program, which is then run from a complete set of parent trees.

Another advantage of the present approach lies in the forwardness of the generation scheme (the redundant graph generation is minimized) and in the fact that not all the generated structures need to be saved (as their TIs), which allows computations to be performed using only the main memory of a personal computer. Disk-access operations are thus avoided and quite reasonably short computing times are obtained.

The present approach is applicable also to cyclic structures. All monocyclic graphs with a limited number of vertices were obtained from the acyclic trees with the same number of vertices. These results will be presented separately [41], together with an attempt of compact coding.

## Acknowledgements

Mr. Valeriu Beiu is thanked for the various discussions related to computer programming and for his efforts to produce a PASCAL version of the GENLOIS program. The suggestions of Dr. Kirby regarding the manuscript are gratefully acknowledged. T.S.B. thanks the Alexander von Humboldt Foundation for a research grant.

## References

- [1] A. Cayley, *Phil. Mag.* 47(1874)444; *Ber. dtsch. Chem. Ges.* 8(1875)1056.
- [2] H.R. Henze and C.M. Blair, *J. Amer. Chem. Soc.* 53(1931)3042, 3077; and further papers in this series.
- [3] G. Pólya, *Acta Math.* 68(1937)145.
- [4] L.M. Masinter, N.S. Sridharan, J. Lederberg and D.H. Smith, *J. Amer. Chem. Soc.* 96(1974)7702; J. Lederberg, G.L. Sutherland, B.G. Buchanan, E.A. Feigenbaum, A.V. Robertson, A.M. Duffield and C. Djerassi, *J. Amer. Chem. Soc.* 91(1969)2973.
- [5] D.H. Smith, *J. Chem. Inf. Comput. Sci.* 15(1975)203.
- [6] Y. Kudo, Y. Hirota, S. Aoki, Y. Takada, T. Taji, I. Fujioka, K. Higashino, H. Fujishima and S.-I. Sasaki, *J. Chem. Inf. Comput. Sci.* 16(1976)50.
- [7] J.V. Knop, W.R. Müller, Ž. Jeričević and N. Trinajstić, *J. Chem. Inf. Comput. Sci.* 21(1981)91.
- [8] S.-Y. Zhu and J.-P. Zhang, *J. Chem. Inf. Comput. Sci.* 22(1982)34.
- [9] J.V. Knop, K. Szymanski, P. Křivka and N. Trinajstić, *J. Comput. Chem.* 4(1983)23; N. Trinajstić, Ž. Jeričević, J.V. Knop, W.R. Müller and K. Szymanski, *Pure Appl. Chem.* 55(1983)379; J.V. Knop, W.R. Müller, K. Szymanski and N. Trinajstić, *J. Chem. Inf. Comput. Sci.* 30(1990)159.
- [10] V. Kvasnička and J. Pospichal, *J. Chem. Inf. Comput. Sci.* 30(1990)99.
- [11] J.B. Hendrickson and C.A. Parks, *J. Chem. Inf. Comput. Sci.* 31(1991)101.
- [12] D.H. Rouvray, *J. Chem. Soc. Rev.* 3(1974)355.
- [13] N. Trinajstić, *Rep. Mol. Theory* 1(1990)185.
- [14] A.T. Balaban (ed.), *Chemical Applications of Graph Theory* (Academic Press, London, 1976).
- [15] F. Harary and E.M. Palmer, *Graphical Enumeration* (Academic Press, New York, 1973).
- [16] N. Trinajstić, *Chemical Graph Theory* (CRC Press, Boca Raton, 1983).
- [17] J.V. Knop, W.R. Müller, K. Szymanski and N. Trinajstić, *Computer Generation of Certain Classes of Molecules* (SKTH, Zagreb, 1985).
- [18] G. Pólya and R.C. Read, *Combinatorial Enumeration of Graphs, Groups and Chemical Compounds* (Springer, Berlin, 1987).
- [19] H.L. Morgan, *J. Chem. Doc.* 6(1965)107; A.L. Goodson, *J. Chem. Inf. Comput. Sci.* 20(1980)167.
- [20] P.A. Filip, T.S. Balaban and A.T. Balaban, *J. Math. Chem.* 1(1987)61.
- [21] A.T. Balaban and T.S. Balaban, *J. Math. Chem.* 8(1991)383.
- [22] E.H. Neville, *Proc. Cambridge Phil. Soc.* 49(1953)381.
- [23] A.T. Balaban, *J. Mol. Struct. (THEOCHEM)* 165(1988)243.
- [24] A.T. Balaban, *Theor. Chim. Acta* 53(1979)355; *J. Mol. Struct. (THEOCHEM)* 120(1985)117.
- [25] A.T. Balaban, I. Motoc, D. Bonchev and O. Mekenyan, *Topics Curr. Chem.* 114(1983)21.
- [26] A.T. Balaban, A. Chiriac, I. Motoc and Z. Simon, *Steric Fit in QSAR, Lecture Notes in Chemistry*, Vol. 15 (Springer, Berlin, 1980).
- [27] L.B. Kier and L.H. Hall, *Molecular Connectivity in Chemistry and Drug Research* (Academic Press, New York, 1976); *Molecular Connectivity in Structure-Activity Analysis* (Research Studies Press, Chichester, 1986).
- [28] D. Bonchev, *Information Theoretic Indices for Characterisation of Chemical Structures* (Research Studies Press, Chichester, 1983).
- [29] V.E. Golender, V.V. Drgoblav and A.B. Rosenblit, *J. Chem. Inf. Comput. Sci.* 21(1981)196.
- [30] D. Plavšić, S. Nikolić and N. Trinajstić, *J. Math. Chem.*, submitted; O. Ivanciuc, T.S. Balaban and A.T. Balaban, *J. Math. Chem.*, submitted.
- [31] L.H. Hall, B. Mohny and L.B. Kier, *J. Chem. Inf. Comput. Sci.* 31(1991)76.
- [32] M. Randić, *J. Amer. Chem. Soc.* 97(1975)6609; *Int. J. Quant. Chem.* S5(1978)245.
- [33] A.T. Balaban, *Chem. Phys. Lett.* 89(1982)399; *Pure Appl. Chem.* 55(1983)199.

- [34] D. Bonchev and N. Trinajstić, *J. Chem. Phys.* 67(1977)4517; *Int. J. Quant. Chem. Symp.* 12(1978)293, 16(1982)463;  
D. Bonchev, O. Mekenyan and N. Trinajstić, *J. Comput. Chem.* 2(1981)127.
- [35] O. Ivanciuc, T.S. Balaban and A.T. Balaban, *J. Math. Chem.*, in press.
- [36] R.C. Read, *J. Chem. Inf. Comput. Sci.* 23(1983)135.
- [37] R. Hoffmann, *Angew. Chem.* 100(1988)1653.
- [38] M. Randić, *J. Amer. Chem. Soc.* 97(1975)6609.
- [39] D. Bonchev and N. Trinajstić, *J. Chem. Phys.* 67(1977)4517.
- [40] S.H. Bertz, *Discr. Appl. Math.* 19(1988)65.
- [41] T.S. Balaban, P.A. Filip, O. Ivanciuc and A.T. Balaban, to be published.