

An Approach to the Assignment of Canonical Connection Tables and Topological Symmetry Perception

CRAIG A. SHELLEY and MORTON E. MUNK*

Department of Chemistry, Arizona State University, Tempe, Arizona 85281

Received June 25, 1979

Many nonnumerical computer applications in chemistry require algorithms for topological symmetry perception (constitutional symmetry, the automorphism partitioning problem), the assignment of canonical connection tables (the coding problem), or the detection of graph isomorphism. Important principles in designing such algorithms (vertex-classification, depth-first construction of sequence number permutations, and the use of automorphisms to restrict the construction process) are presented. In addition, a detailed description is presented for the algorithm used in program CASE for the assignment of canonical connection tables and topological symmetry perception.

Representing and perceiving the connectivity of molecular structure is of substantial importance to many nonnumerical computer applications in chemistry as indicated by the sizable number of papers on the subject. This paper is concerned with two problems, the assignment of canonical connection tables and topological symmetry perception. In addition to describing an approach to both problems, it is hoped that this paper might be beneficial to chemists only recently encountering these two challenging problems. Readers interested in pursuing the subject in greater detail are referred to a bibliography compiled for these problems.¹ Although mathematical and computer science terms will be used, they will first be defined using expressions more familiar to the chemist.

The connectivity of a molecule may be represented by a graph. A graph consists of a set of vertices (atoms) and a set of edges (bonds) connecting the constituent vertices. The graph isomorphism problem is to discover a "good" algorithm for determining if two graphs are isomorphic;² i.e., do they have identical connectivity and constituent vertices. A "good" algorithm has a worst-case time complexity bounded by a polynomial function of the size of the problem (n). The worst-case time complexity of an algorithm is the maximum number of computational steps required to solve any problem of size n .³ For graph isomorphism, the size of the problem (n) is the number of vertices in the graph. An "inefficient" algorithm is bounded by an exponential (or worse) function of the size. No "good" algorithm for graph isomorphism is known, but a linear time algorithm exists for planar graphs.⁴ (A planar graph can be drawn without crossing edges. Most molecules are planar graphs.) The computational complexity of algorithms has been covered in detail for the chemist by Aho.⁵ In addition, Read and Corneil have discussed the complexity of the graph isomorphism problem in some detail.²

The graph isomorphism problem is algorithmically equivalent to the automorphism partitioning problem (topological symmetry perception).² To partition the vertices of graph G into classes such that vertices V_i and V_j belong to the same class if and only if an automorphism exists between V_i and V_j is called the automorphism partitioning problem.² An automorphism exists between two vertices (i.e., the two vertices are topologically equivalent), V_i and V_j , of a graph G if a duplicate graph G' of G can be laid upon G so that vertex V_i' lies on vertex V_j of the same type, and each other vertex of G' lies on a vertex of G with the same type, and each edge of G' lies on an edge of G of the same type.⁵

After labeling the n vertices of a graph with integers 1, 2, 3, . . . , n (the sequence number assignment), the graph can be represented by a connection table (e.g., see Table I for the connection table of cyclopropane). Each connection table represents one and only one graph. The assignment of a

Table I. A Connection Table for Cyclopropane

| sequence no. | connected vertices | connections | element |
|--------------|--------------------|-------------|---------|
| 1 | (2,3) | 2 | C |
| 2 | (1,3) | 2 | C |
| 3 | (1,2) | 2 | C |

canonical connection table (i.e., the assignment of a code such that two graphs are assigned the same code if and only if they are isomorphic) has been called the coding problem.² The solution to the coding problem is trivial if the vertices of the graph can be assigned sequence numbers in a unique manner independent of the original assignment. A graph of n vertices can be associated with $n!$ different sequence number assignments. If no automorphism exists in the graph, each permutation of sequence numbers will be associated with a different connection table. On the other hand, where automorphisms exist, some of the permutations will necessarily lead to the same connection table.

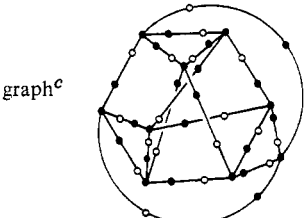
For two sequence number permutations giving the same connection table, two vertices with the same sequence number are automorphic by definition. For example, the two sequence number permutations shown in Figure 1 for cyclopropane prove that all vertices are automorphic. A rigorous method to derive the automorphism partitioning is to construct all $n!$ permutations and extract the one-to-one correspondences as above. However, most graph isomorphism algorithms (and also coding algorithms) use a heuristic approach to potentially reduce the possible one-to-one mappings of the vertices. Any graph invariant can be used to reduce the one-to-one mappings. A graph invariant is any property which does not depend on the sequence number assignment,² e.g., elemental type. The graph invariants of isomorphic graphs must necessarily be equivalent, e.g., two isomorphic graphs must have the same number of elements of each type. No graph invariant is known which is sufficient to establish isomorphism² (or the equivalent problem, the automorphism partitioning). Adjacency information can also be used to refine partitions based on graph invariants by an iterative vertex-classification method.² The Morgan algorithm,⁶ used by Chemical Abstracts Service, assigns "extended connectivity values" by initially setting the extended connectivity of each vertex to the number of adjacent vertices (the degree of the vertex). The next step assigns a new extended connectivity value for each vertex by summing the previous extended connectivity values of adjacent vertices. This step is repeated until no additional discrimination is achieved between vertices.

Although efficient, the Morgan vertex-classification scheme can lose information, e.g., it can exhibit nonuniform convergence, oscillatory behavior and nonautomorphic vertices



Figure 1. Two permutations which prove that all vertices in cyclopropane are automorphic.

Table II. Graph Where Invariants and Vertex-Classification Fail to Establish the Automorphism Partitioning



| graph invariants | | | | |
|------------------|------------------|------------------------------|------|--------------|
| element | deg ^a | cycle invariant ^b | NOON | step 1 class |
| C | 4 | 1 | 7 | 2 |
| C | 2 | 2 | 8 | 1 |
| O | 2 | 2 | 8 | 3 |

^a Equivalent to the number of two-electron covalent bonds to nonhydrogen atoms for this graph. ^b Cycles only distinguish vertices of degree 4 from 2. ^c Carbon = ●; oxygen = ○.

can possess the same extended connectivity values in every step.⁷ An algorithm^{8,9} which is somewhat more time-consuming than the Morgan classification method but is generally more discriminating follows:

- STEP 1:**
- Associate with each of the n vertices of the graph an ordered list of graph invariants.
 - Order vertices by sorting their property lists lexicographically and name the vertices V_1, V_2, \dots, V_n consistent with this ordering.
 - Given there are M distinct property lists, assign each vertex to one of M classes as follows: (i) Let $j = 1, M = 1$. (ii) Let V_k be the first vertex whose property list differs from that of vertex V_j . If none exists, let $k = n + 1$. Assign vertices V_j through V_{k-1} to class M . (iii) Let $j = k, M = M + 1$. If j is greater than n , stop. Otherwise repeat step ii.
- STEP 2:**
- Within each class of more than one member, associate with each vertex, V_i , an ordered ascending property list of integers designating the class of adjacent vertices. Let $M' = M$.
 - Let class i be the first class wherein the vertices possess $k > 1$ distinct property lists. If no such class exists, proceed to step 3.
 - Using the vertices of class i , construct k new classes, i through $i + k - 1$ by assigning vertices with identical property lists to the same class. Renumber classes $i + 1$ through m' , $i + k$ through $M' + k - 1$. Let $M' = M' + k - 1$ and repeat step 2b.
- STEP 3:** If $M' = M$, stop. Otherwise let $M = M'$ and repeat step 2.

Some graph invariants which could be used for molecules in step one are degree, elemental type, the number of two-electron covalent bonds to nonhydrogen atoms,⁷ the number of cycles of a specific length containing the vertex, and the number of the outermost occupied neighbor sphere or the "NOON" of the vertex.⁵ However, even with iterative vertex-classification (steps 2 and 3), no invariant or combination of invariants in step 1 is known which is sufficient to establish the automorphism partitioning for a graph. For example, the graph in Table II has three vertex-classifications after step 1 using all of the above graph invariants. Since each vertex in class 2 has two adjacent vertices in class 1 and two adjacent vertices in class 3, each vertex in class 1 has an adjacent vertex in classes 2 and 3 and each vertex in class 3 has an adjacent

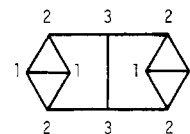


Figure 2. Vertex-classification based on cycles for a 10-vertex regular graph. Each integer represents class membership.

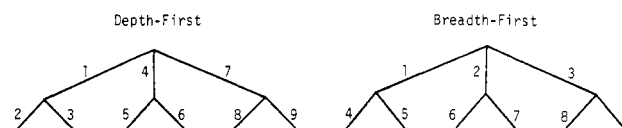
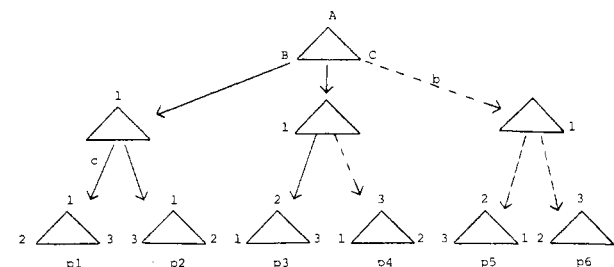


Figure 3. The breadth-first and depth-first order. Integers designate the order in which the search tree is traced.



^a Lexicographic order is the priority convention used here for assigning sequence numbers.

^b Symmetry relationships can be used to prune dashed edges from the tree.

^c The last two assignments are made in a single step to conserve space.

Figure 4. Permutation construction for cyclopropane.^a

vertex in classes 1 and 2, no additional discrimination is achieved in steps 2 and 3. In reality, no vertices are equivalent!

To determine the automorphism partitioning, each class resulting from the iterative procedure described could be analyzed for nonautomorphic vertices by resorting to the previous definition of automorphism. However, only vertices in classes of degree two or greater need to be considered since terminal vertices attached to the same vertex or automorphic vertices must be automorphic.¹⁰ By using vertex-classification, the complexity of the problem is reduced to the construction of $\Pi_i(C_i)!$ permutations where C_i is the number of vertices in class i . For example, vertex-classification based on cycles of the 10 vertex regular graph (all vertices of the same degree) shown in Figure 2 produces three classes. This drastically reduces the complexity from $10!$ (3 628 800) to $4!4!2!$ (1152) sequence number permutations to construct.

Either a breadth-first or depth-first approach can be used to construct the sequence number permutations. The construction process is a rooted tree where the root node is the unlabeled graph and each terminal node is a labeled graph corresponding to one permutation. (To avoid confusion, node will be used when referring to a vertex in the tree of the construction process.) The order in which the tree is traced for both the breadth-first and depth-first methods is shown in Figure 3. Generally, the depth-first method is preferred to decrease storage requirements. The permutation construction process is shown for cyclopropane in Figure 4. With the depth-first method, the automorphisms detected can be used to prune branches from the tree which do not detect additional automorphisms. For example, because permutations p_1, p_2 , and p_3 prove that automorphisms exist between each pair of vertices, permutations p_4, p_5 , and p_6 can be pruned from the tree. A knowledge of automorphisms derived in this manner was used to a limited extent by Wipke and Dyott in their modification of the Morgan algorithm.¹⁰

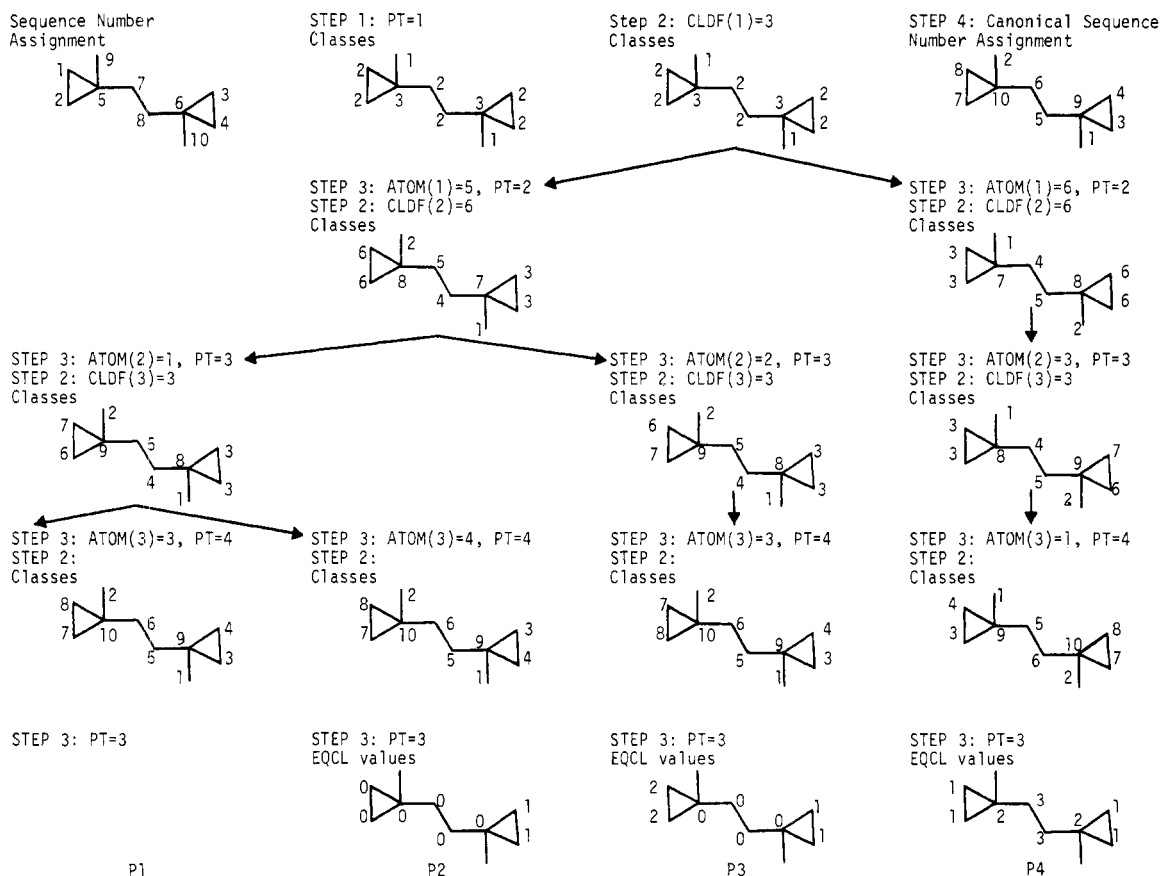


Figure 5. Tracing of the coding and automorphism partitioning algorithm.

Another technique can also be used to further constrain the permutation construction procedure. The assignment of a sequence number to a vertex distinguishes that vertex from all other vertices; i.e., it is the only member in its class for the partially labeled graph. In essence, the Morgan rule which labels vertices attached to the previously labeled vertex first is an application of this rule;⁶ i.e., the vertices adjacent to a labeled vertex must be in a different class from nonadjacent vertices. However, the vertex-classification procedure previously described can often be applied to the partially labeled graph to classify further the vertices.²

A combination of the previous methods will construct all the permutations necessary to rigorously determine the automorphism partitioning. However, in most instances, permutations which lead to different connection tables will also be constructed. The coding problem can be solved by comparing these few (hopefully) remaining connection tables and designating the one with the largest (or smallest) lexicographical order as the canonical code.⁶

One problem remains to be covered before presenting the coding and automorphism partitioning algorithm. Molecular graphs may have different edge types—single, double and triple bonds—which may be represented in a variety of ways. Many connection tables designate edge types by using labels. In this manner, a fourth edge type, aromatic, can be used to incorporate aromaticity. However, only one edge type is necessary if a pair of vertices may be attached by more than one edge. Unfortunately, these connection tables give rise to topological differences between π electron resonance forms, e.g., the two Kekulé forms of *o*-xylene. By replacing multiple edges with a single edge without altering the value designating the number of two-electron covalent bonds joining a vertex to nonhydrogen vertices, the difference between identical π resonance forms can be removed while retaining unambiguity for other chemical systems.¹¹ (Conjugated cyclic structures

which do not obey Hückel's rule are also identical using this convention.) In addition, this form of the connection table can be used to "correctly" detect the automorphism partitioning for a graph. A coding and automorphism partitioning algorithm which uses this ambiguous form of the connection table follows:

- STEP 1:
- Let $EQCL_i = 0$ for each vertex. (EQCL will designate automorphic relationships. Where $EQCL_i = EQCL_j$ and $EQCL_i \neq 0$, an automorphism exists between vertices i and j . Where $EQCL_i = 0$, no automorphism for vertex i has been detected.)
 - Let $PT = 1$. (PT is a pointer to the current level in the tree.)
 - Let $FLAG = \text{false}$.
 - Apply STEP 1 of the vertex-classification procedure and store these values in the "class stack" at level PT. (Two graph invariants are sufficient for molecules—the number of two-electron covalent bonds to nonhydrogen atoms and elemental type.)
- STEP 2:
- If not FLAG then:
- Apply STEPS 2 and 3 of the vertex-classification procedure to the class stack values at level PT.
 - Let $CLDF(PT)$ be the largest class with more than one member at level PT. If no such class exists, proceed to STEP 3.
- STEP 3:
- If FLAG or $CLDF(PT)$ was assigned a value in STEP 2b and vertices in this class have a degree greater than one, then do STEPS 3a–c. Otherwise, do STEPS 3d–f.
- If not FLAG then: (i) Let $ATOM(PT)$ equal the smallest sequence number of vertices in class $CLDF(PT) - 1$. (ii) Let $FLAG = \text{true}$.
 - Let $ATOM(PT) = ATOM(PT) + 1$.
 - If $ATOM(PT) > n$ then $PT = PT - 1$. Otherwise, if $ATOM(PT)$ is in the class $CLDF(PT)$ and a vertex with a smaller sequence number does not exist which is automorphic to $ATOM(PT)$ then: (i) Let $PT = PT + 1$. (ii) Let the class stack values at level PT be assigned the values at $PT - 1$. (iii) Let the class stack values at level

- PT which are greater than CLDF(PT - 1) be incremented by one. (iv) Let the class stack value at level PT of ATOM(PT - 1) be incremented by one. (v) Let FLAG = false.
- d. If $PT \neq 1$ then: (i) Construct with the sequence number assignment at level PT of the class stack a code for the vertices of the CT with degree greater than one. (ii) If the code is new then store it with the corresponding sequence number assignment. Otherwise, each pair of vertices with degree greater than one and with the same sequence number are automorphic and the relationship is stored in EQCL. If any ancestor of this node would have been forbidden owing to the automorphism(s) detected, then decrement PT to the level of the "forbidden" node.
- e. Let $PT = PT - 1$.
- f. Let FLAG = true.
- STEP 4: If $PT \neq 0$ then go to STEP 2. Otherwise:
- a. "Lone" vertices are automorphic if they were in the same class at the root node.
- b. Terminal vertices are automorphic if they were in the same class at the root node and they are attached to the same vertex or automorphic vertices.
- c. Automorphisms for all other vertices are designated by EQCL values.
- d. A canonical sequence number assignment for all vertices, using the sequence number assignment for the smallest code (SN), is obtained by: (i) Let $i = 1$. (ii) For each vertex j do if vertex j is terminal then let $SN_j = SN_i * n + i$, $i = i + 1$. Otherwise, let $SN_j = SN_i * n$. (iii) Assign values to SN between 1 and n according to the current order of SN values.

A tracing of this procedure is shown in Figure 5. In step 1 the vertices are partitioned into three classes using two graph invariants: the number of two-electron covalent bonds to nonhydrogen atoms and elemental type. This reduces the complexity of the problem to constructing at most $6!2!2!$ (2880) permutations. In step 2, adjacency information is unsuccessfully used in a further attempt to reduce the problem's complexity. In step 3, vertex 5 is assigned to a new class, distinguishing it from all other vertices. The conditional statement at step 4 is true, and the vertices are partitioned again in step 2. Now, vertex 1 is selected to be differentiated. The vertices are unsuccessfully partitioned in step 2, after which vertex 3 is assigned to its own class leading to permutation p_1 which is saved for future reference. After backtracking to the parent node, vertex 4, the next possible choice, is assigned to its own class which gives permutation p_2 . Permutations p_1 and p_2 give the same connection table, thus indicating that vertices 3 and 4 are automorphic. Proceeding through the remaining portion of the tree, two more permutations are constructed which reveal all other automorphisms. Since all four permutations give the same connection table, only one was saved, p_1 , which can be used to uniquely code the molecule. Thus, a combination of the principles described above reduces a potential $10!$ (3 628 800) problem to constructing only four permutations.

The coding and automorphism partitioning algorithm is efficient and particularly suited for graphs resistant to many other algorithms, as the following empirical results reveal. A representative collection of 184 organic molecules¹² was processed by a Fortran implementation of the algorithm in less

than 18 s of processor time on a UNIVAC 1100/42. (The "mean" molecule possessed 18.15 nonhydrogen atoms. The number of cycles varied from 0 to $1.6n$, where n is the number of vertices.)

Although the number of molecular graphs resistant to most coding algorithms is small (e.g., of the 677 000 structures processed in 1975 by Chemical Abstracts Services, only 0.15% could not be coded in less than 3 cpu s¹³), they can be important in some applications, e.g., computer-assisted structure elucidation.¹⁴ Cornell and Read suggested that, "The test of a graph isomorphism algorithm is how it behaves under 'worst case' conditions, i.e., how it handles the really recalcitrant graphs. . .".² The dodecahedron represents such a case, a potential $20!$ problem for many algorithms, but requires only 1980 ms for the procedure described. As further evidence of the procedure's effectiveness on "recalcitrant" graphs, it was tested on the 59 regular graphs of degree 4 which contain 10 vertices. (These graphs were constructed by program CASE,¹⁴ and were each loop-free and void of multiple edges.) Even though each of these graphs represents approximately $10!$ problems for many algorithms, the mean execution time was only 253 ms.

Although efficient, the coding and automorphism partitioning algorithm presented in this paper is not intended to replace other theoretically sound or effective algorithms. It is our hope that the principles discussed and the algorithm presented in this paper will be a beneficial guide to those designing other algorithms for nonnumerical computer applications in chemistry where coding and/or automorphism partitioning problems are encountered.

REFERENCES AND NOTES

- (1) Colbourn, C. J. "A Bibliography of the Graph Isomorphism Problem", Technical Report No. 123/78, Computer Science Department, University of Toronto, 1978.
- (2) Read, R. C.; Cornell, D. G. "The Graph Isomorphism Disease", *J. Graph Theory* **1977**, *1*, 339-363.
- (3) Aho, A. V. "Algorithms and Computational Complexity", *Acta Cryst.* **1977**, *33*, 5-12.
- (4) Tarjan, R. E. In "Algorithms for Chemical Computation"; Christoffersen, R. E., Ed.; American Chemical Society: Washington D.C., 1977; Vol. 46, Chapter 1, pp 1-20.
- (5) Jochum, C.; Gasteiger, J. "On the Misinterpretation of Our Algorithm for the Perception of Constitutional Symmetry", *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 49-50.
- (6) Morgan, H. L. "The Generation of a Unique Machine Description for Chemical Structures—A Technique Developed at Chemical Abstracts Service", *J. Chem. Doc.* **1965**, *5*, 107-113.
- (7) Randic, M. "On Unique Numbering of Atoms and Unique Codes for Molecular Graphs", *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 105-108.
- (8) Cornell, D. G.; Gottlieb, C. C. "An Efficient Algorithm for Graph Isomorphism", *J. Assoc. Comput. Mach.* **1970**, *17*, 51-64.
- (9) Shelley, C. A.; Munk, M. E. "Computer Perception of Topological Symmetry", *J. Chem. Inf. Comput. Sci.*, **1977**, *17*, 110-113.
- (10) Wipke, W. T.; Dyott, T. M. "Stereochemically Unique Naming Algorithms", *J. Am. Chem. Soc.* **1974**, *96*, 4834-4942.
- (11) Shelley, C. A.; Munk, M. E.; Roman, R. V. "A Unique Computer Representation for Molecular Structures", *Anal. Chim. Acta* **1978**, *103*, 245-251.
- (12) Shelley, C. A.; Munk, M. E. "Signal Numer Prediction in Carbon-13 Nuclear Magnetic Resonance Spectrometry", *Anal. Chem.* **1978**, *50*, 1522-1527.
- (13) O'Korn, L. J. In ref 4, Chapter 6, pp 122-148.
- (14) Shelley, C. A.; Woodruff, H. B.; Snelling, C. R.; Munk, M. E. In "Computer-Assisted Structure Elucidation"; Smith, D. H., Ed.; American Chemical Society: Washington D.C., 1977; Vol. 54, Chapter 7, pp 92-107.