(24) Kenknight, C. E. *Acta Crystallogr.* **1984**, *A40*, 708-712.
(25) Ghose, A. K.; Crippen, G. M. *J. Comput. Chem.* **1985**, *6*, 350-359.
(26) Lejeune, J.; Michael, A.; Vercauteren, D. P. *J. Comput. Chem.* **1986**, *7*, 739-744.
(27) Stouch, T. R.; Jurs, P. C. *J. Chem. Inf. Comput. Sci.* **1986**, *26*, 4-12.
(28) Hansch, C.; Leo, A. *Substituent Constants for Correlation Analysis in Chemistry and Biology*; Wiley: New York, 1979.
(29) *CRC Handbook of Chemistry and Physics*, 65th ed.; Weast, R. C., Ed.; CRC: Boca Raton, FL, 1984.
(30) *Dictionary of Organophosphorus Compounds*; Edmandson, R. S., Ed.; Chapman and Hall: London, 1988.
(31) Takeda, T.; Ohashi, Y.; Sasada, Y. *Acta Crystallogr.* **1970**, *B30*, 825-827.
(32) Miyashita, O.; Kasahara, F.; Marumoto, R. *J. Antibiot.* **1985**, *38*, 981-986.
(33) Pruisner, P.; Brennan, T.; Sundaralingam, M. *Biochemistry* **1973**, *12*, 1196-1202.
(34) Loomis, C. R.; Bell, R. M. *J. Biol. Chem.* **1986**, *263*, 1682-1692.
(35) Smulson, M. E.; Suhadolnik, R. J. *J. Biol. Chem.* **1967**, *242*, 2872-2876.
(36) Ghose, A. K.; Crippen, G. M. *J. Med. Chem.* **1984**, *27*, 901.
(37) Kato, Y.; Itai, A.; Iitaka, Y. *Tetrahedron* **1987**, *43*, 5229-5236.
(38) (a) *QCPE No. 395*; Department of Chemistry, Indiana University, Bloomington, IN. (b) Allinger, N. L. Private communication, MMP2 1985, parameters, dated June 17, 1987.
(39) Allinger, N. L.; Kok, R. A.; Imam, M. R. *J. Comput. Chem.* **1988**, *9*, 591-595.
(40) Momany, F. A.; Carruthers, L. M.; McGuire, R. F.; Scheraga, H. A. *J. Phys. Chem.* **1974**, *78*, 1595.
(41) Hooke, R.; Jeeves, T. A. *J. Assoc. Comput. Mach.* **1961**, *8*, 212.

# Review of Ring Perception Algorithms for Chemical Graphs

GEOFFREY M. DOWNS, VALERIE J. GILLET, JOHN D. HOLLIDAY, and MICHAEL F. LYNCH*

Department of Information Studies, University of Sheffield, Sheffield S10 2TN, U.K.

Received October 18, 1988

Current ring perception algorithms for use on chemical graphs concentrate on processing specific structures. In this review, the various published ring perception algorithms are classified according to the initial ring set obtained, and each algorithm or method of perception is described in detail. The final ring sets obtained are discussed in terms of their suitability for use in representing the ring systems in structurally explicit parts of generic chemical structures.

## INTRODUCTION

A necessary component of the generic chemical structure storage and retrieval system currently under development at Sheffield is a method of analysis and representation of ring systems for both specific and generic structures, such as those found in patents. The problems to be addressed are substantial. The representation should ensure consistent description of specific structures and specific ring systems within generic structures for both full and substructure searching. Furthermore, it must permit the description of those parts of generic ring systems for which full structural characterizations are given (**structurally explicit generics**), as well as being capable of extension to the characterization of those parts for which only intensional descriptions are given (**structurally implicit generics**).

Generic ring systems introduce several additional areas of complexity. A variable group may occur within a ring or may substitute onto a ring system to form additional rings, while identical generic structures may be declared different by virtue of the orientation of their partial structures and the partitioning of them into partial structures.

These circumstances call for the production of a searchable representation by means of a ring perception algorithm with the following characteristics:

- It is independent of the projection, orientation, and partitioning of the ring system.
- It is consistent in the selection of rings to ensure maximum recall with minimum false drops.
- It should permit the identification of inter-ring relationships.
- It should enable the overall logic of relationships among ring systems and acyclic parts to be represented faithfully.

A necessary preliminary stage to the production and evaluation of such an algorithm for use on structurally explicit generics is an appraisal of existing algorithms and ring sets used for specific ring systems, together with their potential for extension into a generic environment. This review is a result of such an appraisal, originally given in reference 1. Subsequent papers outline the resultant ring set and algorithm developed from this appraisal.[2-4]

An attempt has been made to standardize the different terminologies used to those of the graph theoretic concepts considered in the paper subsequent to this review. As a result, **vertex** ($v$) is used instead of atom or node, **edge** ($e$) instead of bond or arc, **connectivity** instead of degree or valency, and **nullity** ($\mu$) instead of Frerejacque number or cyclomatic number. The sets of all vertices and edges are denoted by $V$ and $E$, respectively. In addition to the definitions given below, an elementary knowledge of graph theory is assumed.

A **walk** is an alternating sequence of vertices and edges, starting and ending at vertices, where each edge is incident with the vertices on either side of it in the sequence. If the start and end vertices are the same, then the walk is closed and is called a **circuit**; otherwise, it is open. If all edges of a walk occur only once, then it is a **trail**, and if all vertices are also distinct, then it is a **path**. A closed path is a **cycle** ($v \geq 3$). If one or more vertices of a graph occur more than once in a circuit, then it contains **Doppelpunkte**.[5] A circuit without Doppelpunkte is a cycle. If a pair of vertices in a cycle is connected by an edge that does not occur in the cycle, then these vertices are **Nachbarpunkte**. A cycle without Nachbarpunkte is a **simple cycle**.

A graph is **connected** if every pair of vertices in it is joined by a path. The set of all such vertices forms a **component**. If the graph consists of several disconnected sets of vertices, then each set forms a separate component; hence, a disconnected graph has at least two components.

An edge is a **cut edge** if its removal disconnects the graph and increases the number of components; hence, it cannot be part of any cycle. Similarly, a vertex is a **cut vertex** (sometimes called an articulation point) if its removal increases the number

0095-2338/89/1629-0172$01.50/0

of components. Obviously each vertex incident to a cut edge is a cut vertex, but a cut vertex can also be part of a cycle (as a spiro-fusion). A component or subcomponent that has no cut vertices is called a **block**.

Note that in the chemical sense a **bridge** is a series of two or more edges crossing a cycle, whereas in the mathematical sense it is a series of one or more cut edges linking two blocks of a component.

Most algorithms rely on a connection table representation of the structure. This is simply a compact form of adjacency matrix that can be converted easily into the full adjacency or incidence matrix. Ring perception can then proceed by one of two fundamental methods. The first is to "walk" through the connection table, or matrix, and the second is to use graph theoretical operations on a matrix.

The walking method is the earliest and is the more obvious. The algorithm starts from some atom of the graph representation and simply walks about the structure while noting the paths traced and any branch points. If the path closes (to give a cycle, which can be stored) or cannot continue, the path is shortened to the last branch point. Walking continues along a new path from this branch point. After all paths have been traced from every branch point, walking is complete.

When used on the undirected graphs representing molecular topology, path tracing goes both ways around a ring, and so each ring is found at least twice. In addition, as the size and complexity of a ring system increase, so does the processing time. These effects can be minimized to some extent by ignoring acyclic vertices, by carefully choosing the **start atoms** (vertices from which path tracing begins), and by applying rules to ensure that paths that would be definitely unproductive are not traced.

In spite of potential problems in controlling the walks, some of the benefits of this approach are that it can work directly from a connection table and will produce an ordered list of the constituent atoms and bonds around the perceived rings.

The more mathematical graph theoretic approach uses matrices, trees, and sets to perceive the required set of rings. Most algorithms applying graph theory use a matrix to derive an acyclic **spanning tree**. The **chords** (i.e., minimum number of edges whose removal is required to turn the structure from cyclic to acyclic) of the resultant spanning tree are used to find a fundamental basis of cycles. A fundamental basis contains the minimum number of linearly independent cycles that cover every ring vertex and edge in the structure. The set cardinality is given by the nullity, $\mu$, and corresponds to the number of chords, i.e.

$$\mu = \text{no. of edges} - \text{no. of vertices} + \text{no. of components}$$

**Fundamental Bases Are Generally Nonunique.** Further cycles can be found by using a vector–space algorithm to take all ring sum combinations of the basis cycle vectors, by growing further trees from the ends of the chords, or by deriving further spanning trees from different start atoms to obtain further fundamental bases.

The rings found by the vector–space method are given in vector form and are thus unordered sequences. Further processing is required to give the ordered atom–bond sequence for each cycle. However, vector processing is inherently fast, and there are several techniques for increasing the efficiency of the additional processing.

The various ring perception algorithms developed are largely a result of differing requirements. In this review, ring perception techniques are presented according to the initial approach used to find the required set of rings for each structure. These initial approaches are as follows:

- to find all possible cycles and then select those required
- to find all possible simple cycles and then select those required
- to generate a fundamental basis of cycles from which all other cycles can be derived as necessary
- to determine directly the smallest fundamental basis, known as the smallest set of smallest rings

Within these four categories, the techniques are ordered by year and presented by author since most have not been given any specific name. Table I gives the necessary structured overview of the major aspects of the 24 techniques reviewed.

Previous reviews of ring perception algorithms used on chemical graphs include those given by Carruthers[6] (the most thorough to date), Sorkau,[7] and Gray.[8] Most of the original papers give introductory reviews, the best of which can be found prior to the algorithms presented by Corey and Petersson,[9] Gasteiger and Jochum,[10] Roos-Kozel and Jorgensen,[11] Wipke and Dyott,[12] and Zamora.[13]

A complete comparison of the algorithms, to assess the computational efficiency of each, is not possible on the basis of the published papers. Many do not give any indications of performance, and those that do are not directly comparable due to such factors as the programming language used, the computer used, and the amount of additional processing included (such as aromaticity detection). The purpose of this appraisal is thus to ascertain their appropriateness for extension into a generic environment, rather than to code and implement as many as possible to compare their efficiency for processing specific structures.

The review by Mateti and Deo[14] does give a comparison of performance in terms of the upper bounds on time and space, with a useful tabulated summary. However, only 1 of the 21 algorithms they mention has been used for chemical structures (by Welch); most are designed for use on directed graphs, and many have been superseded by later publications. Most of these algorithms will only be referenced here. In addition, Read and Tarjan[15] consider the theoretical upper bounds and present suitably efficient algorithms for listing cycles, paths, and spanning trees.

## ALGORITHMS THAT INITIALLY PERCEIVE ALL CYCLES

Perception of all cycles in a structure may be very time-consuming, but it does at least ensure thorough analysis. It is one of the earliest techniques used and in many ways is the simplest to implement. Those cycles required to categorize the structure can then be selected from the complete set of cycles as desired.

**Tiernan**[16] presents an algorithm designed to process directed graphs rather than the undirected graphs used to represent chemical structures. It is mentioned here for completeness, as a representative directed graph algorithm, and because it was claimed to be the "theoretically most efficient search algorithm". Due to the directed nature of the graph, the path trace is easier to implement and finds each cycle only once. The algorithm can also cope with self-loops. Conversion to an undirected graph would make this less efficient than other methods such as Paton's (see later).

The connection table has some arbitrary numbering of the vertices and is converted to a $V \times V$ adjacency matrix. In addition, there is a list of length $V$ to store the current sequence and a $V \times V$ array to store the list of vertices closed to each vertex.

Path tracing starts from the first vertex in the adjacency matrix. Choice of the next extension vertex is governed by the following rules:

- It must not already be in the path (to ensure that a cycle is traced).
- Its label must be larger than the first vertex in the path (to ensure that each cycle is traced once only—starting from its lowest vertex label).

**Table I.** Comparative Tabulation of Ring Perception Techniques

| author | perception method | initial ring set | final ring set | implemented | comments |
|---|---|---|---|---|---|
| Tiernan (1970) | walk through connection table | all cycles | all simple cycles | | for use on digraphs |
| Corey, Wipke, Cramer, and Howe (1972) | walk through connection table | all cycles | real rings (maximum proper covering set) | LHASA synthesis program | real/pseudoring concept has problems |
| Shelley (1983) | graph theory (spanning tree) | all cycles | set of $\mathcal{K}$-rings? (see Plotkin) | graphic display program | to generate coordinates |
| Balaban, Filip, and Balaban (1985) | graph theory (edge combinations) | all cycles | all cycles | desk-top computer | for use on basic graphs |
| Fujita (1988) | graph theory | all cycles (Wipke and Dyott) | essential set of essential rings | for synthesis programs | complex ring set definition |
| Nickelsen (1971) | not known | all simple cycles | $\beta$-rings | GREMAS, to replace fundamental ring concept | $\beta$-ring a useful concept |
| Welch (1966) | graph theory (incidence matrix) | fundamental basis | all cycles | Wipke and Dyott's algorithm | manipulates matrices |
| Gotlieb and Corneil (1967) | graph theory (adjacency matrix) | fundamental basis | fundamental basis | in Gibbs' algorithm | manipulates matrices |
| Fugmann, Dolling, and Nickelsen (1967) | graph theory (half-ring tracing) | fundamental basis | fundamental rings (not the same as a fundamental basis) | GREMAS search system | led to the $\beta$-ring concept |
| Gibbs (1969) | graph theory (adjacency matrix) | fundamental basis (Paton's method) | all cycles | in Wipke and Dyott's algorithm | takes ring sum of basis vectors |
| Paton (1969) | graph theory (adjacency matrix) | fundamental basis | fundamental basis | in Corey and Petersson's algorithm | manipulates matrices |
| Corey and Petersson (1972) | graph theory (adjacency matrix) | fundamental basis (Paton's method) | SSSR + all ≤7-edged rings | LHASA synthesis program | trees from chord for extra rings |
| Wipke and Dyott (1975) | graph theory (incidence matrix) | fundamental basis (Welch's method) | all cycles then SSSR + all ≤8-edged rings | in SECS synthesis program | considers components separately |
| Gasteiger and Jochum (1979) | graph theory | fundamental basis | SSSR | in EROS synthesis program | inconsistent SSSR tracing |
| Sorkau (1985) | graph theory | SSSR | SSSR | improvements to Gasteiger and Jochum | mostly a review paper |
| Plotkin (1971) | graph theory | SSSR | set of $\mathcal{K}$-rings | CIDS search system | $\mathcal{K}$-ring concept very useful |
| Bersohn (1973) | walk through connection table | SSSR | as for Corey and Petersson | for synthesis programs | tracing avoids envelopes |
| Esack (1975) | walk through connection table | SSSR | as for Corey and Petersson | for synthesis programs | extension to Bersohn |
| Zamora (1976) | walk through connection table | SSSR | SSSR | CAS Registry system | uses 3-phase path tracing |
| Schmidt and Fleischhauer (1978) | graph theory (adjacency matrix) | SSSR | SSSR | | manipulates matrices |
| Roos-Kozel and Jorgensen (1981) | graph theory (connection table) | SSSR | set of $\mathcal{K}$-rings (Plotkin) | CAMEO synthesis program | problems with complex graphs |
| Deo, Prabhu, and Krishnamoorthy (1982) | breadth-first growth of spanning tree | SSSR | SSSR | to study efficiency of growth methods | compares use of heuristics |
| Hendrickson, Grier, and Toczko (1984) | graph theory (maximal adjacency matrix) | SSSR | SSSR | TRGEN procedure on minicomputer | manipulates matrix |
| Elk (1984) | planar projection | SSSR | SSSR | visual aid | SSSR theory and projection |

- It cannot be closed to the last vertex in the path (to ensure that no path is considered more than once).

At some point no further suitable vertex will be available. A check is made to see whether the first and last vertices have a common incident arc; if so, then a cycle has been found and can be listed. Either way, "vertex closure" (i.e., completion of that particular path) is performed as follows: (1) Assign the last vertex to the closed vertex list of the penultimate vertex (to prevent the path being traced again). (2) Clear the closed vertex list of the last vertex (to allow the last vertex to be considered again if it occurs in another cycle). (3) Delete the last vertex from the path-trace.

These stages operate in a similar manner to the backtracking to previous choice points outlined in the next algorithm (Corey et al.) except that, when vertex 1 is reached again, the process is repeated for the next vertex in the connection table rather than only for a vertex in another component.

A similar adjacency matrix backtracking algorithm for finding all cycles in a directed graph is that by Beiss, Jänicke, and Meissler.[17] This paper includes comparative timings to show that it is faster than Tiernan's algorithm. Several mathematically based algorithms are available for application to the directed graphs of, for instance, electrical networks, and operate by manipulation of an adjacency matrix. It is easy to find all circuits by using such an approach, but additional computational overheads are involved in selecting the cycles from the circuits. The difference between these algorithms is in the methods employed to obtain just the cycles rather than all circuits. Danielson,[18] for instance, uses a variable adjacency matrix, i.e., an adjacency matrix that includes the edge labels. Similar algorithms have been proposed by Ponstein,[19] Yau,[20] Kamae,[21] and Ardon and Malik.[22] Mateti and Deo[14] cite related backtrack algorithms by Floyd,[23] Tarjan[24] (later improved by Johnson,[25] Read and Tarjan,[15] and Schwarcfiter and Lauer[26]), Roberts and Flores,[27] Bertziss,[28] Weinblatt,[29] and Ehrenfeucht.[30] The algorithm by Syslo[31,32] is modified by Loizou,[33] who also presents a formal proof of the time complexity of the algorithm. There is also the edge–digraph approach of Cartwright and Gleason.[34] So far as is known, none of these have been applied to chemical graphs and so will not be mentioned further.

**Corey, Wipke, Cramer, and Howe**[35] developed the original ring perception algorithm used in the LHASA organic synthesis system. A detailed description is given in their paper and in the previous publication by Corey and Wipke.[36]

From the connection table the analysis determines which cycles are present, their size, and their relationship to each other, i.e., isolated, spiro, fused, or bridged. Each cycle is represented as a cycle vector and is found by a simple spanning tree depth-first path trace.

Processing starts at the first atom in the connection table and randomly traces paths through the connection table with arbitrary choices at atoms with a ring connectivity of more than two. Such atoms are labeled **choice points** to which the processing returns on backtracking. If the next atom is already in the path, then a cycle has been found. This cycle is stored in an ordered list, and the path is shortened to the last choice point. When all paths from the start atom have been covered, a check is made to see whether all vertices are included in the ring set obtained. If not, the structure has more than one component and path tracing is repeated by using an unvisited vertex as start atom. An illustrative example of this path tracing is given in the Corey et al. paper.

The set of all cycles in the structure is then divided into those regarded as **real rings** and **pseudorings**. The set of real rings, $S$, must include all cycles that can be included in a **maximum proper covering set**, defined as follows: (1) $S$ contains all ring bonds present in the connection table (a covering set); removal of any ring from $S$ leaves a noncovering set. (2) $S$ is chosen so that the sizes of individual rings are as small as possible. This is achieved through the requirement that the intersection of any two rings on $S$ should not involve more than half the edges of either (proper set). (3) $S$ contains the maximum number of rings present in any set fulfilling the first two criteria (maximal set); i.e., $S$ contains exactly the nullity number of rings, except where there is a set of equal-sized rings all of which fulfill the first two criteria, and hence all should be included. All other rings are regarded as pseudorings.

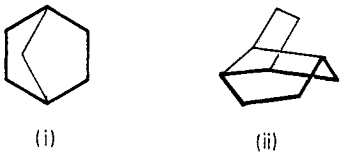

**Figure 1.**

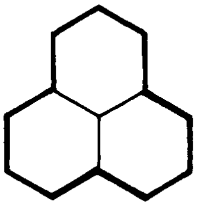

**Figure 2.**

From the examples given by Corey et al. it can be seen that the concept of real rings does have its limitations. For instance, in Figure 1i the six-edged cycle is one of the "key synthetic units" of the norbornane system due to its importance in Diels–Alder reactions; however, it is classified as a pseudoring.

In Figure 1ii, the highlighted six-edged cycle is arbitrarily classed as a pseudoring, while the other six-edged cycle is classed as real (criterion 1 precludes them from both being included as real).

The authors make the observation that few synthetic reactions are known in which key synthetic units are larger than six-edged cycles. Those that are are generally classified as real anyway. As a result, it was suggested that the set of real rings be supplemented by all rings of size six or smaller. Corey and Petersson[9] subsequently developed an alternative strategy, based on sets of fundamental cycles, that can generate such an extended set directly, without first finding all cycles. This algorithm is reviewed later and has also been used in LHASA.

**Shelley**[37] has developed a procedure for ring identification as a step in the production of the coordinates necessary to display chemical structures graphically. The overall program was developed in response to the need to maintain the same orientation for similar structures and to ensure that this orientation corresponds to what a chemist regards as the proper one.

The ring perception module is used to assign the relative coordinates of the atoms in each ring system. Unlike many other applications, one of the requirements is to determine the connectivity of each atom.

The procedure is simple and straightforward, comprising two basic steps: (1) Assign a spanning tree by breadth-first growth from the connection table, making a separate note of the chords. (2) For each chord perform a depth-first search of the tree for all paths between the two vertices of the chord and then add that chord to the tree. This will produce an initial set of all cycles. The set is scanned, and all cycles that contain all the vertices of a smaller cycle are removed; i.e., all cycles containing Nachbarpunkte are deleted to give the set of all simple cycles.

The complexity, $C$, of the system is calculated from

$$C = S - E + V - 1$$

All the simple cycles of simple ring systems ($C = 0$) and bicyclic ring systems ($C = 1$), such as norbornane, are used to generate coordinates. More complex systems ($C \geq 1$) are simplified by including only those simple cycles present in the simplest subring systems. These are determined by a process of successively "peeling" cycles from the ring system. For instance, in Figure 2 the highlighted 12-edged simple cycle is removed.

There are cases in which a ring system cannot be simplified in these terms, and the program is thus unable to display the structure. From the account given it appears that an attempt is being made to choose those rings that are included in the set of $\mathcal{K}$-rings defined by Plotkin (see later). Unfortunately, this attempt is not always successful.

**Balaban, Filip, and Balaban**[38] simplify the structure representation to its basic graph form (which they call a "homeomorphically reduced graph") before any ring perception is performed. The **basic graph** is constructed by ignoring vertices with ring connectivity less than 3 (**reducible vertices**). The reducible vertices are stored as paths between the relevant remaining irreducible vertices of the basic graph. The problem thus becomes one of finding all the cycles present in the potentially much simpler basic graph (although there is the problem of monocycles that have no irreducible vertices). The reducible vertex paths are then slotted back to produce the complete cycles.

Before the basic graph is processed, the number of iterations can be reduced by visually determining the minimum number, $k_{\min}$, of irreducible vertices in its smallest cycle (the default is 1). The program generates all basic-graph cycles with $k \geq k_{\min}$.

All possible combinations of $k$ edges of the basic-graph adjacency matrix are generated, and each is tested to see whether it is a cycle or union of disjoint cycles (each vertex appears twice). To eliminate unions of disjoint cycles, the path is backtracked to see whether the starting point is reached before $k$ steps. If so, then the cycle is rejected.

The idea of using irreducible vertices and of linking all combinations of intervening paths is directly comparable with the much earlier fundamental ring concept of Fugmann, Dölling, and Nickelsen (see later). Consequently, this method suffers from many of the same disadvantages, such as the factorial increase in processing time as the number of paths to be combined increases, i.e., as the graph complexity increases. Unlike Fugmann's method, no attempt is made to select a particular set of simple cycles, thus making the process somewhat simpler.

Overall, although the concept of using basic topology to simplify molecular topology processing is a useful one, several implementation improvements are necessary, such as the automatic separate processing of components and the removal of visually derived heuristics.

**Fujita**[39,40] details an algorithm to find the essential set of essential rings (ESER) as a representation for organic reactions and synthesis design. The resultant ring set bears some similarity to the extended set of smallest rings (ESSR) developed as a result of this appraisal,[2] but the two are based on entirely different concepts.

Fujita's algorithm starts by finding all cycles. It is suggested that this can be achieved by using Wipke and Dyott's method (see later) or Kudo's method.[41] The cycles are sorted by size, and the list is analyzed according to criteria based on ideas of synthetic importance.

Cycles are classified as **essential** and **nonessential**. Nonessential cycles are **tied**, **multi-tied**, or **dependent**. A tied cycle is one that contains a pair of Nachbarpunkte, while a multi-tied cycle contains more than one pair of Nachbarpunkte. Hence, from the set of all cycles, only the simple cycles are considered further. A dependent cycle is defined partly in terms of synthetic importance. Cycles are regarded as containing three classes of non-hydrogen atom: carbon, heteroatom (N, O, S, P), and abnormal (all other atoms). A cycle is dependent if all its bonds occur in a subset of the tied cycles, and

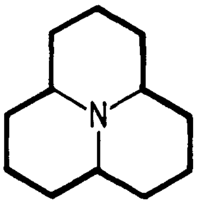

**Figure 3.**

- All of this subset are the same size or smaller than the dependent cycle.
- The dependent cycle contains less than half the bonds in the subset.
- All cycles in the subset are of the same class as the dependent cycle.
- All cycles in the subset have the same or smaller numbers of heteroatoms and/or abnormal atoms as the dependent cycle.

The ESER contains all cycles that are not classified as nonessential; i.e., it is a subset of simple cycles that does not contain any dependent cycles. One consequence of this is that the presence of a heteroatom or abnormal atom alters the status of a cycle. For instance, the 12-edged simple cycle in Figure 2 (outlined in bold) is a dependent cycle, and thus not in the ESER, whereas the same cycle in Figure 3 is not a dependent cycle, due to the central heteroatom, and thus is included in the ESER.

The ESER is used by Fujita to represent the ring systems present in the imaginary transition structures[39] used to depict reaction-site changes during organic reactions. The ESER concept evidently draws upon the much earlier work by Fugmann et al.[46] and Nickelsen (see below) and has similar problems with respect to processing overheads and visual interpretation.

## ALGORITHMS THAT INITIALLY PERCEIVE A SET OF SIMPLE CYCLES

Only one of the reviewed algorithms falls into this category.

**Nickelsen**[42] developed the so-called $\beta$-ring concept in response to the requirement for a mathematically based ring perception technique for manual and automated use in the GREMAS system. It draws upon the earlier fundamental ring concept[46] (see later) by classifying a $\beta$-ring first as a simple cycle, but the processing overheads are less. The literature available all assume manual application, and so no algorithm description for computerized implementation is publicly available. As a result it is not known how the set of simple cycles is generated. All that is available is the $\beta$-ring definition.

Nickelsen gives four definitive propositions: (1) The smallest simple cycles in a graph are $\beta$-rings. (2) If a graph has at most two smallest simple cycles of size $S$, then all simple cycles in the graph of size $S + 1$ are also $\beta$-rings. (3) If none of the edges of a simple cycle, $C$, is common to any of the other cycles in the graph, then $C$ is a $\beta$-ring (i.e., an isolated or spiro ring). (4) If just one edge of a simple cycle, $C$, is common to any of the other cycles in the graph, then $C$ is a $\beta$-ring (i.e., a fused ring). These have been refined into the definition that a $\beta$-ring is a simple cycle that has just three or four vertices or that cannot be constructed from three or more smaller simple cycles, i.e., by the linear combination of their vectors.

All examples given by Nickelsen illustrate the choice of $\beta$-rings using vertex-labeled diagrams. This leads to great confusion since it is the edge vectors that need to be combined

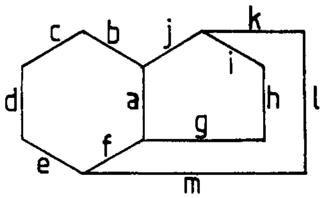

**Figure 4.**

rather than the vertex vectors. In the following example the original vertex numbering has been replaced by edge labeling, and the associated simple-cycle matrix constructed. The ring sum of triplets of cycle vectors is then sufficient to differentiate the $\beta$-rings.

For the structure in Figure 4 the simple-cycle matrix is

| Simple-cycles | Cycle-vectors | | | | | | | | | | | | | Ring |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | a | b | c | d | e | f | g | h | i | j | k | l | m | size |
| *C*1 (a,b,c,d,e,f) | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| *C*2 (a,g,h,i,j) | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5 |
| *C*3 (a,j,k,l,m,f) | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 6 |
| *C*4 (f,g,h,i,k,l,m) | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 7 |
| *C*5 (b,c,d,e,m,l,k,j) | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 8 |

From proposition 1, *C*2 is a $\beta$-ring, and from proposition 2, *C*1 and *C*3 are $\beta$-rings. For *C*4, there are three smaller simple cycles (*C*1, *C*2 and *C*3); the ring sum of these is

$$C1 \oplus C2 \oplus C3 = (1111101110111) \neq C4$$

hence *C*4 is a $\beta$-ring.

For *C*5, the four other simple cycles are all smaller. Of these

$$C1 \oplus C2 \oplus C4 = (0111100001111) = C5$$

hence *C*5 is not a $\beta$-ring.

The main problems with the $\beta$-ring concept are that it is difficult to apply manually, it initially requires the generation of all simple cycles (by an undisclosed method), and it requires a considerable number of ring sum operations for complex ring systems. However, it operates on a firm mathematical basis (the only heuristic being the inclusion of all three- and four-edged simple cycles), and it does not lead to the arbitrary exclusion of rings (as for an SSSR). Unfortunately, it does not always produce a set of rings that conforms to the chemist's fuzzy concept of real rings, and it does occasionally miss faces[2] of structures due to the exclusion of "perimeter" rings.

## ALGORITHMS THAT INITIALLY PERCEIVE A FUNDAMENTAL BASIS SET OF RINGS

The usual technique for finding a fundamental basis involves growing a spanning tree and noting the chords. The basis is then constructed by finding one ring associated with each chord. A spanning tree is derived by manipulation of an adjacency or incidence matrix or by path tracing through a connection table.

The derivation of a fundamental basis has received much attention from mathematicians due to its importance in many areas of graph and network theory. Once a fundamental basis has been established, all other cycles associated with a particular structure can be generated by taking the ring sum of combinations of the basis cycle vectors.

All possible cycles can be derived by a vector–space algorithm such as that proposed by Maxwell and Reed.[43] However, for a system with nullity, $\mu$, there may be up to $2^{\mu} - 1$ rings, requiring an expensive $2^{\mu} - \mu - 1$ possible combinations of vectors. Most vector–space algorithms thus try to limit the number of combinations to the minimum necessary to generate all cycles.

As an alternative to vector combination, a backtracking algorithm can be used to grow trees from one or both ends of each chord until all rings involving each chord have been found. These contrast with the backtrack algorithms mentioned earlier that directly generate the required set without first finding a fundamental basis.

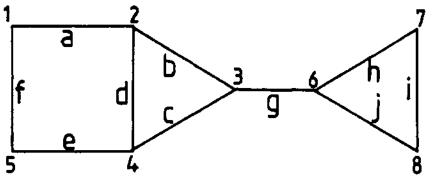

**Figure 5.**

The smallest set of smallest rings (SSSR) is a special case involving the smallest fundamental cycles. Some of the algorithms given in this section do seek to find an SSSR but do not necessarily generate just an SSSR fundamental set. Algorithms that initially generate an SSSR directly, and no other fundamental cycles, are given in the next section.

**Welch**[44] starts with a $V \times E$ incidence matrix, in which the columns are reordered and partitioned into a form to which matrix operations can be applied more easily to obtain a fundamental basis. This matrix manipulation uses a method equivalent to that of Seshu and Reed.[45] The resultant cycle vectors are used as rows in the basis cycle matrix so that subsequent stages can perform standard ring sum operations to derive all cycles.

The rules for manipulation of the matrix and for isolating the cycle vectors are as follows: (1) For each column, *j*, choose a preferably unused row, *i*, with a 1 in the *j*th column. For such an *i*th row, first label the top of the *j*th column with the integer *i* and then replace any other row having a 1 in the *j*th column by its sum, modulo 2, with the *i*th row. (2) Combine each unlabeled edge, corresponding to the *j*th column, say, with edges labeled by integers *k* such that $a_{k,j} = 1$.

The structure in Figure 5 would be manipulated as follows:

incidence matrix

| | a | b | c | d | e | f | g | h | i | j |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 4 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

⟹

label matrix

| | 1 | 2 | 3 | | 5 | | 6 | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | a | b | c | d | e | f | g | h | i | j |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**Rule 2**: the fundamental cycles are (d,b,c), (f,a,b,c,e), and (j,h,i), giving the cycle matrix

| | a | b | c | d | e | f | g | h | i | j |
|---|---|---|---|---|---|---|---|---|---|---|
| *C*1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| *C*2 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| *C*3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| $C1 \oplus C2$ | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| $C1 \oplus C3$ | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| $C2 \oplus C3$ | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |

The ring sum operation, $C1 \oplus C2$ gives another fundamental cycle (a,d,e,f), while the other two ring sum operations give edge disjoint unions of cycles.

As can be seen, the label matrix has one null vector (row 5) produced by the manipulation. Welch shows that this is always true and is a necessary result of the processing.

A fundamental basis is obtained for each component of the structure. All combinations of fundamental cycle vectors within each component could be produced; however, to reduce the number of ring sum operations required to find all cycles, Welch prefers to use two further stages of processing.

Stage 2 orders the cycles such that cycles contained within the same component of the graph will appear consecutively.

Stage 3 progressively combines pairs of cycles. Welch attempts to show that any unions of disjoint cycles produced during stage 3 can be ignored. However, this is an incorrect assumption and can lead to certain cycles being missed. This was originally pointed out by Gibbs, whose corrected algorithm was based on Welch's ideas and is considered later.

**Fugmann, Dölling, and Nickelsen**[46] return to basic considerations of which rings are regarded as really present in a structure by bench chemists, and define a "fundamental ring" set. The terms fundamental ring and fundamental basis should not be confused. A fundamental basis is used as a subset from which the set of fundamental rings is derived.

Use is made of the "condensation points" within a structure, i.e., vertices of the basic graph. Unlike spanning tree and walking techniques, this topological approach combines half-rings or **fundamental paths** by using certain exclusion rules to avoid processing all possible combinations to give all possible cycles rather than the required subset. Thus, all cycles giving path combinations with Doppelpunkte ($\alpha$-paths) and Nachbarpunkte ($\beta$-paths) are excluded by coupling only those paths that will not lead to such occurrences.

The algorithm begins by using the "from attachment list" of Morgan,[47] which is equivalent to a minimum spanning tree, and the associated "ring closure list", which contains the chords. From these a fundamental basis is generated. Any of these containing no more than one condensation point are fundamental rings and are stored. All pairs of condensation points are determined, and the paths between them are traced and stored in a matrix. A matrix of all condensation points is also constructed. Matrix transformation and manipulation find coupled paths, from which a matrix of coupling is produced. To form the remaining fundamental rings, the row vector representing the linkage of a pair of condensation points is added, modulo 2, to each of the other vectors. These fundamental rings are added to the previously stored list of fundamental rings.

This process is somewhat complicated and time-consuming, with the possibility of a great many elementary paths being generated (for instance, 384 paths have to be considered in cubane[10]). The concept was an early one; it was one of the first to be successfully implemented, and it provided the basis for the more powerful $\beta$-ring concept.

Another interesting aspect of this paper is the construction of ring complexes. These are similar to the labeled dual of a graph except that the infinite region[2] is not included. Each fundamental ring is represented as a vertex, while the edges denote incidence between these rings and carry a value equal to the number of condensation points in common.

**Gotlieb and Corneil**[48] present an alternative to Welch's stage 1 incidence matrix manipulation with an algorithm that uses an adjacency matrix to derive a spanning tree and chords. The process of matrix manipulation is slower than Welch's method, but it requires less storage, using three $V \times V$ adjacency matrices rather than the two $V \times E$ incidence matrices plus two $E$-tuple vectors required by Welch.

The spanning tree is constructed by the following matrix operations:

(1) The set of $V$ vertices in adjacency matrix $A$ is partitioned to form a set of disjoint trees in adjacency matrix $B$. For each row $i$ of $A$, locate the first superdiagonal element of the $i$th row of $A$ that contains a 1, and set this element and the corresponding supradiagonal element to 1 in matrix $B$; i.e., set $b_{j,i} = b_{i,j} = 1$ (where $b$ is an element of matrix $B$). If there are further superdiagonal elements of the $i$th row of $A$ that contain 1's, then set them and their corresponding supradiagonal elements to 0 in matrix $B$; i.e., set $b_{j,i} = b_{i,j} = 0$. Unless overwritten by this procedure, all other elements are set to 0. For example, matrices $A$ and $B$ for the graph in Figure 5 are

Matrix $A$ Matrix $B$

$$\begin{array}{c|cccccccc} & 1&2&3&4&5&6&7&8 \\ \hline 1&0&1&0&0&1&0&0&0 \\ 2&1&0&1&1&0&0&0&0 \\ 3&0&1&0&1&0&1&0&0 \\ 4&0&1&1&0&1&0&0&0 \\ 5&1&0&0&1&0&0&0&0 \\ 6&0&0&1&0&0&0&1&1 \\ 7&0&0&0&0&0&1&0&1 \\ 8&0&0&0&0&0&1&1&0 \end{array} \quad \Longrightarrow \quad \begin{array}{c|cccccccc} 1&2&3&4&5&6&7&8& \\ \hline 1&0&1&0&0&0&0&0&0 \\ 2&1&0&1&0&0&0&0&0 \\ 3&0&1&0&1&0&0&0&0 \\ 4&0&0&1&0&1&0&0&0 \\ 5&0&0&0&1&0&0&0&0 \\ 6&0&0&0&0&0&0&1&0 \\ 7&0&0&0&0&0&1&0&1 \\ 8&0&0&0&0&0&0&1&0 \end{array}$$

(2) The $n$ vertices can now be divided into components with respect to matrix $B$. Take a vector $X$ of length $n$ and initialize each element to 0. Take a row $i$ and set $X(i) = 1$. If in row $i$ any element $b_{i,j} = 1$, then set each $X(j) = 1$ also. Set $i = j$ for each $j$, and repeat until paths can no longer be traced. Mark each row $i$ as having been processed. For each component of matrix $B$, these vectors form the rows of a matrix $C$. For the example above this will give

$$X1 = (11111000),\ X2 = (00000111)$$

and

Matrix $C$

$$\begin{array}{c|cccccccc} & 1&2&3&4&5&6&7&8 \\ \hline X1&1&1&1&1&1&0&0&0 \\ X2&0&0&0&0&0&1&1&1 \end{array}$$

As can be seen, matrix $C$ is in block-diagonal form.

(3) The components are amalgamated by adding appropriate edges to $B$. The component with the smallest number of vertices is repeatedly examined. An edge between this component and another is added to matrix $B$. This process will terminate with matrix $B$ representing a spanning tree. For example, in matrix $C$, the second row has the smallest number of vertices. There is only one vertex, and so only one iteration is performed. Vertex 6 can be joined to one of the vertices in row 1 by adding edges (3,6) and (6,3) to matrix $B$:

$$\begin{array}{c|cccccccc} & 1&2&3&4&5&6&7&8 \\ \hline 1&0&1&0&0&0&0&0&0 \\ 2&1&0&1&0&0&0&0&0 \\ 3&0&1&0&1&0&1&0&0 \\ 4&0&0&1&0&1&0&0&0 \\ 5&0&0&0&1&0&0&0&0 \\ 6&0&0&1&0&0&0&1&0 \\ 7&0&0&0&0&0&1&0&1 \\ 6&0&0&0&0&0&0&1&0 \end{array}$$

(4) Each edge of $A$ not in $B$ is a chord. In this example the resultant chords are (1,5), (2,4), and (6,8). These are found and passed on to generate a set of fundamental cycles.

The partitioning of step 2 isolates cyclic components (i.e., separates the cyclic blocks), while the connection of these components in step 3 can be seen to eliminate acyclic portions from chord production.

**Gibbs**[49] prefers to use the adjacency matrix partitioning technique of Gotlieb and Corneil to generate a fundamental basis in place of Welch's stage 1 incidence matrix manipulation. A modified form of Welch's stage 3 is then used on this set to find all cycles. This modified algorithm specifically separates out unions of disjoint cycles and reprocesses them at the next iteration.

The main algorithm is given in six lines of compact graph theoretical notation, to which the reader is referred if interested! In outline, the algorithm seeks progressively to accumulate all cycles into one set, $S$, by using three intermediate sets at each iteration. Set $R\star$ holds the unions of disjoint cycles, set $R$ holds the smallest cycles associated with the chords used in the fundamental set not already in $S$ (after the necessary elimination step is complete), and set $Q$ holds the cycles still to be considered. After each iteration, any cycles in $R$, $R\star$, $Q$, and the original fundamental set are placed in set $Q$. $R$ and $R\star$ are reset to 0, and the process is continued until the number of iterations has reached nullity. The result is the complete set of cycles.

These modifications increase the processing time compared with Welch's original algorithm, but at least all cycles are correctly perceived! Gibbs suggests that efficiency could be improved by applying the algorithm separately to the components highlighted by Welch's stage 2. This idea is taken up by Wipke and Dyott, who go a stage further by limiting processing to the cyclic blocks (see later).

According to Mateti and Deo[14] a similar attempt to obtain all cycles without trying all vector combinations is given by Hsu and Honkanen.[50] This paper is not reviewed here, but Mateti and Deo show examples where all vector combinations are used and propose an alternative algorithm. However, according to Syslo[31]

> "The algorithm proposed by Mateti and Deo for enumerating all cycles of a graph utilising a cycle graph [matrix] depends on generating connected induced subgraphs of a cycle graph and then testing whether the corresponding elements of the cycle space are cycles. In general, this approach is inefficient."

Syslo then presents an efficient cycle vector-space algorithm for use on planar graphs that is claimed to be as efficient as the backtrack algorithms.

**Paton**[51] uses a depth-first path trace through an adjacency matrix to construct a spanning tree. This is achieved by using two such matrices plus a pushdown list of vertices in the spanning tree that have not yet been examined. The original adjacency matrix, $A$, is gradually destroyed during processing while the new matrix, $B$, is gradually filled with the spanning tree. The main claim of this method is that its storage requirements are as low as Gotlieb and Corneil's algorithm, while the speed of execution rivals that of Welch's algorithm.

The adjacency matrix $A$ has a set of $E$ edges and $V$ vertices. Initially the set, $T$, of vertices already in the spanning tree is empty while the set, $X$, of vertices still to be examined equals $V$. Generation of the spanning tree is started by arbitrary selection of a vertex in $X$ to become the root of the tree; this vertex is added to the matrix $B$ and to the top of the pushdown list. The algorithm then iteratively processes all vertices in the pushdown list in the following manner: (1) Take the last vertex added to the list and successively locate each edge associated with it in matrix $A$. If the list is empty, then stop; the spanning tree is complete, and a set of fundamental cycles have been found. (2) If such an edge is already in $T$, then it is a chord; backtrace the fundamental cycle in matrix $B$. Otherwise, add the edge to matrix $B$ and associated vertex to set $T$. In either case remove the edge from $X$. (3) When all edges associated with this particular vertex have been examined, remove the vertex from $X$ and the matrix $A$. Return to step 1. It is shown that taking the last element added to the list as the next vertex to be examined is simpler and faster than taking the first element; i.e., the depth-first approach to growing the spanning tree is more efficient than the breadth-first approach.

A modification to Paton's algorithm is given by Jovanovich,[52] which makes it even more efficient by reducing the length of the working vector to less than the number of vertices in the graph. A similar matrix manipulation procedure that is more efficient than Paton's original algorithm is given by Ito and Kizawa.[53] In this approach the $V \times V$ adjacency matrix $A$ is complemented by a linear list of length $V$ that contains each row index of $A$. Thus, all accesses to $A$ are made through the list, which is first sorted into the most efficient order for processing by collecting adjacent vertices together. It is shown that growing a spanning tree using the list is more efficient by breadth-first than by depth-first tracing, in contrast to Paton's method. Ito and Kizawa then use a simple backtrack procedure to trace a fundamental cycle from each chord.

**Corey and Petersson**,[9] as mentioned earlier, have developed another algorithm for use in the LHASA system. This uses a fundamental basis to derive an extended set containing all "synthetically important" rings.

Paton's algorithm is used to grow a spanning tree. Instead of simply producing an adjacency matrix, each vertex in the tree is stored along with the set of edges and vertices above it in the tree. When a vertex already in the tree is found, a fundamental basis ring is obtained by taking the ring sum of the appropriate pair of bond sets. This eliminates the backtracking of Paton's algorithm but requires additional storage.

From the set of fundamental basis rings found, the corresponding reduced basis is derived in an attempt to find an SSSR (referred to as a "minimum spanning set").

The reduced basis is found by taking the ring sums of each pair of fundamental basis cycles vectors ($R$) and retaining the smallest two from the resultant triplet; i.e., for each $R_i$,$R_j$ the ring sum $R_i \oplus R_j$ is taken, where $i > j$ and $j$ varies from 1 to $\mu - 1$. The smallest pairs of rings from $R - i$,$R_j$, and $R_i \oplus R_j$ are retained as reduced basis rings. The problem of arbitrary choice between equally sized cycles or the exclusion of synthetically important rings is tackled by supplementing the reduced basis with further cycles, if necessary, so that it contains at least an SSSR. To do this, trees are grown from both ends of each chord until either a new cycle or the original fundamental cycle is found or the size of the largest reduced basis cycle has been exceeded. The smallest of any new cycles found can then be added to the reduced basis.

The SSSR is formed by selection of the smallest rings from the reduced basis. The resultant set is then checked to ensure that all edges of the graph are included; if not, then further smallest rings containing those edges are added. Their description does not make it clear what circumstance might lead to such an omission of edges.

As a final check, a set $C$ of all chords whose fundamental basis rings are not in the SSSR is created. If an SSSR ring contains just one of these chords, then that chord is removed from $C$. All linear combinations are formed from the rings containing the remaining chords in $C$. Once again, if any contain just one of the chords in $C$, then that chord is removed from $C$. If $C$ is not now empty, then all smallest rings are found from each of the remaining chords (by a process not mentioned), and if they are not already in the set of combinations, they are added to the SSSR.

Since the original fundamental basis contains all edges, subsequent processing should not eliminate these edges without raising doubts about the efficiency and reliability of the method.

**Wipke and Dyott**[12] describe a "Welch-assembly-Gibbs" algorithm, an assembly being a block of a graph. An efficiently implemented form of Welch's stage 1 is applied to derive a fundamental set of cycles. The cyclic blocks are found by grouping those fundamental cycles with edges in common. This is achieved by employing rapid and simple bit-vector matching to test for intersection. Each cyclic block is processed by a form of Gibbs' algorithm that finds all cycles within that assembly by means of the ring sum of all combinations of the associated cycle vectors.

The Gibbs algorithm has been modified for efficiency to require only one array and one "superset". This eliminates the overheads incurred through the constant changing between the four sets used in Gibbs' original algorithm.

Gibbs was certainly correct is assuming that efficiency would improve by limiting the application of the algorithm to separate components or blocks. The use of blocks gives increasingly dramatic improvements as the complexity of the structures increases. The processing time decreases by several orders of magnitude for the most complex structures tested.

After all cycles have been found for a block, an attempt is made to isolate the reduced basis set of cycles, as in Corey and Petersson's algorithm. This is achieved by taking the ring sum of pairs of fundamental cycles already found and retaining the two smallest of each triplet in a set. This is repeated with all possible pairings until no further changes can be made to the set. It is stated that

> "A reduced basis is an attempt, not always a successful one, to find a basis set of rings which consists of the smallest rings that can form a basis set, sometimes called a 'minimum covering set'."

Hence, as with Corey and Petersson, an SSSR is being sought.

In addition to the reduced basis, any other "chemically interesting" cycles (i.e., those with up to eight edges) are added from the set of all cycles found for each block. The definition of chemically interesting is application dependent and can be altered to choose any required set from the sets of all cycles.

One of the closing comments is that

> "It would seem to be very inefficient to generate all rings and then select the desired rings from them, as opposed to only generating the desired rings, but owing to the speed of the set operations used this is not the case. The algorithm presented here is faster than any previously published algorithm for finding rings in chemical structures."

This comment evidently impresses Gray,[8] who devotes much of his review to discussing Wipke and Dyott's method without mentioning other notable algorithms and in spite of the difficulties in comparing efficiencies from data given with the various publications.

More recently, Matyska[54] has applied Wipke and Dyott's algorithm to a derivative of Balaban et al.'s homeomorphically reduced graph (see above). The resultant processing times are an improvement on both this and the HRG algorithm.

**Gasteiger and Jochum**[10] aim to find an SSSR from a spanning tree and associated fundamental set of cycles. They also give a concise and convincing argument for the use of an SSSR instead of finding all cycles or using heuristics and an interesting account of the nature of a ring set appropriate for synthesis planning.

The algorithm is implemented in two stages. For stage 1, from an arbitrary starting atom, a spanning tree is grown in a breadth-first manner. The chords are stored and ordered according to their distance away from the root. From either end of each chord backtracking toward the root looking for the first common atom between the two paths ensures that the closest linking path is found. The fundamental set found is thus usually an SSSR. To cater for circumstances in which it is not, stage 2 is performed. The complexity, $C$, of the ring system is determined from

$$C = \frac{\text{sum of the no. of ring vertices in the system}}{\text{sum of the no. of vertices in rings found so far}}$$

If $C \leq 1.5$, then the processing is terminated; otherwise, another root is chosen by using the following criteria: (1) It must be a member of a smallest cycle that does not contain any previous roots. If this is not possible, then arbitrarily choose one. (2) It must be the vertex with the greatest ring connectivity for the cycle from which it was chosen. The basic algorithm is then repeated for the new root, up to a maximum of three iterations. When all iterations are completed, an SSSR is selected from the accumulated ring set.

The method is an attempt to derive an SSSR by direct production of a fundamental set containing as many smallest cycles as possible, in contrast to the reduced basis technique of Corey and Petersson and Wipke and Dyott. If this is not successful, then other fundamental sets are generated in the hope that any missing smallest cycles will be included in these other sets. The criteria used ensure that the new fundamental sets have the greatest chance of differing from previously derived sets, but the process does not unequivocally guarantee success. Furthermore, it is not explained why a complexity of 1.5 or more should indicate that a structure will have been adequately processed by stage 1.

**Sorkau**[7] presents a brief review paper plus a suggestion for extension of Gasteiger and Jochum's ideas that gets around the problem of failures. The review is useful in that algorithms are categorized by whether they use backtracking (path tracing) or graph theory (spanning trees) rather than by the set of rings initially produced. However, no mention is made of the SSSR algorithms by Roos-Kozel and Jorgensen and Hendrickson et al. or the GREMAS approach of Fugmann et al. and Nickelsen.

Sorkau concentrates on the perception of an SSSR and dismisses the algorithms of Plotkin and Zamora since they fail for certain complicated ring systems. Gasteiger and Jochum's algorithm has similar failings, but certain improvements are suggested. First, the graph is reduced to its basic topological form by contracting vertices with a ring connectivity of less than 3. This basic graph provides the start atoms for growing trees from the original graph. Simultaneous with the tree building, the individual vertices of the tree are relabeled so that the shortest path between the start atom and neighboring vertices is always found (a method presumably obtained from Weise[55]). Once the spanning tree is complete, all branches from a chord are traced back, breadth-first, until the first common atom is encountered. The cycle, or cycles, closed at this point are "guaranteed" to be the smallest from that chord. This process is repeated with the remaining start atoms. Finally, the resultant list of cycles is sorted by size, and the first $\mu$ linearly independent smallest cycles form the SSSR.

This improved algorithm gets around the problems of selectively producing fundamental sets, but no mention is made of the possibility of there being several SSSRs. Sorkau states that the algorithm works on the two structures for which Zamora's algorithm is known to fail and therefore is able to process relatively complex structures successfully.

## ALGORITHMS THAT FIND A SMALLEST SET OF SMALLEST RINGS DIRECTLY

In most algorithms the amount of work is a function of the set of rings found. An SSSR is a particularly useful set of rings because it is a basis of known cardinality and comprises the smallest fundamental cycles. As a fundamental basis it is therefore the most useful for consistent characterization of a structure, and as a smallest basis it should make processing more rapid, especially if the set is then extended to include other cycles. However, in certain complex structures, with many heteroatoms or with many symmetrically equivalent rings, the consistency can break down since there may not be a unique SSSR. Nevertheless, the SSSR is regarded as adequate for most applications or can at least be used as an initial set on which to build a deeper ring analysis.

**Plotkin**[56] gives one of the first accounts of the theory behind the SSSR principle, to which the interested reader is referred. The algorithm outlined has been incorporated into the Chemical Information and Data System (CIDS) used by the U.S. Army. Its aim is to find all cycles with fewer than nine atoms or which are classed as $\mathcal{H}$-rings.. $\mathcal{H}$ is the set of all possible SSSR rings and so gets around the problem of arbitrary exclusions when an individual SSSR is considered. The algorithm starts by finding an SSSR and then extending the search.

The initial SSSR is found by pruning the graph of all acyclic side chains. Ring perception is then based on Plotkin's theorem:

> If $P$ is an unforked path in a structure $G$ and there is a shortest ring $R$ through $P$ such that $|R| \leq 2|P|$, there is an SSSR of $G$ that contains $R$ and no other ring in which $P$ occurs.

Each edge of the structure is tested for a suitable unforked path $P$ and ring $R$. The longest unforked path is determined, and paths are grown from one end. If the other end is not reached before the length equals $P$, then $|R| > 2|P|$. Once one or more suitable cycles are found, one is stored and the path $P$ deleted from $G$. After the number of iterations has reached the nullity, no rings are left in $G$ and an SSSR has been found. If the structure has no suitable $P$, then an edge is chosen to be temporarily deleted. This edge is chosen to lie within the largest smallest cycle already found. If no cycles have been found, an edge is chosen at random. Finding the SSSR is repeated with this edge removed.

To find all $\mathcal{H}$-rings the following theorem is applied:

> Given an SSSR, if $R_m$ is the longest ring of the SSSR and $P$ is an unforked path that is part of $R_m$ but of no other ring of the SSSR, then the rings of class $\mathcal{H}$ that pass through $P$ are the rings of length $|R_m|$ that pass through $P$. These are the shortest rings through $P$.

Hence a longest unforked path $P$ (it may be a single edge) is found for the largest cycle in the SSSR. By the previous procedure all cycles of length $|R_m|$ through $P$ are added to $\mathcal{H}$. $P$ is deleted from $G$, and $R_m$ is deleted from the SSSR. After the nullity number of iterations, no cycles are left.

Application of the two theorems is known to fail in certain examples where an edge is chosen for deletion but which still does not enable an SSSR to be found.

**Bersohn**[57] generates those cycles conforming to Corey and Petersson's set of synthetically important rings, i.e., those cycles that have fewer than seven atoms or are not an envelope to other cycles.

If the nullity is greater than 0 (indicating the presence of rings), the acyclic side chains are pruned from the connection table and ring perception proceeds as follows:

(1) If $\mu = 1$, then the pruned structure is an isolated ring and processing can stop; otherwise, a start atom is arbitrarily chosen.

(2) All paths of length $k$ are traced from the start atom (initially $k = 3$). If the path returns to the start atom, the cycle is checked and, if not already found, stored. If the cycle has been found previously and has a path length of more than six, then all paths starting with the new path sequence are deleted. If the path is of length less than 6, then these paths are deleted only when they exceed a length of 6. Application of this limit ensures that once a smallest ring has been found only symmetrically equivalent rings or rings up to six atoms are traced from that start atom, thus eliminating larger envelope rings. Path tracing continues until all possible paths have been eliminated by success (a new cycle) or failure (a previously discovered cycle is found, ring closure occurs at a vertex other than the start atom, or the path is eliminated to prevent envelope tracing). If a cycle is not found, $k$ is incremented by one and step 2 is repeated; i.e., this is a depth-first trace from each start atom. Step 2 terminates once a cycle is found.

(3) A new start atom is selected and step 2 repeated.

Iteration continues until no more start atoms are left or the nullity number of rings has been found.

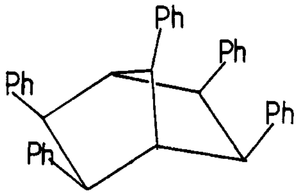

**Figure 6.**

This algorithm is incorporated within an unnamed organic synthesis program. No mention is made of the performance, and no indication is given as to whether it processes complex ring systems consistently.

**Esack**[58] presents an improvement to Bersohn's algorithm for certain classes of structures. It imposes restrictions on the choice of start atom and the choice of which atom the path tracing continues along next.

Taking Bersohn's pruned structure, if the nullity is greater than 1, then all vertices with ring connectivities of more than 2 ("multiconnected") are listed as choices for start atoms; otherwise, the pruned structure is an isolated ring. If there is a vertex such that three or more of its neighbors have a connectivity of 3 or more, then Bersohn's original algorithm is used. Such an occurrence is characteristic of structures that this algorithm cannot successfully process due to their complexity.

For instance, Esack cites the norbornane derivative in Figure 6 as too complex due to the high connectivity of the vertices. Notice that the pruning will remove side chains but not the inter-ring system edges connecting the phenyl groups to the norbornane; i.e., it does not isolate the cyclic blocks and will include the interblock edges in the ring-connectivity calculation.

If the structure can be processed, then a start atom is taken from the list of multiconnected vertices. Tracing paths of length $k$ starts along edges attached to doubly connected vertices not in a previously discovered ring. This ensures each ring is only found once. If all edges lead to vertices in known rings, then the next start atom is chosen.

Once all start atoms have been tried, then the resultant rings are examined for the presence of overlapping rings. If all edges occurring in more than one ring have multiconnected vertices at both ends, then the algorithm terminates, otherwise Bersohn's original algorithm has to be implemented. For instance, in norbornane these steps will have found only the two five-edged rings and not the synthetically important six-edged ring. The original algorithm has to be used to find the missing ring.

The level of complexity at which this algorithm fails and has to call Bersohn's original algorithm is very low. It is recognized that spiro-fused and bridged systems must still go through Bersohn's original algorithm. Esack gives the example in Figure 2 as one that could be processed correctly, but instead the ring-connectivity criteria used lead to Bersohn's algorithm being called. These are severe limitations and as such give little improvement on the original. The situation might be better if the blocks were isolated and processed separately. The algorithm does attempt to order the processing rather like the selection of highest connectivity atoms in Zamora's algorithm (see below), but is not nearly so generally applicable or effective.

**Zamora**[13] starts from basic principles and returns to the simplest of ideas, namely, that of choosing a start atom and path tracing from that atom until the desired ring is found. Although a depth-first trace is used in the original algorithm, Zamora makes it clear that a breadth-first trace would be equally amenable to programming and would ensure finding the smallest cycle, or cycles, first.

Zamora's algorithm uses several criteria to control the path tracing to make it more efficient and for performing elemental ring analysis during processing to enable selection between

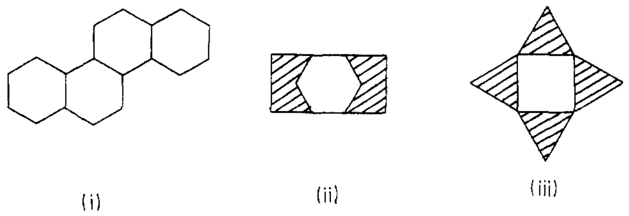

**Figure 7.**

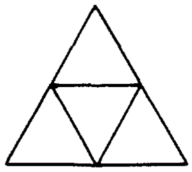

**Figure 8.**

symmetrically equivalent rings.

The basic principle on which the algorithm works is that there occur three classes of ring system with respect to their SSSR:

**Type I.** No subset of the smallest cycles contains all vertices of the ring system, as shown by Figure 7i.

**Type II.** All vertices but not all edges are contained by a subset of the smallest cycles, as in Figure 7ii.

**Type III.** All vertices and all edges are contained by a subset of the smallest cycles, as in Figure 7iii.

This categorization is reflected in the three phases of Zamora's algorithm:

**Phase 1, To Include All Vertices in the Ring Set.** Initialize all **atom-used** and **bond-used** values associated with the vertices and edges to 0, to denote them as unused. Starting from an unused vertex of highest ring connectivity, the **start atom**, trace the smallest ring associated with that vertex. This is made more efficient by setting an upper limit to the length of the path trace equal to the size of the smallest ring found so far for that vertex (initially set to the number of vertices in the whole structure). For all vertices and edges in the smallest ring increment their atom-used and bond-used values, respectively, and store the ring. If there is a choice between smallest rings, then apply heuristics based on the numbers of used vertices, used edges, heteroatoms, and the connectivity sum to select just one of them. Continue until all vertices have nonzero atom-used values. Check the number of cycles found; if it equals the nullity, then terminate; otherwise, continue.

**Phase 2, To Include All Edges in the Ring Set.** If there are any unused edges left (i.e., with bond-used values of 0), then similarly trace the smallest ring associated with each of these edges and terminate if the nullity is reached.

**Phase 3, To Include All Faces in the Ring Set.** If there are any unfound faces, then trace these faces by following paths with all bond-used values equal to 1 (the bond-used limit), or at most one bond-used value of more than 1, and with all vertices with a ring connectivity greater than 2.

The faces traced in phase 3 are restrictively defined in terms of their ring connectivity and bond-used values. These refer only to limited cases; the narrow definition of "face" does not provide a general solution and leads to several failures, of which Zamora gives two examples.

In a discussion of cycle bases, Cribb, Ringeisen, and Shier[59] present a theorem that the innermost cycle basis (i.e., the finite regions) of the structure in Figure 8 cannot be derived from any spanning tree. This derives from work by people such as Syslo and Deo et al. (see later) and neatly explains those situations in which it is necessary for Zamora's algorithm to go to phase 3. Unfortunately, although the authors have considered many theoretical aspects to help develop an algorithm to find the minimum cycle basis (i.e., an SSSR), they seem unaware of the wealth of research that already exists in this direction, in particular with respect to many of the algorithms in this review.

**Schmidt and Fleischhauer**[60] find rings from an adjacency matrix by an unusual first stage that uses algorithms designed for oriented graphs to detect cyclic vertices. The algorithm is designed to give better performance compared with Bersohn's algorithm by subjecting only small parts of the graph to costly path tracing.

The first stage is to construct an oriented adjacency matrix from the original graph, as detailed in their paper. From this oriented matrix, Warshall's algorithm[61] can be used to produce the corresponding path matrix. The diagonals of this matrix give those vertices that are part of a cycle and their connectivities.

For the second stage, vertices with a ring connectivity of more than two are stored in a 5 × $V$ matrix along with their neighbors. A start node is selected from this matrix, and one of its edges is cut. The shortest paths are then traced by growing a tree from this start node to find the smallest ring. If there is more than one shortest path, then one is arbitrarily chosen. Nodes present in the smallest ring are set to zero in the matrix. This is continued until all entries of the matrix are zero, except for the first row.

The algorithm is known to fail for embedded rings surrounded by smaller rings. For instance, the nullity for the ring system in Figure 7iii is five, but only the four three-edged rings will be traced.

**Roos-Kozel and Jorgensen**[11] utilize the fundamental atom and bond sets derived from the connection table used in the CAMEO organic reaction system, as previously described by Salatin and Jorgensen.[62] The aim is to generate an SSSR and any symmetrically equivalent rings (i.e., Plotkin's set of $\mathcal{K}$-rings).

As with many of the fundamental basis algorithms already described, tracing of cycles occurs during spanning-tree production. If one spanning tree does not yield the nullity number of rings, then another tree is grown; this situation may arise due to the restrictions imposed on the path tracing. The algorithm has three stages: (1) All acyclic side chains are pruned. (2) The vertex of highest connectivity is selected as start atom, unless it is already in two or more known cycles and other unused vertices are available. Iterations are continued until the nullity number of rings has been reached and all atoms are included in the ring set or until there are no more suitable start atoms left. (3) From each start atom a tree is grown, breadth-first, until a cycle is formed or the path branches. Tracing stops at this point to avoid finding an envelope ring.

Any cycles found are checked for duplication and stored if new. Stopping at a branch point has the unfortunate consequence of ensuring that embedded rings cannot be found (i.e., Zamora's type III rings). To overcome this, if the number of paths from the start atom, divided by 2 and incremented by 1, is greater than the number of cycles found from that start atom, then an embedded ring is sought!

If all the start atoms have been used but the nullity number of rings has not been reached and there are still vertices not present in the ring set, then separate procedures are called to check for asteranes, cyclophanes, or porphyrins (which are the classes known to fail at this point). These procedures rely on the observation that if two cycles have been found and four start atoms are present, then it is a potential cyclophane, while more than four start atoms indicates an asterane and more than two cycles indicates a porphyrin.

It is immediately obvious that stopping path tracing at stage 3, when a branch is encountered, leads to a host of problems that require several additional subprograms to solve. It is not evident whether all possible circumstances are catered for even after all this laborious additional processing. It would seem more sensible to extend the capabilities of the original path tracing rather than rely on further levels of processing.

**Deo, Prabhu, and Krishnamoorthy**[63] show that the problem of finding the "minimum-length fundamental cycle" set (i.e., an SSSR) as efficiently as possible is the same as finding the associated shortest total path length spanning tree. They then prove that to find such a spanning tree is NP-complete, and so it is necessary to use heuristics to achieve a sub-optimal solution.

This proof concludes a long line of research started originally by Stepanec[64] and Zykov.[65] The research led by Syslo over many years[66–69] covered much of the theory involved, showed the limitations to the Stepanec–Zykov algorithm, and led to the proposal of the Hubicka–Syslo algorithm. Kolasinska[70] has described both these algorithms in terms of matroids and by so doing has revealed problems in the Hubicka–Syslo approach. The paper by Kolasinska gradually comes to the conclusion that only a suboptimal solution is feasible.

Deo et al. start with a proof of this assertion before moving on to consider the efficiency of generating the spanning tree from which an SSSR is generated, a matter not considered elsewhere. In common with many other NP-complete graph theoretic algorithms, the approach to generation of the spanning tree can be classified as depth-first, breadth-first, or mixed search (see Tarjan[71] and Deo[72]). They state that the depth-first method, such as that of Tarjan (and the approach used in many algorithms in this review), will inevitably produce a spanning tree that generates very long fundamental cycles. Paton's mixed-search method is reckoned to be better on average, but can still produce unacceptable worst cases. Since the breadth-first approach usually produces short-diameter spanning trees, it is more likely to produce the smallest fundamental cycles. Deo et al. suggest and test four different heuristics to make the breadth-first search more efficient. The four heuristic approaches are the static degree sort, the dynamic degree selection, the unexplored edges, and the multipoint breadth-first search (the term degree is equivalent to the term connectivity used in this review).

In the static degree sort, the adjacency matrix is reordered in descending order of the connectivities of the corresponding vertices. The spanning tree is then generated in the normal breadth-first fashion, starting with vertex 1, the highest connectivity vertex. For the next vertex, the highest connectivity vertex from the successors of the oldest vertex is chosen, which is not necessarily the highest connectivity vertex in the partial tree.

The dynamic degree selection method differs in that the next vertex is taken from the highest connectivity vertex already in the partial tree.

For the unexplored edges approach, as each vertex is chosen, its connectivity is decremented by 1 so that when the next vertex is chosen, it is taken from the vertex of highest unexplored edge connectivity in the partial tree.

The opposite approach to these three is the multipoint breadth-first search in which exploration proceeds from the vertex of highest connectivity irrespective of whether it is already in the partial tree. This produces a forest of subtrees. If both vertices associated with a new edge belong to the same tree, then that edge is a chord; otherwise, the edge is a tree edge, and the two subtrees are merged. The edges are chosen with those with the highest sum of the incident vertex connectivities first.

The performance of each of these approaches is analyzed in detail and the conclusion reached that the static degree search and multipoint breadth-first search are the best to use.

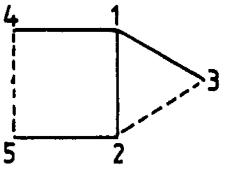

**Figure 9.**

In contrast to such attempts to generate a spanning tree with a short fundamental set of cycles, Horton[73] has recently proposed the first polynomial-time algorithm for generating the shortest cycle basis for two-connected graphs. Unfortunately, on average it is much slower than the other algorithms mentioned in this category. In conclusion, Horton suggests that further research is necessary to integrate the various approaches to obtain greater efficiency.

**Hendrickson, Grier, and Toczko**[74] use a method based on work from a previous paper by Hendrickson and Toczko which showed that any graph can be uniquely numbered by creating its maximal adjacency matrix.[75] In the later paper reviewed here they state

> "This maximal matrix is created by assigning numbers to the atoms, or graph points, in such a way that each row in the matrix, considered as a binary number, must be the maximum possible number."

The result is a symmetric $V \times V$ matrix, usually with the 1's as far up to the left as possible, leaving the upper right and bottom left areas as 0's. This not only enables the representation to be stored efficiently but also allows an SSSR to be found easily and rapidly.

The maximal matrix corresponds to the maximal spanning tree for the structure, plus its chords if it is cyclic. The connectivity for each vertex of the spanning tree and for each chord is given in the so-called $T$ and $R$ lists. From these lists the complete matrix can be reconstructed. These lists are therefore a very compact structure representation. The structure in Figure 9, for instance, gives the matrix

| | 1 | 2 | 3 | 4 | 5 | T | R |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 | 0 | 3 | 0 |
| 2 | 1 | 0 | c | 0 | 1 | 2 | 1 |
| 3 | 1 | c | 0 | 0 | 0 | 1 | 1 |
| 4 | 1 | 0 | 0 | 0 | c | 1 | 1 |
| 5 | 0 | 1 | 0 | c | 0 | 1 | 1 |

where $c$ is a chord, the $T$ list is the sum of spanning-tree edges for each row, and the $R$ list is the sum of chords for each row.

In any row, the chords are defined first (superdiagonally), i.e., before or to the left of the spanning-tree edges. According to the authors

> "The smallest rings are those that link previously defined atoms soonest and, hence, are favoured by the maximization procedure, which puts their ring-closure entries as far up and to the left as possible."

For each row in the matrix, if there is a ring-closure entry, this usually defines the smallest cycle in terms of entries above and to the left in the matrix. Cycles can thus be found by backtracking up the matrix. Usually this will define an SSSR; however, where there are cycles with no unique edges (i.e., embedded rings), it is necessary to resort to the familiar ring sum combinations of cycles already found.

Isolated, spiro, or simple fused rings can be removed from the set used for these linear combinations, since they do not lead to new smallest rings. This can be done by looking for paths of unique edges between two points that are longer than, or of equal length to, any other path between those points. This path (or paths) can be removed since any smaller cycles can be defined by the remaining paths. This is effectively a pruning of outer rings equivalent to Shelley's peeling mentioned earlier.

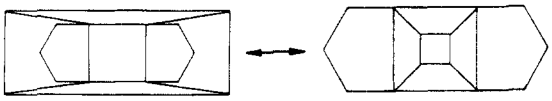

Figure 10.

Such pruning often exposes previously hidden edges that can then be used to find the embedded rings.

This process is known to fail for such structures as cubane and dodecahedrane, for which the full range of ring sum combinations is required.

**Elk,**[76] in contrast, does not present an algorithm for finding an SSSR, but expounds a technique for drawing a structure such that all SSSR rings are included within the perimeter ring that, in bridged polycyclic structures, is usually non-SSSR. (In two further papers he investigates the theoretical basis on which ring perception for taxonomic purposes should be based.[77,78])

He reiterates that Euler's formula refers to a spherical projection while a planar (Schlegel) projection moves exactly one of the faces to the outer perimeter of the whole structure; hence, the nullity (determined by the Cauchy formula) is 1 less than Euler's polyhedral formula.

Elk then proceeds to give an alternative to the Cauchy formula for determining the nullity. This relies on the concept of "incidence excess", a function of the number of vertices with ring connectivities (incidences) greater than 2. The number of rings is calculated as 1 plus half the incidence excess, i.e.

$$\mu = 1 + \tfrac{1}{2}\sum(i_v - 2)$$

where $\mu$ is the nullity, $\sum$ is the sum over all vertices $v$, and $i_v$ is the ring connectivity of each vertex $v$.

In terms of visual inspection to determine an SSSR, Elk shows that if the largest ring of a planar projection is larger than the perimeter ring, then there exists a different planar projection in which the largest ring is the perimeter ring, as shown in Figure 10.

The underlying logic of the Schlegel projection is considered and produces the following theorem: If the largest ring illustrated in a connected set of rings is bigger than the boundary ring, the Schlegel projection chosen does not yield the SSSR. The aim is to produce the **maximal** Schlegel projection to ensure that, for visual inspection at least, there is no ambiguity in determining ring sizes for an SSSR. However, where there are several largest rings leading to one becoming the boundary ring, the ambiguity with respect to the choice of rings remains.

Although this paper raises some very important topological points related to projection and the SSSR, the idea of manipulating a structure to obtain its maximal Schlegel projection seems to have little practical value for trying to find an SSSR automatically. By virtue of the nature of the processing, algorithms such as Zamora's will usually find the maximal projection SSSR rings by default, without any active manipulation of the structure input. However, these ideas could be important in the display of structures and could be used in algorithms similar to that of Shelley.

## CHOOSING A RING SET

This review has shown that many algorithms have been devised and implemented in an attempt to achieve a variety of ring sets. It can be shown quite readily that each of these approaches has its disadvantages or failures, some more major than others, in terms of the efficiency of the algorithm or of the ring set obtained. As a quick summary, Table II shows that when the categories in this review are used, four initial ring sets give rise to nine final ring sets.

In addition to presenting the algorithms, it is also desirable to review the ring sets produced by them, in particular with respect to obtaining some idea of the "optimum" ring set for representing structures for retrieval. The main problem is one of including only the *necessary* number of rings that is *sufficient* to describe ring systems. For instance, Elk[77] considers the definition of a face of a structure and uses cubane to illustrate his points. Viewing cubane as a polyhedron, one can identify 6 "simple" faces (four-edged), 12 "double" faces (six-edged), 20 "triple" faces (eight-edged and six-edged), 15 "quadruple" faces (eight-edged and six-edged), 6 "quintuple" faces (four-edged), and 1 "sextuple" face (the entire cube). It can be seen that the simple and quintuple faces share a common contour, as do the six-edged double and quadruple faces, the simple and eight-edged quadruple faces, and half of the triple faces with the other half. Thus, each edge is represented as having effectively two sides. By counting those faces sharing a common contour only, one is left with 28 faces in cubane. Although these are definitely sufficient to describe cubane, they are far more than is strictly necessary for most purposes. Elk observes that to define completely the topology of cubane it is necessary only to include four of the simple faces, chosen so that they cover all vertices and edges. However, since the nullity for cubane is 5, most systems regard cubane as having five such rings due to their reliance on a fundamental basis or an SSSR. This requires either the arbitrary addition of one simple face to the set of four already chosen or the arbitrary exclusion of one simple face from the set of all six simple faces. However, if one considers the heteroatom analogue of cubane given in Figure 11, choosing four or five of the simple faces will not sufficiently describe it. If the infinite region is excluded, the set of four simple faces would all contain one phosphorus atom, while the SSSR would also contain the all-carbon simple face. Neither of these would contain the simple face with two phosphorus atoms. It is obvious that such a situation could lead to identical structures producing different ring analyses and hence a loss of recall. These problems are particularly acute for highly symmetric or complex structures. If any multiple faces are added, then no more discrimination is achieved; the only effect is to reduce the precision by the inclusion of unnecessary rings.

**Table II.** Summary of Ring Set Categories[a]

| initial ring set | final ring set |
|---|---|
| All simple-cycles | $\beta$-rings |
| All cycles | Essential Set of Essential Rings<br>(Set of $\mathcal{K}$-rings?)<br>Real rings (maximum proper covering set)[*]<br>All cycles |
| Fundamental basis | All cycles<br>Fundamental basis<br>Fundamental-rings<br>Supplemented Smallest Set of Smallest Rings [*]<br>Smallest Set of Smallest Rings |
| Smallest Set of Smallest Rings | Smallest Set of Smallest Rings<br>Set of $\mathcal{K}$-rings |

[a] The asterisk indicates that heuristics were used to supplement the ring set.

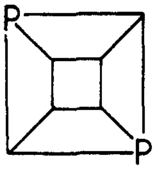

Figure 11.

Hence, for cubane, each simple face is necessary, but not sufficient. Including all simple faces achieves sufficiency with the minimum of necessary rings. The inclusion of multiple faces is not necessary. For cubane, the optimum ring set is realized by inclusion of all simple faces, i.e. all regions, finite and infinite, of the maximal Schlegel projection.

For the demanding retrieval requirements and perception environment presented by the structurally explicit parts of generics, the following broad criteria have been used to define the general optimum ring set:

- 100% recall
- high precision
- invariant set
- concept amenable to manual analysis and assignment
- consistent handling of graph and subgraph isomorphism (full and substructure searching)
- extendable to structurally explicit parts of generics (and potentially infinite graphs) in such a way as to be orientation and partition independent and have limits on cycle generation.

The list of final ring sets can now be considered in terms of this list of requirements (the fundamental ring set is omitted since it has been superseded by the $\beta$-ring set).

**The Set of All Cycles and the Set of All Simple Cycles.** These guarantee 100% recall, but in large and complex structures there will be an overgeneration of rings to give far more than are necessary, with resultant high processing costs and low precision for substructure search. The ring sets are invariant, but can be too large for manual analysis. They are consistent for isomorphism and can be extended to structurally explicit generics, but with consequently severe processing overheads compounding the low precision encountered with specifics. All current retrieval systems regard these ring sets as impractical, with the primary use being as the initial ring set from which a more practical subset can be chosen.

**Fundamental Basis Sets.** These can give less than 100% recall due to a variant ring set. Obviously, the rings generated are dependent upon such things as the tree or matrix used so that similar or identical query and file structures may produce a wide range of different fundamental sets. This has unpredictable effects upon both recall and precision, making manual analysis difficult and extension into structurally explicit generics fraught with problems. However, due to their ease of generation, they are generally used as a basis for refinement of the ring set to an SSSR or a heuristically supplemented set.

**Smallest Set of Smallest Rings and the Set of $\mathcal{K}$-Rings.** Although the SSSR provides a more refined fundamental basis, it still suffers from being a variant ring set and so cannot guarantee 100% recall. An SSSR is thus not consistent for isomorphism, as has been shown already by the cubane analogue in Figure 11. The set of $\mathcal{K}$-rings, being the union of all SSSRs, overcomes this variance. Both these sets have a lack of precision in complex ring systems, but are easy to generate and can readily be implemented for structurally explicit generics. The set of $\mathcal{K}$-rings is easy to analyze manually and is particularly suitable for extension into structurally explicit generics. These sets form the basis for most implemented systems, with a few using additional heuristic supplements.

**Heuristically Supplemented Ring Sets.** The maximum proper covering set assumes that all cycles have already been found, from which a selection can be made. Several limitations of this set have already been given. The classification of rings as being either real or pseudo results in the anomaly of the infinite region of cubane being pseudo, while the infinite region of norbornane is real. This does not even have the predictability of the arbitrary exclusions of the SSSR or the omission of nonsymmetrically equivalent infinite regions from the set of $\mathcal{K}$-rings. The minimum spanning set and the minimum covering set are other names used for the SSSR and have been supplemented by the inclusion of all rings up to and including seven and eight vertices, respectively. These heuristic selection criteria are intended to reflect the essential features of a ring system for synthesis planning. For retrieval purposes, the inclusion of these rings permits certain multiple faces to join the set. Cubane, for instance, would be regarded as having 16 six-edged rings in addition to the 6 four-edged rings, while inclusion of eight-edged rings will add another 6 rings.[10] Such an unnecessary situation can significantly decrease the precision.

**$\beta$-Rings.** This set may be inconsistent for subgraph isomorphism, and so 100% recall is not guaranteed. Precision is relatively high, and the ring set is invariant. However, it is not readily amenable to manual analysis, especially not when extension to structurally explicit generics is considered. It relies upon generating the set of all simple cycles first, which has severe consequences with respect to cycle generation limits and subsequent elimination of nonrelevant simple cycles. Due to its reliance upon taking all linear combinations of cycle vectors, it would need a "whole structure" approach, making determination of logical relationships very difficult. Trying to take linear combinations across the whole spectrum of partial structures could result in rings classified as $\beta$-rings at a lower partial structure ceasing to be $\beta$-rings at a higher level. Similarly, in substructure searching, the query may contain a ring that is not the linear combination of three or more smaller $\beta$-rings, but which in the file structure is such a combination and hence not a $\beta$-ring. In these cases, the substructure will fail to retrieve the relevant file structure, leading to a loss of recall.

**Essential Set of Essential Rings.** In a similar manner to the set of $\beta$-rings, this ring set is selected from the set of all cycles and so there is a problem with limiting the production of cycles in complex ring systems. From the set of all cycles, the set of simple cycles is chosen, to which a series of complex rules are applied to enable elimination of the non-ESER cycles. As with the above, these rules can make manual analysis difficult and introduce the problem of inconsistent substructure handling. In this case, the inclusion of cycles in the set is affected by the atom types present in the ring system. If only part of the ring system is described, as in a substructure query, then there is a possibility of retrieval failure.

## CONCLUSIONS

In terms of specific structure systems it does not matter particularly which ring set is used for full structure retrieval, so long as the ring set obtained is consistent. The governing factor becomes the ease of representation of the chosen ring set and the retrieval performance obtained from this representation. Unfortunately, many of the algorithms reviewed fail for the more complex ring systems, and so consistency of ring set perception is not maintained. Of the ring sets reviewed, the set of $\mathcal{K}$-rings seems the most appropriate minimum requirement, but Plotkin's algorithm to find this set has known limitations.

For substructure retrieval of specific structures, the only ring set fulfilling the criteria for subgraph isomorphism is the set of all simple cycles. However, perception of this set is time-consuming for complex ring systems, and the size of the set makes it difficult to represent concisely. Furthermore, the precision of the retrieval is likely to be low, especially if the end-user does not expect a six-membered ring to retrieve cubane, for instance! The minimum requirement is, once again, the set of $\mathcal{K}$-rings.

The general conclusion is that none of the ring sets suggested so far are ideal for specific structure systems and that most of the algorithms have their problems. It is necessary therefore to consider the theoretical basis of ring perception further, to obtain a better solution that is applicable to specific and generic environments. This is the topic of the following paper.[2]

## ACKNOWLEDGMENT

This research was made possible through funding provided by IDC (Internationale Dokumentationsgesellschaft für Chemie mbH), whose staff also provided many useful discussions. Thanks also go to Dr. P. Willett, Department of Information Studies, for many helpful suggestions.

## BIBLIOGRAPHY

(1) Downs, G. M. Computer storage and retrieval of generic structures in patents: ring perception and screening to extend the search capabilities. Ph.D. Thesis, University of Sheffield, March 1988, Chapter 3 (Ring perception algorithms for chemical graphs).

(2) 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.* (second of four papers in this issue).

(3) Downs, G. M.; Gillet, V. J.; Holliday, J. D.; Lynch, M. F. Computer Storage and Retrieval of Generic Chemical Structures in Patents. 9. An Algorithm To Find the Extended Set of Smallest Rings in Structurally Explicit Generics. *J. Chem. Inf. Comput. Sci.* (third of four papers in this issue).

(4) Downs, G. M.; Gillet, V. J.; Holliday, J. D.; Lynch, M. F. Computer Storage and Retrieval of Generic Chemical Structures in Patents. 10. Assignment and Logical Bubble-Up of Ring Screens for Structurally Explicit Generics. *J. Chem. Inf. Comput. Sci.* (fourth of four papers in this issue).

(5) Fugmann, R.; Braun, W.; Vaupel, W. GREMAS—a new method of classification and documention in chemistry. *Nachr. Dok.* **1963**, *14(4)*, 179–190.

(6) Carruthers, L. Automatic ring perception in generic chemical structures. M.Sc. Dissertation, University of Sheffield, Sept 1983.

(7) Sorkau, E. Ringerkennung in chemischen Strukturen mit dem Computer. *Wiss. Z. Tech. Hochsch. Leuna-Meuseburg* **1985**, *27*, 765–770.

(8) Gray, N. A. B. *Computer-assisted structure elucidation*: Wiley-Interscience: New York, 1986.

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

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

(11) 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.

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

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

(14) Mateti, P.; Deo, N. On algorithms for enumerating all circuits of a graph. *SIAM J. Comput.* **1976**, *5*, 90–99.

(15) Read, R. C.; Tarjan, R. E. Bounds on backtrack algorithms for listing cycles, paths and spanning trees. *Networks* **1975**, *5*, 237–252.

(16) Tiernan, J. An efficient search algorithm to find the elementary circuits of a graph. *Commun. ACM* **1970**, *13*, 722–726.

(17) von Beiss, G.; Jänicke, W.; Meissler, H. G. Ein Algorithmus zur Bestimmung aller Elementarkreise eines gerichteten Graphen. *Wiss. Z. Tech. Hochsch. Leuna-Merseburg* **1971**, *19*, 103–111.

(18) Danielson, G. H. On finding the simple paths and circuits in a graph. *IEEE Trans. Circuit Theory* **1968**, *CT-15*, 294–295.

(19) Ponstein, J. Self-avoiding paths and adjacency matrix of a graph. *SIAM J. Appl. Math.* **1966**, *14*, 600–609.

(20) Yau, S. S. Generation of all Hamiltonian circuits, paths, and centers of a graph, and related problems. *IEEE Trans. Circuit Theory* **1967**, *CT-14*, 79–81.

(21) Kamae, T. A systematic method of finding all directed circuits and enumerating all directed paths. *IEEE Trans. Circuit Theory* **1967**, *CT-14*, 166–171.

(22) Ardon, M. T.; Malik, N. R. A recursive algorithm for generating circuits and related subgraphs; *Asilomar Conf. Circuits Syst.* **1971**, *5*, 279–284.

(23) Floyd, R. W. Non-deterministic algorithms. *J. ACM* **1967**, *14*, 636–644.

(24) Tarjan, R. E. Enumeration of the elementary circuits of a directed graph. *SIAM J. Comput.* **1973**, *2*, 211–216.

(25) Johnson, D. B. Finding all the elementary circuits of a directed graph. *SIAM J. Comput.* **1975**, *4*, 77–84.

(26) Schwarcfiter, J. L.; Lauer, P. E. A search strategy for the elementary cycles of a directed graph. *BIT* **1976**, *16*, 192–204.

(27) Roberts, S. M.; Flores, B. Systematic generation of Hamiltonian circuits. *Commun. ACM* **1966**, *9*, 690–694.

(28) Bertziss, A. T. *Data structures: theory and practice*; Academic Press: New York, 1971.

(29) Weinblatt, H. A new search algorithm for finding the simple circuits of a finite directed graph. *J. ACM* **1972**, *19*, 43–56.

(30) Ehrenfeucht, A.; Fosdick, D. L.; Osterweil, L. J. An algorithm for finding the elementary circuits of a directed graph. Technical Report CU-CS-024-73; Department of Computer Science, University of Colorado: Boulder, 1973.

(31) Syslo, M. M. The elementary circuits of a graph, Algorithm 459. *Commun. ACM* **1973**, *16*, 632–633.

(32) Syslo, M. M. Remark on Algorithm 459: the elementary circuits of a graph. *Commun. ACM* **1975**, *18*, 119.

(33) Loizou, G. On a cycle-finding algorithm. *Inf. Process. Lett.* **1980**, *11*, 33–36.

(34) Cartwright, D.; Gleason, T. C. The number of paths and cycles in a diagraph. *Psychometrika* **1966**, *31*, 179–199.

(35) Corey, E. J.; Wipke, W. T.; Cramer, R. D.; Howe, W. J. Techniques for perception by a computer of synthetically significant structural features in complex molecules. *J. Am. Chem. Soc.* **1972**, *94*, 431–439.

(36) Corey, E. J.; Wipke, W. T. Computer-assisted design of complex organic syntheses. *Science* **1969**, *166*, 179–192.

(37) Shelley, C. Heuristic Approach for Displaying Chemical Structures. *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 61–65.

(38) Balaban, A. T.; Filip, P.; Balaban, T. S. Computer program for finding all possible cycles in graphs. *J. Comput. Chem.* **1985**, *6*, 316–329.

(39) Fujita, S. Logical Perception of Ring Opening, Ring Closure, and Rearrangement Reactions Based on Imaginary Transition Structures. Selection of the Essential Set of Essential Rings (ESER). *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 1–9.

(40) Fujita, S. A New Algorithm for Selection of Synthetically Important Rings. The Essential Set of Essential Rings for Organic Structures. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 78–82.

(41) Kudo, Y. *Organic Synthesis and Computers*; R&D Report 39; CMC Press: Tokyo, 1983; p 49.

(42) Nickelsen, H. Ringbegriffe in der Chemie-Dokumentation. *Nachr. Dok.* **1971**, *3*, 121–123 (and associated microfiche).

(43) Maxwell, L. M.; Reed, G. B. Subgraph identification—segs, circuits and paths. Presented at the 8th Midwest Symposium on Circuit Theory, 13.0-13.10, Colorado State University, June 1965.

(44) Welch, J. A mechanical analysis of the cyclic structure of undirected linear graphs. *J. ACM* **1966**, *13*, 205–210.

(45) Seshu, S.; Reed, M. B. *Linear graphs and electrical networks*; Addison-Wesley: Reading, MA, 1961.

(46) Fugmann, R.; Dölling, U.; Nickelsen, H. A topological approach to the problem of ring structures. *Angew. Chem., Int. Ed. Engl.* **1967**, *6*, 723–733.

(47) 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.

(48) Gotlieb, C. C.; Corneil, D. G. Algorithms for finding a fundamental set of cycles for an undirected linear graph. *Commun. ACM* **1967**, *10*, 780–783.

(49) Gibbs, N. A Cycle Generation Algorithm for Finite Undirected Linear Graphs. *J. ACM* **1969**, *16*, 564–568.

(50) Hsu, H. T.; Honkanen, P. A. A fast minimal storage algorithm for determining all the elementary cycles of a graph; Computer Science Department, The Pennsylvania State University: University Park, 1972.

(51) Paton, K. An algorithm for finding a fundamental set of cycles of a graph. *Commun. ACM* **1969**, *12*, 514–518.

(52) Jovanovich, A. D. Note on a modification of the fundamental cycle finding algorithm. *Inf. Process. Lett.* **1974**, *3*, 33.

(53) Ito, T.; Kizawa, M. The matrix rearrangement procedure for graph-theoretical algorithms and its application to the generation of fundamental cycles. *ACM Trans. Math. Software* **1977**, *3*, 227–231.

(54) Matyska, L. Fast algorithm for ring perception. *J. Comput. Chem.* **1988**, *9*, 455–459.

(55) Weise, A. Ein effectives Verfahren fur die automatische Erkennung von Ringen in chemischen Graphen. *Z. Chem.* **1981**, *21(10)*, 353–353.

(56) Plotkin, M. Mathematical Basis of Ring-Finding Algorithms in CIDS. *J. Chem. Doc.* **1971**, *11*, 60–63.

(57) Bersohn, M. An algorithm for finding the synthetically important rings of a molecule. *J. Chem. Soc., Perkin Trans. 1* **1973**, 1239–1241.

(58) Esack, A. A procedure for rapid recognition of the rings of a molecule. *J. Chem. Soc., Perkin Trans. 1* **1975**, 1120–1124.

(59) Cribb, D. W.; Ringeisen, R. D.; Shier, D. R. On cycle bases of a graph. *Congr. Numerantium* **1981**, *32*, 221–229.

(60) Schmidt, B.; Fleischhauer, J. A FORTRAN IV Program for Finding the SSSR of a Graph. *J. Chem. Inf. Comput. Sci.* **1978**, *18*, 204–206.

(61) Warshall, S. A theorem on Boolean matrices. *J. ACM* **1962**, *9*, 11.

(62) Salatin, J. D.; Jorgensen, W. L. Computer-Assisted Mechanistic Evaluation of Organic Reactions. 1. Overview. *J. Org. Chem.* **1980**, *45*, 2043–2051.

(63) Deo, N.; Prabhu, G. M.; Krishnamoorthy, M. S. Algorithms for generating fundamental cycles in a graph. *ACM Trans. Math. Software* **1982**, *8*, 26–42.

(64) Stepanec, G. F. Basis systems of vector cycles with extremal properties in graphs (in Russian). *Usp. Mat. Nauk.* **1964**, *19/2*(116), 171–175.

(65) Zykov, A. A. *Theory of finite graphs*; Nauka: Novosibirsk, 1969.

(66) Syslo, M. M. Fundamental sets of cycles of a graph. *Zastow. Mat.* **1973**, *13*, 399–409.

(67) Hubicka, E.; Syslo, M. M. Minimal bases of a graph. *Recent advances in graph theory*, Proceedings of 2nd Czechoslovak Symposium on Graph Theory; Fielder, M., Ed.; Academia: Prague, 1975; pp 283–293.

(68) Syslo, M. M. On cycle bases of a graph. *Networks* **1979**, *9*, 123–132.

(69) Syslo, M. M. Minimum length cycle bases of a graph. *Methods Oper. Res.* **1980**, *37*, 285–290.

(70) Kolasinska, E. On a minimum cycle basis of a graph. *Zastow. Mat.* **1980**, *16*, 631–639.

(71) Tarjan, R. Depth-first search and linear graph algorithms. *SIAM J. Comput.* **1972**, *1(2)*, 146–160.

(72) Deo, N. Breadth- and depth-first searches in graph-theoretic algorithms. *J. Comput. Soc. India* **1974**, *4(2)*, 19, 31–37.

(73) Horton, J. D. A polynomial-time algorithm to find the shortest cycle basis of a graph. *SIAM J. Comput.* **1987**, *16(2)*, 358–366.

(74) Hendrickson, J. B.; Grier, D. L.; Toczko, A. G. Condensed Structure Identification and Ring Perception. *J. Chem. Inf. Comput. Sci.* **1984**, *24*, 195–203.

(75) Hendrickson, J. B.; Toczko, A. G. Unique Numbering and Cataloging of Molecular Structures. *J. Chem. Inf. Comput. Sci.* **1981**, *23*, 171–177.

(76) Elk, S. B. Derivation of the Principle of Smallest Set of Smallest Rings from Euler's Polyhedron Equation and a Simplified Technique for Finding This Set. *J. Chem. Inf. Comput. Sci.* **1984**, *24*, 204–206.

(77) Elk, S. B. Effect of Taxonomy Class and Spanning Set on Identifying and Counting Rings in a Compound. *J. Chem. Inf. Comput. Sci.* **1985**, *25*, 11–16.

(78) Elk, S. B. Topologically Different Models To Be Used as the Basis for Ring Compound Taxonomy. *J. Chem. Inf. Comput. Sci.* **1985**, *25*, 17–22.

# Theoretical Aspects of Ring Perception and Development of the Extended Set of Smallest Rings Concept

GEOFFREY M. DOWNS, VALERIE J. GILLET, JOHN D. HOLLIDAY, and MICHAEL F. LYNCH*

Department of Information Studies, University of Sheffield, Sheffield S10 2TN, U.K.

Received October 18, 1988

There are many unresolved issues concerning the definition of an optimum ring set for retrieval purposes. This paper considers the problems associated with processing planar (two-dimensional) representations of three-dimensional structures. To overcome the ambiguity of such representations, a new ring set is defined in terms of simple faces and cut faces. The concept of a cut-vertex graph is introduced to explain the combinatorial relationship between the number of simple faces and the number of planar embedments.

## INTRODUCTION

The previous paper[1] reviewed algorithms and ring sets devised to represent and handle chemical structures and reached the conclusion that none of them presents an ideal solution even for specific structures. It is interesting to note that many of the methods developed have resorted to the use of heuristics, indicative of an underlying and hitherto unresolved problem.

The work on ring perception reported in this series of papers has concentrated on providing a solution for generic environments, of which the specific environment is a special case. In particular, it has concentrated on extension of ring perception capabilities into structurally explicit generics, i.e., those described in full structural detail represented by partial connection tables. To provide a general solution, it has been necessary to consider the theoretical basis of ring perception in greater detail. Due to the lack of suitable formalisms for the description of generalized ring systems as a whole, it has been necessary to consider individual cases and extrapolate to reveal the underlying theory necessary to support such formalisms. This paper presents the conclusions of these considerations, which are reported more fully in reference 2. The task has not proved trivial, but as Elk[3] states

> "What is important is the recognition of the problem rather than an attempt to 'paper over' the inconsistencies. *Ad hoc* solutions would never have been selected in the past if a 'simple', 'obvious' solution existed."

As shown in the preceeding review paper,[1] Zamora[4] has achieved a certain degree of generalization by defining three distinct classes of ring system in terms of overlap within the vertex and edge sets of the smallest set of smallest rings (SSSR). Methods that reduce a graph to its basic form, such as those of Lederberg,[5] Carhart,[6] Sridharan,[7] and Balaban[8] similarly go only part way to providing a generalization and do not help particularly in making a solution any easier.

Analyses of the distribution and type of ring systems within specific structure databases, by Adamson et al.,[9,10] have shown that most occurrences are of very simple systems. The more recent studies of the project's database of structurally explicit generics, by Mawby[11] and Kirk,[12] show similar results. However, ring perception techniques have to cater for the worst possible cases of complex systems, and they are judged by their handling of these cases. The review paper has shown that current techniques have difficulty with the more complex cases, and many instances of failure have been mentioned. Before extension to the complexity of variation allowed in structurally explicit generics, it seemed sensible first to develop a sound perception technique that would process the worst-case specifics, rather than build on the recognized limitations of current specific techniques. It should be noted, therefore, that most of the circumstances considered in this research are extreme cases and are not necessarily representative to the types of ring system commonly encountered. Most ring systems in specific structures are simple to process and do not cause problems for other techniques nor for the extended set of smallest rings (ESSR) concept presented here.

## ASPECTS OF GRAPH THEORY

This paper assumes an elementary knowledge of graph theory; further explanation, clarification, and definition of the concepts outlined below are given in the many general treatments and introductions to the subject, such as those by Behzad and Chartrand,[13,14] Berg,[15] and Mayeda,[16] and the classic work by Harary.[17] More applied works include those by Marshall[18] and Bondy and Murty[19] and the excellent presentation by Deo.[20] A few texts deal specifically with chemical applications, including those by Balaban[21] and the highly recommended two-volume work by Trinajstić.[22] This paper also uses the terms and definitions given in the review paper[1] such as component, block, simple cycle, fundamental basis, $\mathcal{K}$-ring and $\beta$-ring.

The long-recognized correspondence of the atoms and bonds of a chemical structure diagram to the vertices and edges of a graph allows graph theory to be used to represent, manip-

0095-2338/89/1629-0187$01.50/0