EMCSS: A New Method for Maximal Common Substructure Search

Ting Wang and Jiaju Zhou*

Laboratory of Computer Chemistry (LCC), Institute of Chemical Metallurgy, Chinese Academy of Sciences,
P.O. Box 353, Beijing 100080, China

Received December 4, 1996[⊗]

This paper describes a new method-EMCSS for Maximal Common Substructure (MCS) search, which uses a substructure searching algorithm: Xu's GMA algorithm converts the MCS search space into a much smaller space, the connection table space of query graph (QG), and adopts a evolutionary strategy to search the optimum solution. The principle of the EMCSS method and its implementation are described in detail. Some highly complex examples, even a hyperstructure pair, are tested. The investigation demonstrates that the EMCSS method is robust and efficient.

1. INTRODUCTION

The Maximal Common Substructure (MCS) of a pair of structures is the largest substructure that is present in both structures and is measured by the number of nodes or edges in the substructure. The MCS search has been applied to such areas of chemistry as recognition of reaction centers,[1] NMR and mass spectral studies,[2,3] quantitative structure−activity relationships (QSAR) studies,[4−6] and molecular similarity searching.[7] In the drug design arena, the MCS approach was also used for identifying the topological pharmacophores[8] and 3D pharmacophores[9,10] from a set of active and inactive molecules.

Our interest in the MCS algorithm originated from studies on effective components in Chinese medicines. We attempt to search if certain common substructures exist among the compounds exhibiting the same biological activity in various Chinese medicines. Many such compounds have very complex structures such as multiple cycles and high connectivity. Accordingly, a robust and effective algorithm that can perceive the MCSs of various structure pairs is essential.

The MCS problem is an NP-complete problem and so has no polynomial time solution. Simply enumerating all possible MCSs is time-consuming, especially for large and complex structure pairs. For the two given structures containing m and n nodes, respectively, the maximal number of possible node-by-node comparisons for the identification of all the common substructures containing k nodes is[11]

$$\frac{m!n!}{(m-k)!(n-k)!k!}$$

In order to overcome this computational complexity, most MCS algorithms[1,11−14] have used various strategies to prune the MCS search tree, thereby reducing the search space and increasing the search speed. However, it is still difficult for these deterministic algorithms to deal with very complex structure pairs in reasonable time.[15,16,23] Brown[16] reported that a genetic algorithm can be successfully applied to another analogous NP-complete problem: the identification of the maximal overlap sets (MOS) between a structure and a hyperstructure. But Brown's algorithm cannot reduce the search space.

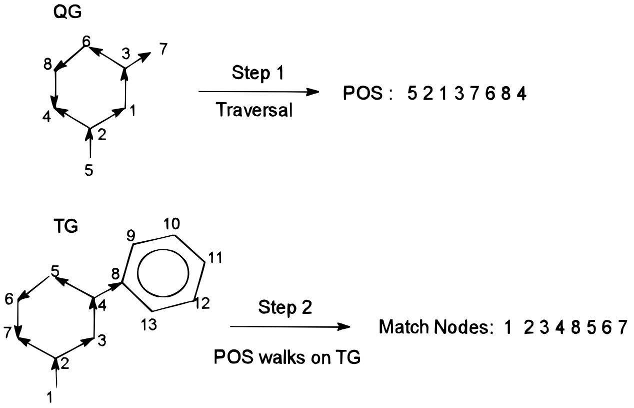


Figure 1. The principle of GMA algorithm.

This paper describes a novel MCS method, EMCSS, which combines a substructure searching algorithm, Xu's GMA algorithm,[17] with an evolutionary strategy. In addition, this paper discusses the comparison between the EMCSS method and Brown's algorithm.

2. ABOUT THE GMA ALGORITHM

The GMA algorithm[17] is the basis of our EMCSS method, and so it is necessary to outline its principle before describing our EMCSS method.

The GMA algorithm is a partial-ordering-based back-tracking substructure searching algorithm. It consists of the following two steps:

Step 1. According to its input connection table, reorder the Query Graph (QG) by a Graph Traversal Algorithm (GTA) to get a Partial Order Set (POS) of QG.

Step 2. Use POS as a instruction set to walk on Target Graph (TG), this is a Constrained Back-tracking Algorithm (CBA). If the walk is complete, then QG and TG are homomorphic or isomorphic.

Figure 1 shows the principle of GMA. Step 1 is a traversal procedure, the resulting POS is a traversal route on QG. Table 1 is the input connection table of QG, then the POS starting from node 5 is 5 2 1 3 7 6 8 4. By using this POS as a instruction to walk on TG, step 2 will output that QG is a substructure of TG and the match node set in TG is 1 2 3 4 8 5 6 7.

So, the GMA algorithm is also called directed match algorithm. The match route in the GMA algorithm is clearer

[⊗] Abstract published in *Advance ACS Abstracts,* August 1, 1997.

S0095-2338(96)00167-9 CCC: $14.00

Table 1. The Input Connection Table of QG in Figure 1

	adjacent node		
node	first node	second node	third node
1	2	3	
2	1	4	
3	1	7	6
4	2	8	
5	2		
6	5		
7	3		
8	6	4	

and more perceptual than that in the Ullmann algorithm[18] used in many commercial substructure search systems. Moreover the GMA's computing complexity is much less than the factorial computing complexity. (More details are provided by ref 17).

3. THE MCS PROBLEM DEFINED BY GMA ALGORITHM

In the homomorphism or isomorphism problem, if TG and QG are homomorphous or isomorphous, the same POS should be extracted from the TG and any POS of QG can be used as the instruction set to walk on the TG. However, in the MCS problem, the situation is much more complicated, because different POSs may give different MCSs of TG and QG. Therefore, in order to obtain the real MCS, all POSs of QG have to be tried.

Since starting from any node to order the QG will give a POS, a QG containing *N* nodes (non-hydrogen atoms) will yield *N* POSs when the connection table of QG is fixed. However, by trying the *N* POSs, we can only obtain the MCS mapping(s) resulting from the input connection table of QG but probably not the real MCS mapping(s), because the N POSs are not all of the POSs of QG unless there is no multiple branch node in QG. The POSs of other connection tables of QG also have to be tried.

Taking the QG and TG in Figure 2 as a example, Tables 2 and 3 are their input connection tables, respectively. The MCSs from the input connection table of QG are shown in Figure 4. Obviously it is not the real MCS (Figure 5) of QG and TG.

Table 2. The Input Connection Table of QG

	adjacent nodes					adjacent nodes			
node	first node	second node	third node	fourth node	node	first node	second node	third node	fourth node
1	2	3			16	15	18	20	
2	1	4			17	15	24		
3	1	5	7		18	16	19		
4	2	6			19	18			
5	3	6	9		20	16			
6	4	5			21	8			
7	3	8	10	22	22	7	23		
8	7	9	21		23	12	22		
9	5	8			24	17	25		
10	7	11	12		25	24			
11	10	28			26	14	27		
12	10	13	23		27	26			
13	12	13			28	11	14	15	
14	13	26	27						
15	17	20	28						

Table 3. The Input Connection Table of TG

	adjacent nodes					adjacent nodes			
node	first node	second node	third node	fourth node	node	first node	second node	third node	fourth node
1	2	3			16	15	18	20	
2	1	4			17	15	24		
3	1	5	7		18	16	19		
4	2	6			19	18			
5	3	6	9	20	16				
6	4	5			21	8			
7	3	8	10	22	22	7	23		
8	7	9	21		23	12	22		
9	5	8			24	17	25		
10	7	11	12		25	24			
11	10	28			26	14	27		
12	10	13	23		27	26			
13	12	13			28	11	14	15	
14	13	26	27						
15	17	20	28						

There are many ways to order the adjacent edges of a multiple branch node (adjacent degree > 2) of a QG, and each way is one connection table of QG. Different connection tables show different sequences of the adjacent edges of multiple branch nodes but maintain the same connectivity between nodes.

In the structure shown in Figure 3, Tables 4 and 5 are its two different connection tables, the difference is that the sequence of the adjacent edges of multiple branch node 5 is different. Though they contain the equivalent QG structure information, they will yield two different POSs with the same

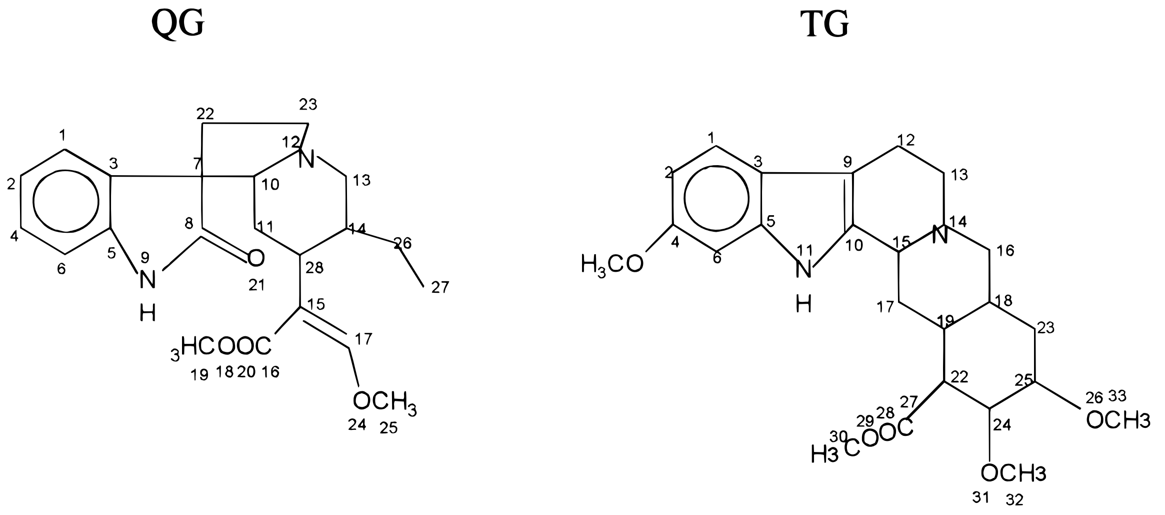


Figure 2. QG and TG.

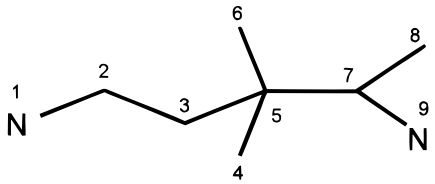


Figure 3. An example QG.

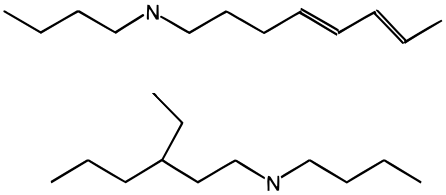


Figure 4. Two MCS mappings from the input connection table of QG and TG in Figure 2.

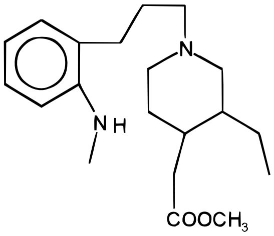


Figure 5. The MCS mapping from the EMCSS method between QG and TG in Figure 2.

Table 4. A Connection Table of the Example QG

	adjacent node			
node	first node	second node	third node	fourth node
1	2			
2	1	3		
3	2	5		
4	5			
5	3	4	6	7
6	5			
7	5	8		
8	7			
9	7			

Table 5. One Connection Table of Example QG

	adjacent node			
node	first node	second node	third node	fourth node
1	2			
2	1	3		
3	2	5		
4	5			
5	7	3	6	4
6	5			
7	5	8	9	
8	7			
9	7			

starting node, e.g., node 1:

POS1: 1 2 3 5 4 7 8 9 6 from Table 4

POS2: 1 2 3 5 7 8 9 4 6 from Table 5

The sum of the connection tables of QG lies in the number of multiple branch nodes and the adjacent degree of each multiple branch node equal to the product of adjacent degrees of all multiple branch nodes. So, the structure shown in Figure 3 involves 3 × 4 connection tables (=12).

$$\textit{sum of connection tables of QG} = \prod_{i}^{num} \textit{degree of multiple branch node}_i$$

where *num* is the number of multiple branch nodes of QG. Thus, the MCS search space is converted into the connection table space of QG, and the MCS problem is converted into the problem of searching the optimum connection table of QG. In general, the connection table space of QG is not large, because most chemical structures have limited nodes with three or four adjacent edges. In addition, the space which must be explored is much smaller, because many different connection tables will give the same MCS mappings.

Our EMCSS method adopts an evolutionary algorithm in which only mutation operations are used to search such a small space.

4. EMCSS—A NEW METHOD FOR MCS SEARCH

4.1. Encoding the Problem. In our EMCSS method, the connection table of QG is encoded as a bit string. The bit string of QG containing *N* nodes consists of *N* fragments. A fragment has *n* bits if the relative node has *n* adjacent edges and each integer in the bit string is the label of an adjacent edge of the relative node according the input connection table of the QG. So, a bit string of QG consists of *M* integers:

$$M = \sum_{i=1}^{N} \text{adjacent degree of node}_i$$

where *N* is the node number of QG.

For the input connection table of any QG, the labels of the adjacent edges of a node with *n* adjacent edges are 1,2,3,...,*n*, respectively. Then the relative fragment of the bit string of the input connection table is 123...*n*, namely, *n* increasing integers.

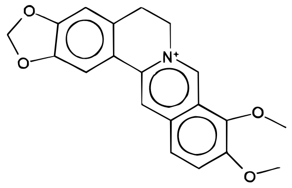
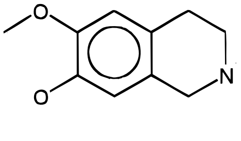
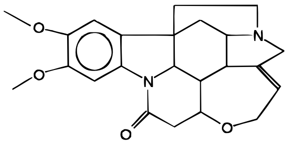
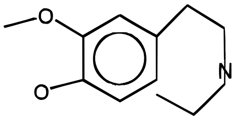
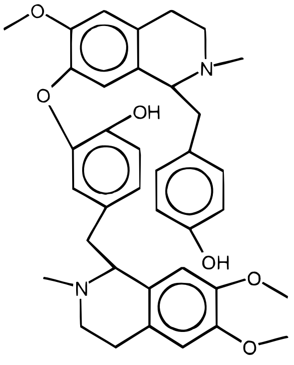
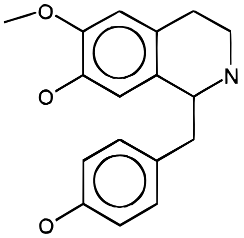
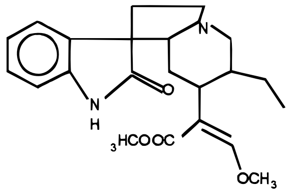
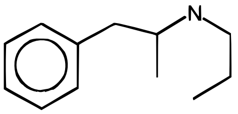
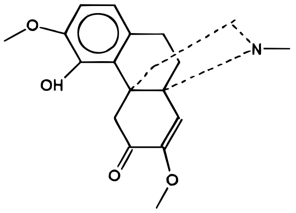
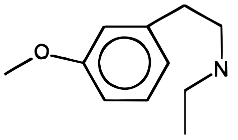
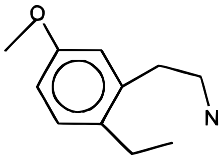
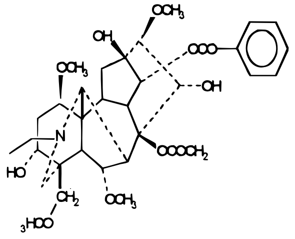
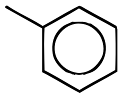
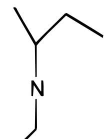
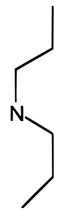
For example, suppose Table 4 is the input connection table of the structure in Figure 3, then its bit string is 1 12 12 1 1234 1 123 1 1. Because the fourth edges of node 5 in the input connection table is changed into the first edges in Table 5, second into fourth and first into second, the bit string of Table 5 is 1 12 12 1 4132 1 123 1 1. The difference of the two bit strings is only in that the positions of bits of the fifth fragment are different. That is to say, changing the positions of bits of fragments of the bit string of the input connection table may result in many different connection tables. Therefore, the space of the connection table can be simply presented by different bit strings.

4.2. The Fitness Function. Each connection table will give rise to an MCS mapping(s). The number of edges of MCS is used as the fitness value of the connection table.

By using the GMA algorithm, the process for a connection table to give a MCS mapping is as follows:

1. Select a node of QG as the starting node to get a POS.
2. According to the connection table of QG, get a POS by traversing on QG.
3. Select a node of TG as the starting node to walk on TG.

Table 6. Results of Six Example Runs, QG Is Coclaurine Showed in Figure 6

TG	Run Time	MCS(s)
 Berberine	2.18s	 (ignoring the charge)
 Brucine	2.98 s	
 Liensinine	1.96 s	
 Rhynchophylline	1.84 s	
 Sinomenine	1.82 s	 
 Aconitine	4.36 s	  

4. Execute POS on TG and record the longest match route(s).
5. Return to step 3 until each node of TG has been used as the starting node.
6. Return to step 1 until each node of QG has been used as the starting node.

Then the longest match route(s) is the MCS mapping(s) from the connection table of QG, and the edge number of MCS is the fitness value of the connection table.

4.3. Mutation. At the start of the EMCSS method, according to the input connection table of QG, many diverse bit strings representing the diverse connection tables of QG are created randomly and are called an initial population. The fitness value of each string is calculated, and then from the first string to the last string of the population, the fitness of each string is compared with the fitness of a randomly created new string (different from any string of the population). If the latter is larger, the new string replaces the old one, otherwise, the old string is kept. This operation is an individual mutation. When all the strings in the population have been processed by the mutation operation, a new generation is created with larger average fitness. Generation by generation, better strings which give a larger MCS will be generated until the maximal generation number GENT is reached.

4.4. Dynamic Parameter Sets. Since the size of the search space is mainly determined by the number of the nodes with more than two adjacent edges of QG, the population size is then set dynamically. Let *num* be the number of the nodes with more than two adjacent edges of QG: the population size is then defined as $2 \times num - 1$.

The maximal generation number GENT is also determined by the number *num*. In general, if the $num \leq 10$, an optimum solution can be obtained even though GENT is 1. So, GENT is defined as $(num-1)/10 + 1$.

The parameter sets above, illustrate that the EMCSS method is different from the general genetic algorithm[16,21] or evolutionary algorithm.[19,20] In those algorithms, the population sizes and the maximal generation numbers are usually set to hundreds or even thousands.

4.5. Termination Conditions. The EMCSS will terminate in two statuses:

1. When the fitness of any string is equal to the number of edges of QG, QG is the substructure of TG.

2. When the maximal generation number GENT is reached, output all the MCS mappings resulting from the string(s) with largest fitness.

4.6. Simplest Case. If there is no multiple branch mode in QG, the input connection table of QG is the only connection table. This case is the simplest and the run time is shortest, because there is no need to search the connection table space. The MCS mappings can be obtained by calculating only the fitness value of the input connection table of QG.

4.7. Example Run. The EMCSS method is programmed in C and runs on an IBM PC Pentium/90MHZ. The EMCSS is run on the example shown in Figure 2. Both QG and TG are complex structures with four and five cycles, respectively, and many nodes have three even four adjacent edges. The number of the multiple branch nodes in QG is 10, so the population size is 19 and GENT is 1. The MCS mappings showed in Figure 5 is obtained in 6.8 s.

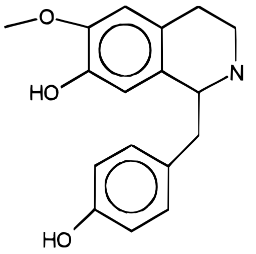


Figure 6. Structure of coclaurine.

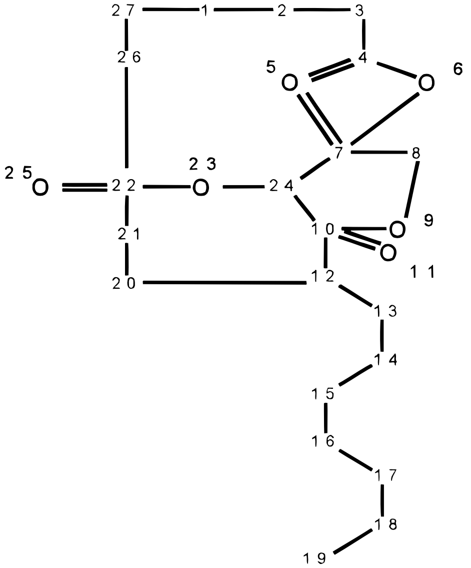


Figure 7. Hyperstructure from Figure 8, with the exception of structure 22380.

The EMCSS has also been tested on a small structure database of 55 complex alkaloid compounds which are effective components in many antihypertensive Chinese medicines. Table 6 lists six run examples and their results while taking the coclaurine as QG showed in Figure 6. The number of multiple branch nodes in QG is 7, and so the population size is 13 and GENT is 1. Both QG and six TGs are complex structures, and the EMCSS gives their MCSs correctly. It can be also seen from Table 6 that the run time increases from 1.82 to 4.36 s with the increasing complexity of TG from sinomenine to the highly complex aconitine while the QG is the same. This illustrates that the run time is also influenced by the complexity of TG though the search space is determined by QG.

4.8. Comparison with Brown's Algorithm. In principle, the EMCSS method can deal with any structure pair, and there is no limitation on the size and complexity of the structures considered. In order to test the robustness and efficiency of the EMCSS method, it was applied to search the MCS between a structure and a hyperstructure, and the result was compared with that of Brown's algorithm.[16] There is no exiting deterministic algorithm, including conventional relaxation or backtrack algorithm,[12,13] capable of solving this problem in any realistic time.

A hyperstructure is a single structure representation for many structures that was first suggested by Vladutz and Gould[22] and is formed by superimposition of a set of molecules. The hyperstructure concept has been applied to molecular similarity searching and to speed up substructure searching of large chemical database.

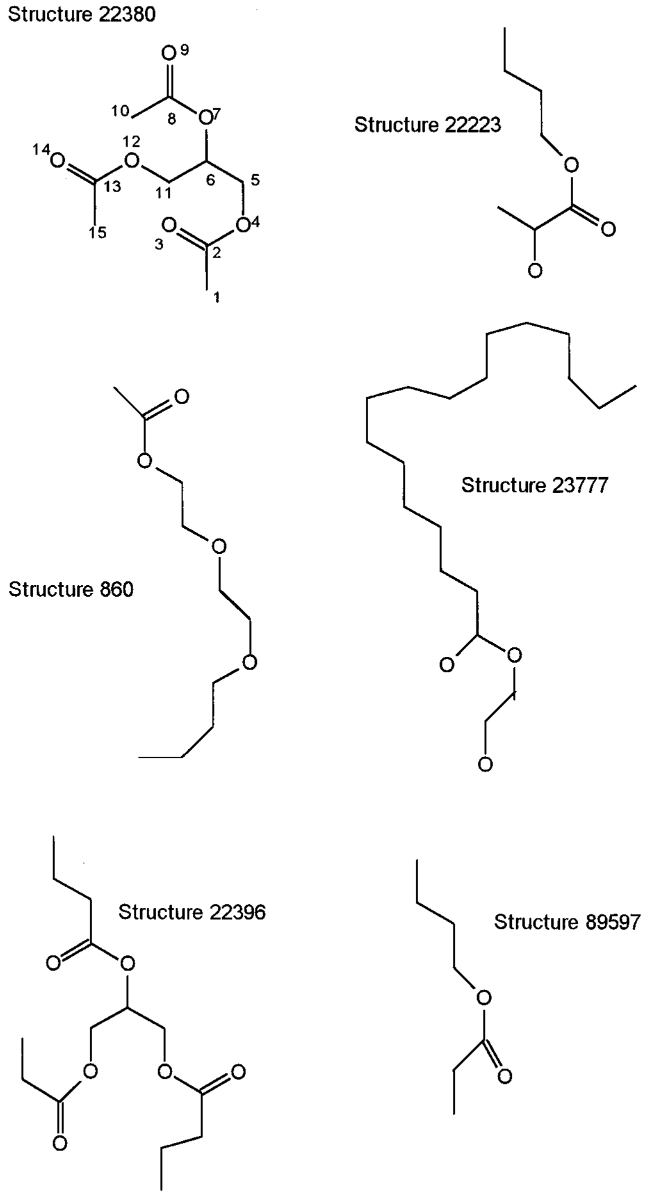


Figure 8. Example structures from ref 16.

A hyperstructure tends to be much larger and very much more highly connected than conventional molecules, and hyperstructure atoms also do not obey the normal rules of valency. Figure 7 is a hand-generated hyperstructure comprising all of the molecules shown in Figure 8 with the exception of structure 22380.

The EMCSS was run on the hyperstructure (Figure 8) and the structure 22380. The run time is 0.22 s on an IBM PC Pentium/90MHZ, and two nonaboundant mappings are obtained:

[structure 22380, hyperstructure]

1. [1,12][2,10][3,11][4,9][5,8][6,7][7,6][8,4][9,5][10,3][11,24][12,23][13,22][14,25][15,26]
2. [1,12][2,10][3,11][4,9][5,8][6,7][7,6][8,4][9,5][10,3][11,24][12,23][13,22][14,25][15,21]

According to ref 16, the run time of Brown's algorithm is 0.98 s on a SUN 4/470, and the mapping is

[1,12][2,10][3,11][4,9][5,8][6,24][7,23][8,22][9,25][10,21][11,7][12,6][13,4][14,5][15,3]

and Brown noted that "This result was considered very promising in that there is no known deterministic algorithm that is capable of generating this mapping within any reasonable amount of time".

The mapping from Brown's algorithm is equivalent to the second mapping of the result of the EMCSS and they are two abundant mappings. It is difficult to exactly compare CPU times between different programs on different machines and written in different languages, but the EMCSS method has the advantage of obtaining all the MCS mappings of a highly complex structure pair in a satisfactory time.

5. CONCLUSION

By using a backtrack substructure searching algorithm, the EMCSS method developed here converts the MCS problem into the problem of searching the optimum connection table of QG and thus greatly reduces the MCS search space. By virtue of a evolutionary strategy, the optimum solution can be obtained in a satisfactory time. Example runs and comparison with Brown's algorithm demonstrate that the EMCSS method is a more robust and efficient method to deal with complicated MCS problem, especially for highly complex structure pair.

ACKNOWLEDGMENT

We are grateful to Dr. Xu Jun of the BIO-RAD Laboratory for helpful discussion during program development.

REFERENCES AND NOTES

(1) McGregor, J. J.; Willett, 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.

(2) Chen, L.; Robien, W. MCSS: A New Algorithm for Perception of Maximal Common Substructures and Its Applications to NMR Spectral Studies. 2. The Applications. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 507−510.

(3) Cone, M. M.; Venkataraghavan, R.; McLafferty, F. W. Molecular Structure Comparison Program for the Identification of Maximal Common Substructures. *J. Am. Chem. Soc.* **1977**, *99*, 7668−7671.

(4) Yuan, S.; Zheng, C.; Zhao, X.; Zeng, F. Identification of Maximal Common Substructures in Structure/Activity Studies. *Anal. Chim. Acta.* **1990**, *235*, 239−241.

(5) 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.

(6) Barakat, M. T.; Dean, P. M. Molecular Structure Matching by Simulated Annealing III. The Incorporation of Null Correspondences into the Matching Problem. *J. Comput.-Aided Mol. Design* **1991**, *5*, 107−117.

(7) Hagadone, T. R. Molecular Substructure Similarity Searching: Efficient Retrieval in Two-Dimensional Structure Databases. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 515−521.

(8) Downs, G. M.; Gill, G. S.; Willett, P.; Walsh, P. *Automated Descriptor Selection and Hyperstructure Generation to Assist SAR Studies. SAR and QSAR in Environmental Research*; Vol. 3, pp 253−264.

(9) Martin, Y. C.; Bures, M. G.; Danaher, E. A.; Delazzer, J., Lico, I.; Pavlik, P. A. A Fast New Approach to Pharmacophore Mapping and Its Application to Dopaminergic and Benzodiazepine Agonists. *J. Comput.-Aided Mol. Design* **1993**, *7*, 83−102.

(10) Bures, M. G.; Danaher, E.; Delazzer, J.; Martin, Y. C. New Molecular Modelling Tools Using Three-dimensional Chemical Substructures. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 218−223.

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

(12) McGregor, J. J. Backtrack Search Algorithms and the Maximal Common Subgraph Problem. *Software-Pract. Exper.* **1982**, *12*, 23−34.

(13) Bayada, D.; Simpson, R. W.; Johnson, A. P. An Algorithm for the Multiple Common Subgraph Problem. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 680−685.

(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) Brown, R. D.; Downs, G. M.; Willett, P.; Cook, A. P. F. A Hyperstructure Model for Chemical Structure Handling: Generation and Atom-by-Atom Searching of Hyperstructures. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 522−531.

(16) Brown, R. D.; Jones, G.; Willett, P. Matching Two-Dimensional Chemical Graphs Using Genetic Algorithms. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 63−70.

(17) Xu, J. GMA: A Generic Match Algorithm for Structural Homomorphism, Isomorphism, and Maximal Common Substructure Match and Its Applications. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 25−34.

(18) Ullmann, J. R. An Algorithm for Subgraph Isomorphism. *J. Assoc. Comput. Mach.* **1976**, *23*, 31−42.

(19) Kubinyi, H. Variable Selection in QSAR Studies. 1. An Evolutionary Algorithm. *Quant. Struct.-Act. Relat.* **1994**, *13*, 285−294.

(20) Fogel, D. B. Applying Evolutionary Programming to Selected Traveling Salesman Problems. *Cybern. Syst. (USA)* **1993**, *24*, 27−36.

(21) Forrest, S. Genetic Algorithms: principles of natural selection applied to computation. *Science* **1993**, *261*, 872−878.

(22) Vladutz, G.; Gould, S. R. Joint Compound/Reaction Storage and Retrieval and Possibilities of a Hyperstructure-Based Solution. In *Chemical Structures. The International Language of Chemistry*; Warr, W. E., Ed.; Springer Verlag: Berlin, 1988; pp 371−384.

(23) Okada, T.; Wipke, W. T. CLUSMOL: a System for the Condeptual Clustering of Molecules. *Tetrahedron Comput. Methodol.* **1989**, *2*, 249−264.

CI9601675