GMA: A Generic Match Algorithm for Structural Homomorphism, Isomorphism, and Maximal Common Substructure Match and Its Applications

Jun Xu

BIO-RAD Laboratories, Sadtler Division, 3316 Spring Garden Street, Philadelphia, Pennsylvania 19104-2596

Received June 26, 1995[⊗]

Structural homomorphism, isomorphism, and maximal common substructure match (MCSS) have been studied for many years. Traditionally, these problems are processed separately and considered as the factorial computing complexity relying on the number of atoms. This paper will show the following: (1) All these problems can be processed in one generic match algorithm (GMA) without raising computing complexity. (2) The computing complexity can rely on the adjacent degrees of atoms instead of simply the number of atoms. (3) Many sophisticated structural perception algorithms can be solved and simplified by using GMA in efficient ways. GMA is based upon the partial ordering set theory. The distinctive concept of GMA is that it considers a query structure as a "program", which will be run on the queried structure (or superstructure). Also, the paper reports its implementation in SSSR and other ring perception algorithms, absolute stereochemical configuration detection, and related problems.

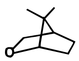
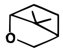
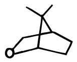
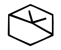
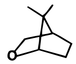
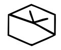
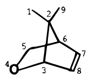
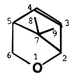
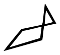
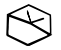
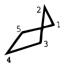
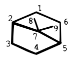
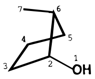
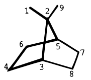
INTRODUCTION

1. Background. The generic structural match problem, including homomorphism, isomorphism and maximal common substructure match (MCSSM), has been studied for about a half century. Any chemical information system, if it has structure databases, structure-based property predictions, or structure/substructure operations, needs high performance structural match or perception algorithms. The structural match problem is a well-known example of an NP-complete problem.[1,2] Since the 1970s, a number of algorithms have been proposed and implemented. These algorithms have been reviewed by monographs and papers.[3−7] Traditionally, MCSSM[8−14] and other structure perception algorithms, such as smallest set of the smallest rings algorithm (SSSRA),[15−21] Cahn−Ingold−Prelog priority detection algorithm (CIP algorithm), chiral center detection algorithm (CCDA), Z−E configuration identification algorithm (ZECIA), and tautomer detection algorithm (TDA)[22−27] are studied separately and cost many man-years of effort. The structural match algorithms can be classified into three types:[7]

1. backtracking,
2. partitioning and relaxation, and
3. screening.

Back-tracking method is mathematically complete. If the algorithm itself is robust, it always gives correct results. Ray and Kirsch[28] reported their backtracking method in 1957, Xu and Zhang[29] reported their backtracking algorithm, which is based upon partial ordering set method, in 1989, and Dengler and Ugi[30] reported their backtracking algorithm in 1991. Generally, the back-tracking method is "time-consuming"; in the worst case, however, it is unlikely the worst case will arise very often in organic chemistry. Ullmann's algorithm[3], Sussenguth's algorithm,[31] Figueras' set reduction algorithm,[32] and Von Scholley's algorithm[33] are considered as the partitioning and relaxation method. It is believed that Lynch and his colleagues have made a significant contribution to the Screening method.[7] Both partitioning-relaxation method and screening method have been available in many commercial chemical information software systems. Partitioning-relaxation and screening can be efficient (if the screen is not too large). However, they can also lead to incorrect results.[7]

Table 1. Some Examples for the Classification of Generic Graph Match Problem

Type	QG[a]	TG[b]	Output
Topological Query			Yes (Homomorphic)
Edge-Colored Query			Yes (Homomorphic)
Node-Colored Query			No (Not homomorphic)
All-Colored Graph Match			Non-abundant Mappings: QG: 1 2 3 4 5 6 7 8 9 TG: 8 7 2 1 6 5 3 4 9
Substructure Query			Yes (Isomorphic)
Substructure Match			Abundant Mappings: QG: 1 2 3 4 5 TG: 2 3 4 5 7 TG: 3 4 5 7 2 TG: 1 6 5 7 2 TG: ....
MCSS			Non-abundant Mappings: QG: 2 3 4 5 6 7 TG: 3 4 6 5 2 1 TG: 3 4 6 5 2 9 TG: 3 8 7 5 2 9 TG: 3 8 7 5 2 1

[a] QG = query graph. [b] TG = target graph.

Based upon our previous HBA algorithm,[29] this paper will outline the GMA, a partial-ordering-based back-tracking algorithm, present the computing complexity. Based upon the GMA algorithm, the other structure perception algorithms become simpler and much more efficient. All algorithms reported here have been tested on Sadtler structure data-bases.

2. The Description and Essence of the Problem. A chemical structure can be represented as a chromatic graph.[34] The generic graph match problem can be divided into three levels:

[⊗] Abstract published in *Advance ACS Abstracts,* December 15, 1995.

0095-2338/96/1636-0025$12.00/0 © 1996 American Chemical Society

Table 2. Relations and the Chemical Applications among the Classifications of Generic Match Problem and Their Computing Complexity[b]

	complexity[a]	type 1	type 2	type 3	type 4
level 1	$m!$	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)
level 2	$\frac{m!}{(m-n)!}$	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)	class 1, class 2 (*a*, *b*, *c*, *d*, *e*, *f*)
level 3	$\frac{m!n!}{(m-k)!(n-k)!k!}$	class 2.2 (*a*, *b*)	class 2.2 (*a*, *b*)	class 2.2 (*a*, *b*, *d*)	class 2.2 (*a*, *b*)

[a] References 1, 2, and 8. Where m is the number of the nodes of query graph, n is the number of nodes of target graph, and k is the number of the nodes appearing both in query graph and target graph. [b] a = the application of computer−assisted structure elucidation. b = the application of computer−assisted organic synthetic design. c = the application of chemical structure/spectra database search. d = the application of molecular modeling. e = the application of combinatorial chemistry. f = the application of QSAR.

level 1: graph match, i.e., homomorphism,

level 2: subgraph Match, i.e., isomorphism, and

level 3: similar graph match, such as maximal common subgraph match. Functionally, the generic graph match can have four types:

type 1: node-colored graph match,

type 2: edge-colored graph match,

type 3: all-colored graph match, and

type 4: topological match. In terms of the output result, the generic graph match can have two classes:

Class 1: Graph query: The output is Yes/No, i.e., it reports if two graphs are homomorphic or isomorphic.

Class 2: Graph match: The output is a set of mappings between two graphs; i.e., it should report that if two graphs are homomorphic or isomorphic. If the two graphs match, the mappings should be reported. These mappings can be:

Class 2.1: nonabundant mapping and

Class 2.2: abundant mapping.

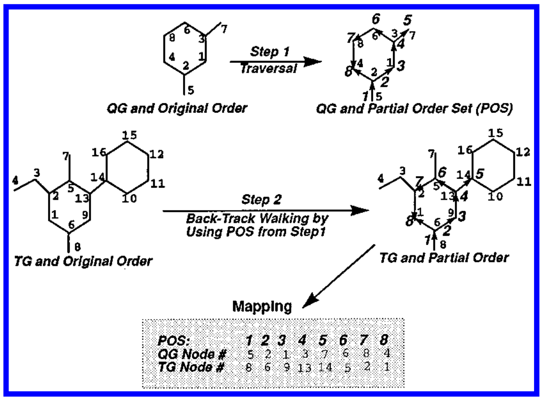
Some examples for the above-mentioned classifications are listed in Table 1. In chemical applications, the structure match requirements are the combinations of these classifications. The relations and the applications among these classifications are summarized in Table 2.

Any node-by-node graph match algorithm is of factorial computing complexity; it is one of the properties of NP-complete problems. However, it does not mean that there is no way to reduce the computing complexity in a practical scale. This paper will show that the back-tracking algorithm can be robust but very efficient. It also shows that a number of graph perception algorithms can be reduced into graph matches or query procedures and become easier to implement and faster to run.

PARTIAL ORDERING AND THE PRINCIPLE OF GENERIC GRAPH MATCH ALGORITHM

Essentially, a graph match procedure relies on an ordering (or labeling) system. In homomorphism, people may find a canonical procedure to determine a "linear" order for each node in a graph; however, there is no such kind of canonical procedure for isomorphism or MCSS search. The ordering in homomorphism is global, but the ordering in isomorphism or MCSS search has to be local. The local ordering is called partial ordering, which can carry local graphic information, such as node color, adjacency degree, the edges attached to the node, and the edge colors.

Traditionally, people concentrate on the result of the ordering rather than the ordering procedure. For example, in the screening method, the graph match procedure is basically divided into two steps: (1) screen creation (enumerating all combinations of orders) and (2) query graph comparison against the screen. If a query subgraph is not in the screen, the algorithm will not give a correct result. However, in GMA (generic match algorithm), the ordering procedure itself is important. The homomorphism or isomorphism is considered as the equivalency of partial (local) ordering procedure and MCSS is of partial equivalency of the procedure. GMA is based upon the following assumptions:



POS:	1	2	3	4	5	6	7	8
QG Node #	5	2	1	3	7	6	8	4
TG Node #	8	6	9	13	14	5	2	1

Figure 1. The principle of GMA.

1. There are many ways to order (label) a query graph QG. These orderings, which are called partial order sets (POS), can be accomplished by a graph traversal algorithm;

2. Each POS contains the equivalent QG graph information although it is encoded differently from the other POSs of the same QG;

3. If a target graph (TG) and QG are homomorphism or isomorphism, the same POS should be extracted from TG, otherwise, the POS can be partially extracted from TG, which should be the common subgraphs(s) of TG and QG.

Therefore, GMA consists of two steps:

Step 1. Traverse on QG to get POS;

Step 2. Use POS as the instruction set to walk on TG. If it is successful then QG and TG are homomorphic or isomorphic; otherwise output MCSS. These steps are graphically displayed in Figure 1.

Step 1 can be considered as encoding QG. The resulting POS is a "program", which will be executed in the step 2 on a target graph TG.

GMA ALGORITHM DESCRIPTION AND COMPUTING COMPLEXITY

1. Description of GMA. The flow chart of GMA is shown in Figure 2. The algorithm consists of part A and part B. Part B is the key part of GMA and is routinely used for graph homomorphism and isomorphism. Part B consists

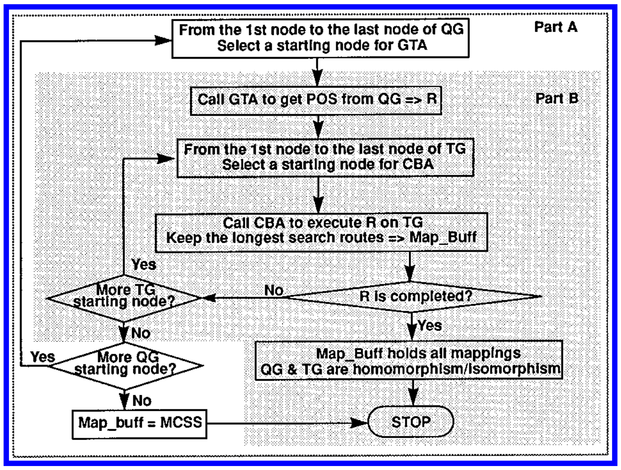

From the 1st node to the last node of QG
Select a starting node for GTA
Part A
Call GTA to get POS from QG => R
Part B
From the 1st node to the last node of TG
Select a starting node for CBA
Call CBA to execute R on TG
Keep the longest search routes => Map_Buff
Yes
More TG starting node?
No
R is completed?
No
Yes
Yes
More QG starting node?
Map_Buff holds all mappings
QG & TG are homomorphism/isomorphism
No
Map_buff = MCSS
STOP

Figure 2. The flow chart of GMA.

of graph traversal algorithm (GTA) and constrained back-tracking algorithm (CBA). GTA takes QG, outputs partial ordering set (POS). CBA walks on TG by using POS as constraints or instructions. CBA's walking is guided by POS instead of random walking.

POS is stored into R, which drives algorithm CBA like a "program". R is a two-dimensional data structure and is defined in Table 3.

R is a stack, which records a trace when GTA walks on QG. When GTA arrives at a multiple branch node, GTA will arbitrarily select one of the branches to go forward, and the rest of the unused branches will be kept in a "branch stack". The GTA keeps going until to it reaches "dead-end".

For example, the longest walking path on QG (see Figure 1) is

$$5 \rightarrow 2 \rightarrow 1 \rightarrow 3 \rightarrow 6 \rightarrow 8 \rightarrow 4$$

Node 2 and node 3 are multiple branched nodes, and their out-degrees are 2. When GTA arrives at node 2, for example, it arbitrarily takes node 1 as the next node to walk and keeps 2 → 4 in the branch stack. When GTA walks from node 5 to node 4, there are two edges pushed to the branch stack as shown in Figure 3.

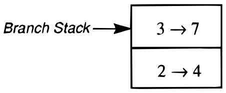


Figure 3. Branch stack running chart: 3 and 2 are "cut points" and 7 and 4 are "branch points".

GTA will "POP" up a new edge from the branch stack to complete the rest of walkings until the stack becomes empty. These pop-stack operations are recorded into R as the part information of the POS.

CBA is a POS-driven back-tracking algorithm; it executes the POS on the target graph TG (see Figure 1: step 2). At the beginning, CBA arbitrarily selects a node from TG. If the node satisfies the constraints of the starting node of the POS in R, then CBA will continue to test the next node, or else CBA will try another available TG node to start. For example, if CBA is testifying whether a TG edge x → y matches a QG edge u → v, CBA will not walk from node x to node y until it finds one of the following conditional expressions is true:

(1) Topological Match:

(edge(x, y) and out_degree(TG, x) ≥ out_degree(QG, u) and

edge(u, v) and out_degree(TG, y) ≥ out_degree(QG, v))

(2) Node-Colored Match:

(edge(x, y) and out_degree(TG, x) ≥ out_degree(QG, u) and

edge(u, v) and out_degree(TG, y) ≥ out_degree(QG, v) and

same_node_color(x, u) and same_node_color(y, v))

(3) Edge-Colored Match:

(same_edge_color(x, u) and same_edge_color(y, v) and

out_degree(TG, x) ≥ out_degree(QC, u) and

out_degree(TG, y) ≥ out_degree(QG, v))

(4) All-Colored Graph Match:

(same_edge_color(x, u) and same_edge_color(y, v) and

same_node_color(x, u) and same_node_color(y, v)

out_degree(TG, x) ≥ out_degree(QG, u) and

out_degree(TG, y) ≥ out_degree(QG, v))

An accessed node may have *n* edges attached to it, where *n* is its adjacent degree. When CBA walks on this node, only (*n* − *m*) edge(s) can be used to go to other nodes because *m* edges have been used before. Therefore, out-degree is (*n* − *m*). Figure 4 graphically shows the definition of the out-degree.

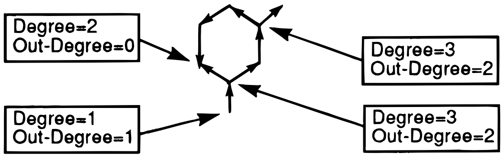


Figure 4. The definitions of node degree and node out degree.

When CBA walks at node x on TG, and x's potential mapping is node u on QG, let the out-degree of x be *j* and the out-degree of u be *k*. The number of ways (*nw*) to go from node x to other nodes can be calculated in formula 1:

$$nw = P_k^j \tag{1}$$

CBA chooses one of the out-degrees from *j*, and the (*k*−1) out-degree(s) from other (*j*−1) out-degrees will be pushed to an edge-stack. A mapping stack (MSK) is built to keep the intermediate mapping results. The length of MSK is variable. However, the current longest mapping path(s) should be stored to another 2D array called Map, which is also upgraded dynamically. Therefore, when CBA stops, it has three statuses:

1. R has been successfully finished and length(TG) = length(QG): homomorphism;

2. R has been successfully finished and length(TG) > length(QG): isomorphism;

3. R has not been successfully finished: MCSS match.

2. The Computing Complexity. As discussed above, in the worst case there are *nw* ways (eq 1) for GMA to walk from node x to other nodes on TG. Let QG have *m* nodes, QG have *n* nodes, the computing complexity (*CC*) of matching QG and TG is factorial as calculated in eq 2.

$$CC\,(QG \rightarrow TG) = \sum_{u=1}^{m}\{\prod_{x=1}^{n} P_{out\_degree(x)}^{out\_degree(u)}\} \tag{2}$$

Equation 2 tells us that the computing complexity of GMA depends on both the out-degree of TG's nodes and the out-degree of QG's nodes.

As known to all, the average adjacency degree of an organic molecule is less than 4, normally, out-degree ≤

Table 3. Data Structure Definition of R

	node	original order	partial order	node degree	node out−degree	edge color
data type	integer	index (integer)	integer	integer	integer	integer
meaning	atom	sequential label	sequential number based upon walking from one node to the other	adjacency of a node	the adjacency which can be used to move from current node to the other(s)	edge type (for chemical applications, it is bond type)
stack operation	POP[a]	cut point (partial order)	branch point (partial order)	no definition	no definition	defined as above

[a] "POP" is the instruction of the POP stack operation. When CBA walks on TG, "POP" tells it to stp going forward and get a new direction from the stack. The new walking direction is from "cut point" to "branch point" (see Figure 1: node 13 → node 14, and node 6 → node 1, where node 13 and node 6 are "cut points", the other two are "branch points").

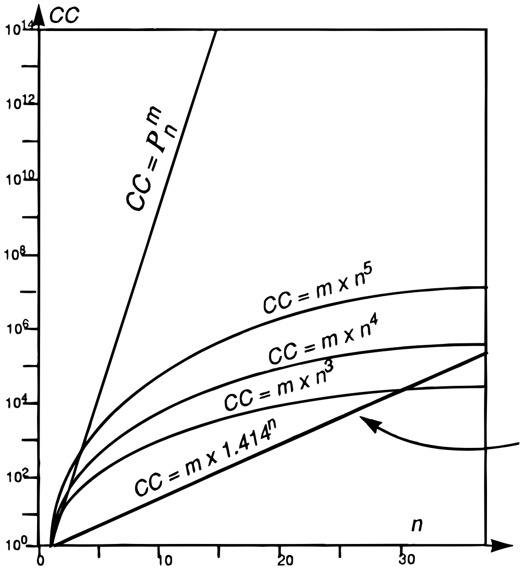


Figure 5. The comparison of GMA's computing complexity and other computing complexity levels.

(adjacency_degree − 1). Also suppose the node of QG always has the same degree as the one of TG (the worst case of the worst cases), then the equation (2) can be rewritten as equation (3):

$$CC'(QG \rightarrow TG) = m \times 2^{(n-k)} \qquad (3)$$

where k is the number of back-tracking. According to our experiments, back-tracking is very often, i.e., $k \gg n/2$, let $k = n/2$, (3) becomes:

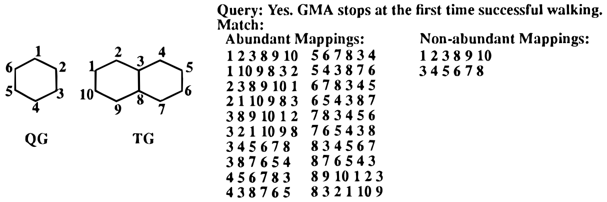
$$CC''(QG \rightarrow TG) = m \times 1.414^n \qquad (4)$$

As shown in Figure 5, GMA's computing complexity is much less than the factorial computing complexity.

For the MCSS match, GMA's computing complexity is calculated in eq 5

$$CC''(QG \rightarrow TG) = l \times m \times 1.414^n \qquad (5)$$

where l is the length of the maximal common substructure. Equation 5 indicates that the MCSS match of GMA does not have the factorial-of-factorial-level computing complexity (cf. Table 2, level 3). It should have higher performance than the other reported MCSS match algorithms. The main reasons why GMA has less computing complexity is that its computing complexity relies on the node out-degree in factorial scale. GMA also relies on the number of nodes in exponential manner. That is, GMA's computing complexity is exponential, not factorial (Table 2).



Query: Yes. GMA stops at the first time successful walking.
Match:

```
 Abundant Mappings:                 Non-abundant Mappings:
 1 2 3 8 9 10    5 6 7 8 3 4        1 2 3 8 9 10
 1 10 9 8 3 2    5 4 3 8 7 6        3 4 5 6 7 8
 2 3 8 9 10 1    6 7 8 3 4 5
 2 1 10 9 8 3    6 5 4 3 8 7
 3 8 9 10 1 2    7 8 3 4 5 6
 3 2 1 10 9 8    7 6 5 4 3 8
 3 4 5 6 7 8     8 3 4 5 6 7
 3 8 7 6 5 4     8 7 6 5 4 3
 4 5 6 7 8 3     8 9 10 1 2 3
 4 3 8 7 6 5     8 3 2 1 10 9
```

Figure 6. Structural query, match, and nonabundant mappings.

3. Methods Used To Enhance the Performance of GMA. Two strategies have been built into GMA to further improve the performance:

1. **Query, Mapping, and Nonabundant Mapping.** The requirement of graphic match and the one of graphic search are different. The former requests exact mappings, the latter (graphic query/search) just asks for the answer of "yes" or "no". In GMA, they consume different computing time. For structure/substructure query, GMA does not have to start searching for every node on TG. It may stop at any starting node when a mapping has been found. For structure/substructure match, GMA has to take a longer time, because all possible mappings should be found in order to answer how QG is mapped onto TG. In order to find nonabundant mappings, even longer time is taken to filter abundant mappings. Figure 6 shows an example.

By setting up the parameters, GMA can meet different match needs, these parameters are

```
int Algorithm_GMA( /* return "yes"/"no" if query=="yes",
                    * return number of mappings if query=="no",
                    * return number of mcss mappings if ((query=="no")&&
                    *                                    (do_mcss=="yes")&&
                    *      (structural or substructural mapping is not found)) */
   molecule: QG,
   molecule: TG,
   boolean: homomorphic_match,

boolean: isomorphic_match,
boolean: ignore_node_color,
boolean: ignore_edge_color,
boolean: query, /* If "yes", GTA has to be called before GMA is called */
boolean: get_mappings, /* if "no", no mapping output, used for structure query*/
boolean: non_abundant_mapping, /*if "yes", get_mappings should be "yes"*/
boolean: Z_E_configuration_match, /* when ignore_edge_color=="no" */
boolean: stereo_match, /* when ignore_node_color=="no" */
boolean: do_mcss);
```

When *do_mcss* == "*yes*", GMA has to spend extra time to keep recording and upgrading Map_buff (cf. Figure 2), so as to output MCSS mappings if it does not find subgraph or homomorphic graph.

2. **Large Structure File Query.** If GMA is used to search a larger structure database, set GMA parameter query to "yes", a much time can be saved. In this case, GTA is called to get the POS (partial order set) information from QG only one time at the beginning; then CBA can search on a number of TGs.

THE APPLICATIONS OF GMA

1. Smallest Set of Smallest Rings Algorithm (SSSRA). SSSRA can be used to generate screens for a structural data

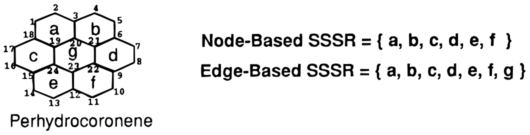
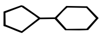
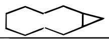
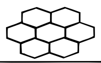
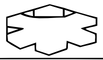
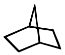
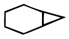
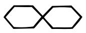
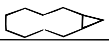
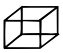
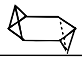
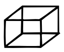
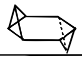
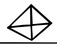
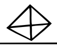

a
b
c
d
e
f
g
Node-Based SSSR = { a, b, c, d, e, f }
Edge-Based SSSR = { a, b, c, d, e, f, g }
Perhydrocoronene

Figure 7. Node-based SSSR versus edge-based SSSR.

Table 4. Examples for Eq 15

Structure	Ring Type	n	μ	ε	δ	Φ	Structure Having Φ-Size Ring
	Isolated-Ring	11	2	11	3	10	
	Fused-Ring	24	7	30	3	21	
	Bridged-Ring	7	2	8	4	6	
	Spiro-Ring	11	2	12	3	10	
	Polyhedron-Ring	8	5	12	3	6	
	Polyhedron-Ring	8	5	12	4	6	
	Polyhedron-Ring	4	3	6	3	3	

base search, structural depiction and many other structural perception algorithms. SSSR can be represented in eq 6

$$\mathrm{SSSR} = (\sum_{i=1}^{n} R_i) \in G \tag{6}$$

where R_i is a ring system, G is a structural graph, and n is the number of SSSR. G can be a connected or disconnected graph. To be general, G is considered as a topological graph. There are two definitions for SSSR as shown in (7) and (8)

$$\sum_{i=1}^{n} \mathrm{nodes}(R_i) = \mathrm{ring\_nodes}(G) \tag{7}$$

$$\sum_{i=1}^{n} \mathrm{edges}(R_i) = \mathrm{ring\_edges}(G) \tag{8}$$

If SSSR obeys eq 7, it is node-based SSSR. If SSSR obeys eq 8, it is edge-based SSSR. Where *nodes(R_i)* means the node set of ring R_i, and *edges(R_i)* means the edge set of ring R_i. Figure 7 shows the different results based upon the different definition.

In chemistry, SSSR implies edge-based SSSR. (If it is not specified, SSSR will mean edge-based SSSR in this paper.). SSSRA is to find the smallest set of smallest rings in a structure. Some example rings and their classifications are listed in Table 4.

Molecular unsaturation (ρ) for following formula[35]

$$C_xH_yN_zO_n$$

can be calculated in eq 9.

$$\rho = x - \frac{1}{2}y + \frac{1}{2}z + 1 \tag{9}$$

The total number of double bond equivalents (σ) is equal to (10)

$$\sigma = \delta + 2 \times \tau + \alpha \tag{10}$$

where δ is the total number of double bonds, τ is the total number of triple bonds, and α is the total number of aromatic bonds.

The number of the ring closure bonds (μ) of a molecule is equal, to eq 11

$$\mu = \rho - \sigma \tag{11}$$

A more general equation to calculate μ is given in (12). It considers the structure as having irregular valences, bonds, and disconnections (see Figure 8)

$$\mu = \beta - \nu + \xi \tag{12}$$

where β is the number of the covalent bonds in the molecule, ν is the number of the atoms (exclusive of hydrogen) in the molecule, and ξ is the number of the fragments of the molecule. For Figure 8, $\xi = 2$, the number of the ring closure bonds $\mu = 34 - 31 + 2 = 5$.

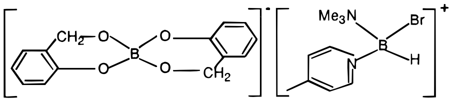


Figure 8. The example to illustrate the calculation of the number of ring closure bonds (μ). μ is the minimum bonds should be moved in order to make a structure become a tree graph.

Let n be the number of nodes of a molecule, and $\mu = 1$. Then the allowed maximal ring size of the SSSR (Φ) should be n. If $\mu > 1$, the maximal ring size will be reduced in $1/2^{\delta-2}$ for each additional ring closure bond.

Let δ be the maximal allowed degree for a structure, and $\delta > 2$, because any node in a ring should be at least a 2-degree node. Φ should be calculated in eq 13. The total number of the ring bonds of the molecule (ϵ) should be calculated in eq 14.

$$\Phi = n - \frac{\mu - 1}{2^{\delta-2}} \tag{13}$$

$$\epsilon = n + \delta(n - \Phi) \tag{14}$$

Combining eqs 13 and 14, eq 15 is figured out to calculate Φ

$$\Phi = \frac{(\delta + 2)n - \left(\epsilon + \frac{\mu - 1}{2^{\delta-2}}\right)}{\delta + 1} \tag{15}$$

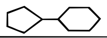
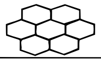
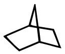
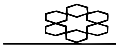
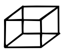
Note: Φ will only take the integer part of the eq 15. Some example calculations for eq 15 are listed in Table 4.

The maximal number of the rings of SSSR (Σ) will follow eq 16

$$\Sigma = \begin{cases} \mu & 2\mu \le n \\ \mu + 1 & 2\mu > n \end{cases} \tag{16}$$

Some calculation examples for eq 16 are listed in Table

Table 5. Examples for Eq 13

Structure	Ring Type	n	μ	2μ	Σ
	Isolated-Ring	11	2	4	2
	Fused-Ring	24	7	14	7
	Bridged-Ring	7	2	4	2
	Spiro-Ring	24	7	14	7
	Polyhedron-Ring	8	5	10	5+1

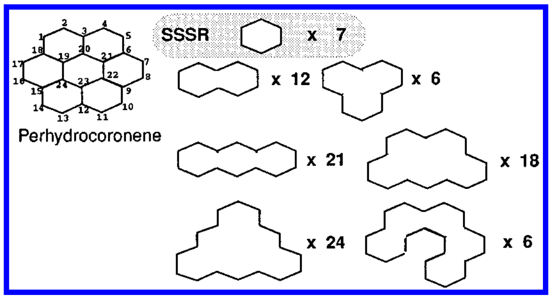


Figure 9. SSSR versus all rings.

5. The SSSRA algorithm is described as follows:

```
Algorithm SSSRA(molecule g)
{
   $\Phi$ = find_maximal_SSSR_ring_size (g);
   $\Sigma$ = find_the number_of_SSSR_rings (g);
   n = 3; /* the size of template ring */
   m = 0; /* the number of SSSR rings */
   while ( ($n \le \Phi$) && ( $m \le \Sigma$) )
   {
      tg = generate_ring_template( n ); /* tg is a template molecule */
      m = topological_match (g, tg); /* Call GMA */
      add new found rings if they are not the super ring set of the previous rings;
      n++;
   };
} /* end of Algorithm SSSRA */
```

An Example and Performance Analysis. As shown in Figure 9, perhydrocoronene has seven hexa-rings and 94 rings in total. Constrained by eqs 14 and 15, our SSSRA will stop when it has found all seven hexa-rings. Without using eqs 14 and 15, an algorithm has to

1. find all rings in the structure, then

2. delete the rings which are the supersets of the other smaller rings. Both of the procedures are time-consuming. The algorithm for deleting the supersets of the other smaller rings has factorial computing complexity, which is equal to eq 16

$$\Theta = m(\sum_{i=2}^{n} C_n^i) \qquad (16)$$

where Θ is the computing complexity, n is the number of currently found rings, and m is the number of the new rings which are potentially the supersets of the currently found rings. Our SSSRA, however, has avoided this computing complexity. In Figure 9, there are 7 6-member rings, which are SSSR, 12 10-member rings, 6 12-member rings, 21 14-member rings, 18 16-member rings, 24 18-member rings and 6 22-member rings. All the rings with sizes bigger than six are the supersets of six-member-rings. It will take more than 10 times the amount of time to find all those larger rings than to find all the six-member-rings.

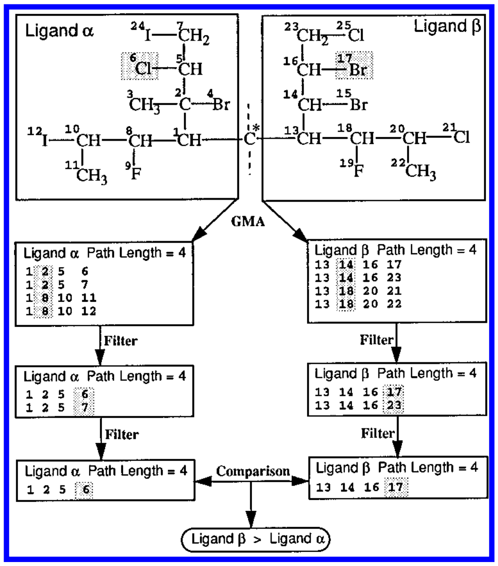


Figure 10. Paths for CIP priority calculation.

2. Cahn−Ingold−Prelog Priority Detection Algorithm (CIP Algorithm). CIP priority rules are rigorous ways to determine stereo configurations. However, the manual calculation of CIP priority is cumbersome for a complicated structure. The CIP priority calculation can be simplified and easily implemented by calling GMA.

Take Figure 10 as an example. In order to determine the CIP priorities of ligand α and ligand β, GMA is used to check if ligand α and ligand β are the same (structure query). If they are different, then the longest paths which can determine the priority should be listed. These paths are the mappings from n-depth nonbranched tree to ligand α and ligand β. The data flow of CIP algorithm is shown in Figure 10.

The CIP algorithm is described as follows:

```
Algorithm CIP_Order(molecule g, atom_index ligand_i, ligand_j)
{
  if (!Ligand_structure_match(g, ligand_i, ligand_j)) /* Call GMA */
  {
    n=1; CIP_found=0;
    while ((!CIP_found)&&(n $\le$ length(ligand_i))&&(n $\le$ length(ligand_j)))
    {
      Paths_i=find_length_n_paths(ligand_i, n);  /* Call GMA */
      Paths_j=find_length_n_paths(ligand_j, n);  /* Call GMA */
      x = filter_Path_i_to_find_the_highest_CIP_priority_atom_from_ligand_i;
      y = filter_Path_j_to_find_the_highest_CIP_priority_atom_from_ligand_j;
      if ((CIP_Priority(x > y)) report ligand_i > ligand_j; else
      if ((CIP_Priority(y > x)) report ligand_j > ligand_i; else
      n++; /* find longer paths */
    };
  } else print("CIP Order same");
} /* end of Algorithm CIP_Order */
```

3. Chiral Center Detection Algorithm (CCDA). If an atom has more than three bonds connecting to other atoms, and the other attachments are different from each other, the atom is a chiral center. To detect chiral centers, CCDA distinguishes these attachments by means of calling GMA

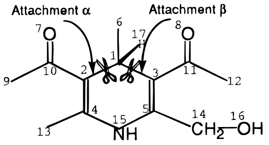


Figure 11. Partial ordering set and chiral center detection.

Table 6. Comparison of Partial Order Sets (POS) Attachments α and β

no.	POS (α)		POS (β)	
1	(1 → 2)	(C → C)	(C → C)	(1 → 3)
2	(2 → 10)	(C → C)	(C → C)	(3 → 11)
3	(10 → 7)	(C → =O)	(C → =O)	(11 → 8)
4	(10 → 9)	(C → C)	(C → C)	(11 → 12)
5	(2 → 4)	(C → =C)	(C → =C)	(3 → 5)
6	(4 → 13)	(C → C)	(C → C)	(5 → 14)
7	(4 → 15)	(C → N)	(C → O)	(14 → 16)
8	(15 → 5)	(N → C)	(C → N)	(5 → 15)
9	(5 → 3)	(C → =C)	(N → C)	(15 → 4)
10	(3 → 11)	(C → C)	(C → C)	(4 → 13)
11	(11 → 8)	(C → =O)	(C → =C)	(4 → 2)
12	(11 → 12)	(C → C)	(C → C)	(2 → 10)
13	(5 → 14)	(C → C)	(C → =O)	(10 → 7)
14	(14 → 16)	(C → O)	(C → C)	(10 → 9)

(structure query). The algorithm is defined as follows:

```
Algorithm CCDA(molecule g, atom_index atom_i)
{
    if (degree (atom_i) > 3) /* the degree includes proton attachment */
    {
      unknown_chiral_center = 1; /* don't know if it is chiral center yet */
      deg = degree (atom_i); /* atom_i is the potential stereo center */
      i = 1; /* the ith attachment */
      while ((unknown_chiral_center)&&(i < deg))
      {
        j = i + 1; /* the jth attachment */
        while ((unknown_chiral_center)&&(j ≤ deg))
        {
            if (all_colored_structure_match(g, atom_i, i, j)) /* call GMA */
            unknown_chiral_center = 0; /* known that atom_i is not a chiral center */
            else j++;
        };
        i++;
      };
      if (unknown_chiral_center) /* all attachments are different */
      print("atom_i is a chiral center");
    } else print("atom_i is not a chiral center");
} /* end of Algorithm CCDA */
```

The key aspect of CCDA algorithm is that of the same partial order set concept. As it is shown in Figure 11, the attachments α and β are the same substructure because it is a ring system, but they have different partial order sets.

The partial ordering set for attachment α and β are listed in Table 6. GMA starts to walk at edge (1 → 2), and gets POS (α) and then starts to walk at (1 → 3) in back-tracking way. GMA will stop walking at the 7th step (see Table 6) because POS (α) and POS (β) have differences at this step. It concludes that a structural graph is a nondirection graph, but its POS has directions, which reflects the local chemical environments. This POS property can also be used to detect the symmetry of a structure (discussed later in this paper).

4. Absolute Stereochemical Configuration Detection Algorithm (ASCDA). Detection of absolute stereochemistry is a well-known complicated problem. The reason is that people were unable to set up rigorous and general rules to canonicalize the procedure. Here we will show that this procedure can become very simple and easy to determine absolute stereochemical configuration.

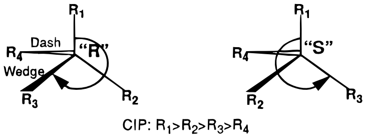


Figure 12. Graphic representation of stereochemistry.

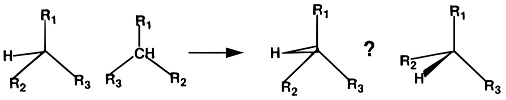


Figure 13. Implicit representations of protons lead to ambiguous in classification. The drawings on the left are not allowed.

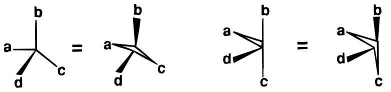


Figure 14. Opposite and adjacent position consistency. It should be prohibited to make two adjacent ligands have the same stereobond type or two opposite ligands have different stereobond type.

This method consists of (1) one clockwise traversal procedure and explicit representation rule; (2) two stereo consistency rules; (3) three canonical models; and (4) one standard *R*/*S* detection rule.

(1) One Clockwise Traversal Procedure[36] **and Explicit Representation Rules. Chiral Center:** If an atom has more than three attachments in a structure, and they are different from each other, the atom will be recognized as a chiral center or called the stereocenter.

CIP: The CIP method is used to rank the attachments to the stereocenter.

Graphic Representation for Stereocenter: wedged bonds are used for bonds *above the plane of the paper*. Dashed bonds are used for bonds *below the plane of the paper*. (See Figure 12.) The bigger end of a stereobond implies the end is closer to the reader; the other end is farther from the reader.

Rule 1: Explicit Representation. A stereocenter should be explicitly specified by at least ONE stereobond (wedged or dashed). A stereocenter should have more than three adjacencies; an implicit proton is considered as an unknown or unspecified stereo configuration (see Figure 13).

"R"/"S": If there are R_1, R_2, ..., R_n attachments to a stereocenter, they are listed in decreasing priority of CIP. Holding the lowest priority attachment (i.e., R_4) away from the viewer, the attachments are in **clockwise** order and so have a CIP descriptor of "*R*". If the attachments had been in a **counterclockwise** order the CIP descriptor would have been "*S*".

Rule 2: Clockwise Traversal. In order to simplify the "***R***/***S***" determination, we choose only **clockwise** traversal as the standard procedure to determine the geometric group sequence. Counterclockwise traversal is not used.

(2) Two Consistency Rules. Rule 3: Adjacent Position Consistency. A tetrahedron stereocenter has four ligands as shown in Figure 14, where **a** and **b**, **b** and **c**, **c** and **d**, and **d** and **a** are adjacent pairs. Each adjacent pair should have **different** stereo bond type, whether they are explicitly specified or not.

Rule 4: Opposite Position Consistency. A tetrahedron stereocenter has four ligands as shown in Figure 14, where **a** and **c** and **b** and **d** are opposite. Each opposite pair should

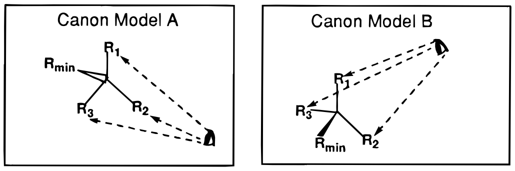


Figure 15. Canonical models.

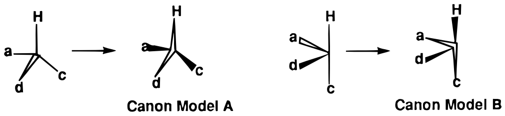


Figure 16. Convert the canonical model C to canon model A or B.

have the **same** stereobond type. No matter if they are explicitly specified or not.

(3) Three Canonical Models. Rule 5: Canonical Model A. If the lowest CIP priority attachment (R_{min}) has "wedge/dash in" bond as shown in Figure 15, this class of stereocenter is canon model A.

Rule 6: Canonical Model B. If the lowest CIP priority attachment (R_{min}) has "wedge/dash out" bond as shown in Figure 15, this class of stereocenter is canon model B.

Rule 7: Canonical Model C. If the lowest CIP priority attachment (R_{min}) has unspecified stereobond, this class of stereocenter can be converted to canon model A or B by using Rule 3 and Rule 4 as shown in Figure 16.

(4) Standard R/S Detection Rule. Rule 8: Standard R/S Detection Procedure (SDP). SDP should "walk" on ligands from R_{min} to the others (R_1, R_2, and R_3) ALWAYS in clockwise fashion as shown in Figure 17.

Canon model A: If $R_1 > R_2 > R_3$, then the stereocenter has "*R*" configuration, otherwise "*S*" configuration.

Canon model B: If $R_1 > R_2 > R_3$, then the stereocenter has "*S*" configuration, otherwise "*R*" configuration.

This canonical method works also for saccharide's stereochemical representation. An example is shown in Figure 18.

Fisher projections and Newman projections can be well handled by these rules as shown in Figure 19.

By means of algorithm CCDA and CIP algorithm, the CIP sequence of the ligands of a stereocenter are calculated. A clockwise sort algorithm (CSA) accepts the coordinates of the ligands of the stereocenter and sorts them in clockwise order. By comparing CIP sequence and clockwise order, the *R*/*S* configuration of the stereocenter are determined. This procedure is described in Figure 20.

Algorithm ASCDA is described as following:

```
Algorithm Algorithm_RSCDA(molecule g, atom_index atom_i)
{
    if ((degree (atom_i) > 3)&&(legal_attachment_draw(g,atom_i)))
    {
      Sequence = for all attachments call CIP_Order algorithm;
      Clockwise_order = CSA (the attachments to atom_i);
      if (sequence_equivalent(Sequence, Clockwise_order))
            atom_i is "R" ;
      else  atom_i is "S" ;
    } else print("atom_i is not a chiral center or stereo unspecified draws" );
} /* end of Algorithm RSCDA */
```

5. *Z*/*E* Configuration Identification Algorithm (ZECIA).

Since we have CIP algorithm, *Z*/*E* configuration identification procedure becomes easy. Any double bond system has to belong to the one of the cases as shown in Figure 21.

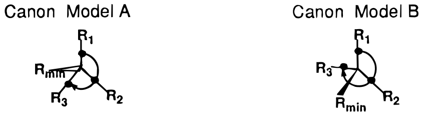


Figure 17. Standard R/S detection procedure always "walks" clockwise.

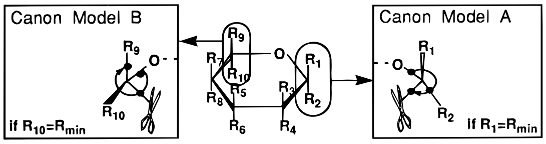


Figure 18. Rule 3−rule 7 applied on a monosaccharide structure.

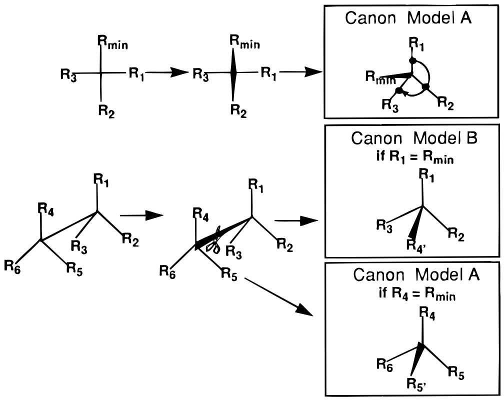


Figure 19. Rule 3−rule 7 applied on Fisher and Newman projections.

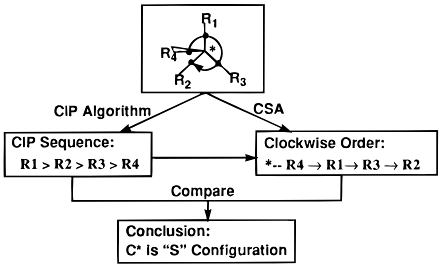


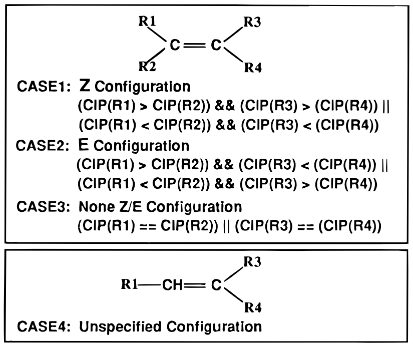
Figure 20. The procedure of determination of *R*/*S* configuration. CIP algorithm should be called first, so CSA knows which ligand has the lowest CIP priority to start to walk clockwise.

Algorithm ZECIA is described as following:

```
Algorithm Algorithm_ZECIA(molecule g, atom_index atom_i, atom_index atom_j)
{
    if (double_bond(g, atom_i, atom_j)
    {
      bi = bigger CIP_Order group at atom_i;
      bj = bigger CIP_Order group at atom_j;
      if (!bi || !bj) /*Figure 22: Case 3 or Case 4*/
      {
           if (same_attachments(g, atom_i)//same_attachments(g, atom_j))
                return "None Z/E configuration" ; /*Figure 22: Case 3*/
           else return "Unspecified configuration" ; Figure 22: Case 4*/
      }  else  /*Figure 22: Case 1 or Case 2*/
      {
           if (geometrically_same_side(bi, bj)
                return "Z configuration" ; /*Figure 22: Case 1*/
           else return "E configuration" ; Figure 22: Case 2*/
      }
    } else print("Not a double bond" );
} /* end of Algorithm ZECIA */
```

6. Potential Aromatic Ring Set Algorithm (PARSA).

In a 2D structure input system, PARSA is needed to check if aromatic rings are drawn correctly. In order to apply 4*n*

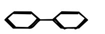
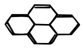
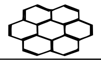
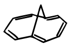
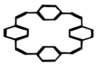
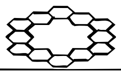


CASE1: Z Configuration
(CIP(R1) > CIP(R2)) && (CIP(R3) > (CIP(R4)) ||
(CIP(R1) < CIP(R2)) && (CIP(R3) < (CIP(R4))
CASE2: E Configuration
(CIP(R1) > CIP(R2)) && (CIP(R3) < (CIP(R4)) ||
(CIP(R1) < CIP(R2)) && (CIP(R3) > (CIP(R4))
CASE3: None Z/E Configuration
(CIP(R1) == CIP(R2)) || (CIP(R3) == (CIP(R4))

CASE4: Unspecified Configuration

Figure 21. Cases of *Z*/*E* configuration determination. Case 4: If one of the attachment bonds has π angular to the double bond, the configuration cannot be specified.

Table 7. Potential Aromatic Rings

Aromatic Rings Included in SSSR	Ring Type	Aromatic Rings NOT Included in SSSR	Ring Type
	Isolated-Ring		Fused-Ring
	Fused-Ring		Bridged-Ring
	Complex-Ring		Fused-Ring

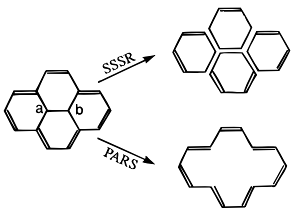


Figure 22. Four SSSR rings have been found from the structure, but none of them are PARS.

+ 2 rule on a ring system, all potentially aromatic rings need to be found. SSSR can partly meet this requirement; however, it cannot work on some other cases (see Table 7).

SSSR may partly include the PARS (potential aromatic ring set); however it excludes PARS but collects the rings which are not PARS. An example is shown in Figure 22.

As shown in Figure 22, atoms **a** and **b** have no sp^2 electrons. If SSSR match algorithm excludes these atoms, PARS should be found. Therefore, PARSA has three steps as shown in Figure 23.

The PARSA algorithm needs find all ring bonds algorithm (FARBA), which is described as following:

```
Algorithm Algorithm_FARBA(molecule g, ring_bond_table rbt)
{
   for (i=the first bond to the last bond of the structure g)
   {
     bc=bond_color_of(i);
     cut_bond(i);
     if (disconnectivity_increase(g))
     rbt <= i; /* i is a ring bond*/
     recover_bond(i, bc);
   };
} /* end of Algorithm FARBA*/
```

By isolating ring atoms without sp^2 electrons, the degree of a ring atom is reduced, and, therefore, SSSRA becomes faster for GMA's computing complexity relies on the adjacent degree.

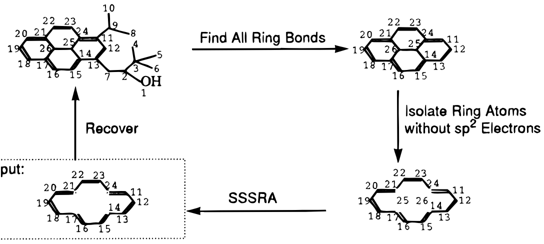


Figure 23. The PARSA flow chart.

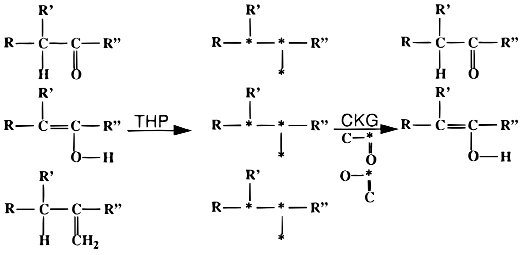


Figure 24. Topological homomorphic procedure and checking key groups for tautomer detection.

7. Tautomer Detection Algorithm (TDA). Tautomer search is difficult in other methods because it requires matching two different color graphs, i.e., keto and enol, or keto and phenol, or nitroso and oxime, or aliphatic nitro and aci, or imine and enamine.[37] TDA can be easily implemented by calling GMA, setting topological homomorphic procedure (THP), and checking key groups (CKG) as shown in Figure 24.

The TDA algorithm is described as follows:

```
Algorithm Algorithm_TDA(molecule g1, molecule g2)
{
   if (topological_homomorphic_query(g1, g2)) /*Call GMA here*/
   {
     /*Also call GMA*/
     if ((is_substructure(g1, C-*=O)||(is_substructure(g1, C=*-O)) &&
       ((is_substructure(g2, C-*=O)||(is_substructure(g2, C=*-O))
     return "g1 and g2 are of tautomerism";
     else   "g1 and g2 are not of tautomerism";
   };
} /* end of Algorithm TDA*/
```

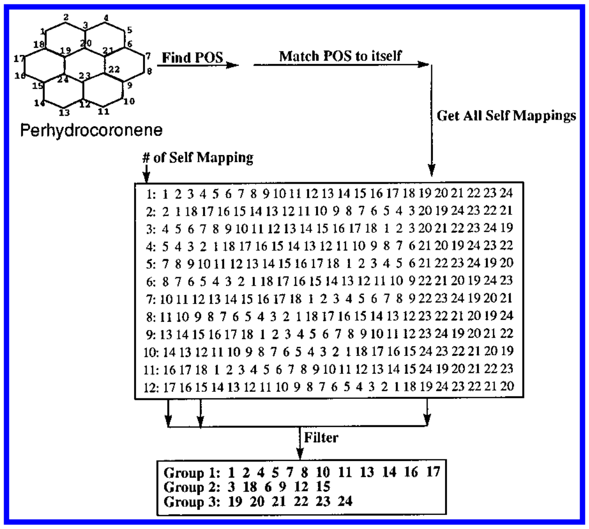
8. Structural Symmetry Detection Algorithm (SSDA). Molecule structural symmetry test is very important for structure elucidation. With this test, the number of ^{13}C and ^{1}H NMR spectral peaks, the intensity of the peaks, and the splitting patterns can be predicted. The performance of a spectral prediction algorithm can be enhanced for prediction chemical shifts for chemically equivalent nuclei.

Based upon POS (partial order set) methodology, SSDA becomes very simple and efficient. The algorithm is listed as following:

```
Algorithm SSDA(molecule g)
{
   find_POS (g);
   back-tracking on g constrained by the POS;
   get all self mapping(s);
   filter the mappings to group equivalent atoms;
} /* end of Algorithm SSDA */
```

The data flow of SSDA is shown in Figure 25.

It is interesting that the similar molecules can have very different symmetries. For example, if one of the carbons of perhydrocoronene is replaced by any other atom or substi-



1: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
2: 2 1 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 20 19 24 23 22 21
3: 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 1 2 3 20 21 22 23 24 19
4: 5 4 3 2 1 18 17 16 15 14 13 12 11 10 9 8 7 6 21 20 19 24 23 22
5: 7 8 9 10 11 12 13 14 15 16 17 18 1 2 3 4 5 6 21 22 23 24 19 20
6: 8 7 6 5 4 3 2 1 18 17 16 15 14 13 12 11 10 9 22 21 20 19 24 23
7: 10 11 12 13 14 15 16 17 18 1 2 3 4 5 6 7 8 9 22 23 24 19 20 21
8: 11 10 9 8 7 6 5 4 3 2 1 18 17 16 15 14 13 12 23 22 21 20 19 24
9: 13 14 15 16 17 18 1 2 3 4 5 6 7 8 9 10 11 12 23 24 19 20 21 22
10: 14 13 12 11 10 9 8 7 6 5 4 3 2 1 18 17 16 15 24 23 22 21 20 19
11: 16 17 18 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 24 19 20 21 22 23
12: 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 18 19 24 23 22 21 20

Group 1: 1 2 4 5 7 8 10 11 13 14 16 17
Group 2: 3 18 6 9 12 15
Group 3: 19 20 21 22 23 24

Figure 25. The data flow of algorithm SSDA.

tuted, then all of the 24 carbon atoms will be structurally different from each other.

SSDA is also used to enhance the performance of a number of other GMA-based algorithms. For example, NMR spectra prediction algorithm (using additive rules) needs only to calculate chemical shifts for atom 1, atom 3, and atom 19 for perhydrocoronen. When SSDA is used in SSSRA and PARSA, these algorithms do not have to test structural equivalent atoms, and the computing complexities are reduced substantially.

CONCLUSIONS

GMA is a backtracking class of algorithm. It characteristics can be highlighted as follows:

(1) GMA considers "query structure data" as a "program", which can be executed on a "queried structure".

(2) GMA's computing complexity is exponential instead of factorial. The computing complexity more relies on node out-degrees.

(3) Traditionally, MCSS algorithm has factorial of factorial computing complexity; however, GMA can handle MCSS with exponential computing complexity. With SSDA algorithm, the computing complexity can be further reduced.

(4) Many previously complicated structure perception algorithms can be simplified by means of GMA and partial ordering concept, and the performances of these algorithms have been significantly improved.

(5) GMA is parallelizable. In a parallel computing architecture, partial ordering searches can be carried out from a number of starting nodes at the same time. It is meaningful for a larger database and many compounds with high degree (>5) nodes.

ACKNOWLEDGMENT

Dr. B. A. Woods is acknowledged for valuable scientific and editorial advice in preparing the manuscript. The author is also indebted to Dr. Rodney Farlee, Mr. Richard Morris, and Mr. Daniel Sears for the help in the implementations and tests of all the algorithms presented in this paper.

REFERENCES AND NOTES

(1) Garey, M. R.; Johnson, D. S. *Computers and Intractability: a Guide to the Theory of NP-Completeness*; W. H. Freeman: San Francisco, 1979.

(2) Harel, D. *Algorithmics: the Spirit of Computing*; Addison-Wesley: Reading, MA, 1987.

(3) Ullman, J. R. An Algorithm for Subgraph Isomorphism. *J. Assoc. Comput. Machinery* **1976**, *23*, 31−42.

(4) *Chemical Structures 2*; Warr, W. A., Ed.; Springer-Verlag: New York, 1993.

(5) Willett, P. *Three-Dimensional Chemical Structure Handling*; John Wiley & Sons: New York, 1991.

(6) Wilson, T. *Chemical Searching on an Array Processor*; John Wiley & Sons: New York, 1993.

(7) Barnard, J. M. Substructure Searching Methods: Old and New. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 532−538.

(8) Levi, G. A Note on the Derivation of Maximal Common Subgraphs of Two Directed or Undirected Graphs. *Calcolo* **1972**, *9*, 341−352.

(9) Barrow, H. G.; Burstall, R. M. Subgraph Isomorphism, Matching Relational Structures and Maximal Cliques. *Information Process. Lett.* **1976**, *4*, 83−84.

(10) Johnson, M. A.; Maggiora, G. M. *Concepts and Applications of Molecular Similarity*. John Wiley: New York, 1990.

(11) Crandell, C. W.; Smith, D. H. Computer-Assisted Examination of Compounds for Common Three-Dimensional Substructures. *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 186−197.

(12) McGregor, J. J.; Willet, P. Use of a Maximal Common Subgraph Algorithm in the Automatic Identification of the Ostensible Bond Changes Occurring in Chemical Reactions. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 137−140.

(13) Willet, P. *Modern Approaches to Chemical Reaction Searching;* Aldershot: Gower, 1986.

(14) Chen, L.; Robien, W. MCSS: A New Algorithm for Perception of Maximal Common Substructures and Its Applications to NMR Spectral Studies. 1. The Algorithm. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 501−506.

(15) Plotkin, M. Mathematical Basis of Ring-finding Algorithms in CIDS. *J. Chem. Doc. Comput. Sci.* **1971**, *11*, 60−63.

(16) 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*(*2*), 460−465.

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

(18) Matyska, L. Fast Algorithm for Ring Perception. *J. Comput. Chem.* **1988**, *9*, 455−459.

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

(20) 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 (ESSR) Concept. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 187−206.

(21) 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 (ESSR) in Structurally Explicit Generics. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 207−214.

(22) Cahn, R. S.; Ingold, C.; Prelog, V. Specification of Molecular Chirality. *Angew. Chem., Int. Ed. Engl.* **1966**, *5*, 385−551.

(23) Prelog, V.; Helmchen, H. Basic Principles of the CIP-System and Proposals for a Revision. *Angew. Chem., Int. Ed. Engl.* **1982**, *21*, 567−583.

(24) Wipke, W. T.; Dyott, T. M. Simulation and Evaluation of Chemical Synthesis. Computer Representation and Manipulation of Stereochemistry. *J. Am. Chem. Soc.* **1974**, *96*, 4825−4834.

(25) Blackwood, J. E.; Elliot, P. M.; Stobaugh, R. E.; Watson, C. E. The Chemical Abstracts Service Chemical Registry System III. Stereochemistry. *J. Chem. Inf. Comput. Sci.* **1977**, *17*, 3−8.

(26) Shelley, C. A.; Munk, M. E. An Approach to the Assignment of Canonical Connection Tables and Topological Symmetry Perception. *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 247−250.

(27) Mockus, J.; Stobaugh, R. E. Chemical Abstracts Service Chemical Registry System. 7. Tautomerism and Alternating Bonds. *J. Chem. Inf. Comput. Sci.* **1980**, *20*, 18−22.

(28) Ray, L. C.; Kirsch, R. A. Finding Chemical Records by Digital Computers. *Science* **1957**, *126*, 814−819.

(29) Xu, J.; Zhang, M. HBA: New Algorithm for Structural Match and Applications. *Tetrahedron Comput. Methodol.* **1989**, *2*, 75−83.

(30) Dengler, A.; Ugi, I. A Central Atom Based Algorithm and Computer Program for Substructure Search, *Comput. Chem.* **1991**, *15*, 103−107.

(31) Sussenguth, E. H. A Graph-Theoretic Algorithm for Matching Chemical Structures. *J. Chem. Doc.* **1965**, *5*, 36−43.

(32) Figueras, J. Substructure Search by Set Reduction. *J. Chem. Doc.* **1972**, *12*, 237−244.

(33) Von Scholley, A. A Relaxation Algorithm for Generic Chemical Structure Screening. *J. Chem. Inf. Comput. Sci.* **1984**, *24*, 235−241.

(34) Balaban, A. T., Ed.; *Chemical Applications of Graph Theory*; Academic Press: New York, 1976; Chapter 11.

(35) Pellegrin, V. *J. Chem. Educ.* **1983**, *60*, 626.

(36) March, J. *Advanced Organic Chemistry*, 4th ed.; John Wiley & Sons: New York, 1992; 94−123.

(37) March, J. *Advanced Organic Chemistry*, 4th ed.; John Wiley & Sons: New York, 1992; 69−73.

CI950061U