MCSS: A New Algorithm for Perception of Maximal Common Substructures and Its Application to NMR Spectral Studies. 1. The Algorithm

Lingran Chen† and Wolfgang Robien*

Department of Organic Chemistry, University of Vienna, Währingerstrasse 38, A-1090 Vienna, Austria

Received March 24, 1992

A new algorithm has been developed which can be used to deal with structure isomorphism, substructure searching, and Maximal Common SubStructure (MCSS) detection. The algorithm described can perceive topological or colored maximal common substructures of two given structures, including structures which consist of two or more disconnected parts. The new strategies used in the algorithm are discussed in detail. The program was tested upon a set of carefully selected structure pairs and then on a database containing about 80 000 structures. The investigation revealed that the algorithm has high efficiency and flexibility. The MCSS algorithm has been implemented into the CSEARCH-NMR database system allowing automatic increment analysis and error detection.

INTRODUCTION

Structure isomorphism identification, substructure detection, and Maximal Common SubStructure (MCSS) perception are three important procedures in computer manipulation of structural information. In the past, those three procedures were usually studied individually, and some excellent algorithms have been reported for both structure and substructure isomorphism identifications. However, although as early as 1967, the methods for the MCSS problem were studied,[1,2] the progress seems somewhat slow.[3-11] This is mainly because of the greater complexity of the MCSS problem itself. A brief analysis of the internal relationship between the MCSS problem with structure and substructure isomorphism problems can help us understanding the difficulty of the MCSS problem better. A MCSS problem is defined as the following: Determine from two (or more) given structures the common substructure set that contains the largest possible number of bonds. If this maximal common substructure is isomorphic to the smaller structure, this kind of MCSS problem is, in fact, a substructure isomorphism problem. If the maximal common substructure found is isomorphic to both of the given structures, this problem belongs to the structure isomorphism problem. Therefore, the structure and substructure isomorphism problems are only two special cases of the more general MCSS problem.

The structure isomorphism problem for general structures is not easy to solve, but substructure isomorphism problems are, however, much more complicated than structure isomorphism problems. It has been proved that the substructure isomorphism problem belongs to the so-called NP-complete problems. The NP-complete problems have the property that if any one of them has a polynomial-time algorithm, they all do. Since no one has discovered a polynomial-time algorithm for any of these problems, it seems likely that none of these problems is solvable in polynomial time.[12]

Although structure and substructure isomorphism problems are difficult to deal with, there are, anyway, some known definite conditions which can be used to guide the search process. For example, for the structure isomorphism problem, an atom with n attached neighbors in one structure must be matched to any atom with exactly the same number of direct neighbors in the other structure. For the substructure isomorphism problem, an atom with n neighbors in the query structure must be matched to atoms with m ($m \geq n$) direct neighbors in the second structure. Furthermore, the number of atoms and/or bonds of the query structure can be used to decide about the continuation of the search tree. Applying these conditions greatly enhances the efficiency of the algorithm. For the general MCSS problem, however, none of these conditions can be used. The main difficulty of the general MCSS problem is just here.

MCSS approaches have found their applications in some areas of chemistry, such as recognition of reaction centers, identification of key intermediates in synthesis planning, and perception of common structural features for structure/activity studies. Recent works are all focused on these areas.[13-15] The application of the MCSS approach to the interpretation of unknown mass spectra has been reported.[9] The MCSS approach was also be used for deriving fairly large structural fragments, which are further processed by programs able to generate all possible isomers using given restrictions derived from spectral data.[16]

Our interest in the MCSS algorithm originated from studies of substituent effects on ^{13}C-NMR chemical shift values, which are important for structure elucidation. In these studies, it is necessary to compare directly the spectra of similar compounds to get the shift increments induced by different substituents. In order to solve this problem, an efficient algorithm that could perceive the MCSS of various structure pairs is essential. Most of the existing MCSS approaches were designed for some special applications and are not very efficient,[15] some of them can only deal with small or simple structures. Therefore, a novel MCSS algorithm was developed by using several new strategies which will be described in detail in this paper.

DESCRIPTION OF THE ALGORITHM

The whole MCSS algorithm consists of two parts: The first one is the basic MCSS algorithm for detection of maximal common substructures, and the second is designed to use the basic algorithm to deal with structures consisting of two or more disconnected parts.

The basic algorithm can further be divided into four parts: (a) quick determination of the initial value for backtracking condition,[17] (b) selection of starting pairs, (c) comparison of

† On leave from the University of Science and Technology of China, Hefei, Anhui 230026, The People's Republic of China.

two structures, and (d) selection of MCSS candidates. The output contains all the possible common substructures that hold the maximal number of bonds. The algorithm can deal with topological or colored graphs depending on the user's choice.

Seven key steps are implemented in the MCSS algorithm:

1. **Connectivity Analysis.** The connectivity of each structure is analyzed; structures which consist of two or more disconnected parts are recognized here.
2. **Structural Environment Calculation.** Structural environments of each atom of the two structures under investigation are calculated, and a symmetry analysis is performed.
3. **Isomorphism Detection.** For isomorphism problems, the algorithm is used to generate atom-by-atom correspondences between the query structures. As soon as one MCSS has been detected, the search is immediately stopped.
4. **Substructure Detection.** After all those atom pairs chosen by means of method 3 have been processed, a check on the obtained MCSS candidates is performed. If the largest candidate contains as many atoms as the smaller query structure, all MCSSs have been detected and the searching process ends.
5. **General MCSS Identification.** During this step, all additional starting pairs are selected in order to get all the possible MCSS candidates. The results obtained from steps 4 and 5 are combined and used for candidate selection.
6. **MCSS Candidate Selection.** Now, all of the MCSS candidates that contain the maximal number of bonds are collected, and duplicates are removed from this set, leading to the final solution.
7. **Disconnected Structure Handling.** If at least one structure consists of two or more disconnected fragments, each part is isolated from the original structure representation. Then, all the possible combination ways of these parts are calculated. According to each combination way, each such structural part from one structure is compared with each structural part of the other structure using the basic MCSS algorithm. All the intermediate results are collected and analyzed. One MCSS candidate set which corresponds to one combination way having the maximal total number of bonds is selected as the final solution.

BASIC MCSS: ALGORITHM

As noted above, the MCSS problem for general structures is usually very complicated to solve. For the two given structures containing m and n atoms, respectively, the maximal number of possible atom-by-atom comparisons for the identification of all the common substructures containing k atoms is[8]

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

It is obvious that trying to explore all the possibilities is too time-consuming except for small structures. The backtrack search technology used here is based on careful selection of a starting pair of atoms from the two structures under investigation. Systematic extension using neighbor atoms leads to larger structural fragments during the process of comparison until a terminating condition is fulfilled or a final solution to the matching problem is obtained. Though the basic backtrack search is better than the brute force method, the computation time of it is still exponential.[12] If, however, other techniques can be established for reducing the number of possibilities to be considered, the backtrack search provides an effective method for systematically considering the remaining possibilities.[7] High efficiency can only be achieved by sophisticated techniques for reducing the number and complexity of the search trees. The following discussion contains only those items where decisive new developments have been included.

(a) Selection of Starting Pairs. The method for choosing starting pairs is very important in designing a fast algorithm. A good method should be the one that can not only choose as few starting pairs as possible but can also guarantee no loss of any necessary one. In our algorithm, several methods for choosing starting pairs were designed to deal with the variety of structure pairs considered.

First, an atom-partitioning scheme is used to analyze the structural environments (EV) of atoms of each structure. EV values are calculated according to following equations:

$$\mathrm{EV}_{j(i,1)} = [\mathrm{NB}_{j(i)}]^2 \tag{1}$$

$$\mathrm{EV}_{j(i,L)} = \mathrm{EV}_{j(i,L-1)} + \sum_{k=1}^{\mathrm{NB}_{j(i)}} \mathrm{EV}_{j(\mathrm{CT}_{j(i,k)},L-1)} \tag{2a}$$

$$\mathrm{EV}_{j(i,L)} = \mathrm{EV}_{j(i,L-1)} + \sum_{k=1}^{\mathrm{NB}_{j(i)}} \mathrm{EV}_{j(\mathrm{CT}_{j(i,k)},L-1)} + \mathrm{ATTP}_{j(i)} + \mathrm{HYBR}_{j(i)} \tag{2b}$$

where j is the jth structure; i is the ith atom; k is the kth neighbor atom; $\mathrm{NB}_{j(i)}$, $\mathrm{ATTP}_{j(i)}$, and $\mathrm{HYBR}_{j(i)}$ are the number of neighbors, the atom types, and the hybridization, respectively; $\mathrm{CT}_{j(i,k)}$ is the connectivity table; and L is the Lth iteration cycle. Equation 2a shows topological structure matching; eq 2b shows colored structure matching.

This method bears some resemblance to the Morgan algorithm[18] and several other algorithms,[19] but with one important difference. In those algorithms, the number of non-hydrogen neighbor atoms (some algorithms may add another term, usually atom type multiplied by 10 to it) of the considered atoms is taken as the corresponding initial EV values. By means of this approach, it is even impossible to distinguish the carbons C-4 and C-1′ in ethylcyclohexane, though they have very different environments. In our method, the square of the number of non-hydrogen neighbors connected to each atom is used as the initial EV value. Therefore, even after the first iteration cycle, the above-mentioned problem has been solved. Further improvements of this simple and efficient partitioning algorithm will be discussed in more detail in ref 20.

The information of atom symmetry within one structure, structural isomorphism, or structural similarities can be deduced from the two EV lists of the query structures under investigation and can be further used to select starting pairs for MCSS detection.

Now the methods for selection of starting pairs are summarized below.

Method 1: The atom correspondency of two isomorphic structures is calculated from one starting pair, one partner of which shares the fewest same EV values with other atoms within the first structure. Another starting atom from the second structure is the one whose EV value is equal to that of the already-chosen starting atom of the first structure. These two atoms comprise the unique starting pair for the isomorphism problem.

Method 2: This method was designed to handle very similar structure pairs. The two query structures containing AT1 and AT2 atoms, respectively, have to satisfy both eqs 3 and 4:

$$\mathrm{EV1}_{(i,p+1)} \quad .\mathrm{NE}. \quad \mathrm{EV2}_{(j,p+1)} \tag{3}$$

$$\mathrm{EV1}_{(m,p)} \quad .\mathrm{EQ}. \quad \mathrm{EV2}_{(n,p)} \tag{4}$$

where i is 1, ..., AT1; j is 1, ..., AT2; $m(n)$ is the atom number belonging to structure **1** (**2**); and p is the number of iterations performed. If those conditions are valid for atom m of the first query structure and atom n of the second structure, these atoms have the largest identical local structural environments with sizes of $p - 1$ layers. For those more complicated query structure pairs which contain the same number of rings but at least three rings systems, one such atom pair (m,n) is selected as the unique starting pair.

Method 3: For substructure matching problems, select one atom having the largest number of neighbors within the query structure to comprise starting pairs with all those atoms which have at least the same number of neighbors in the other structure as the first starting atom.

Method 4: For a general MCSS problem, all those atoms with a number of neighbors equal or greater than 3 are selected as starting pairs from the two query structures. All available symmetry conditions are applied to reduce the number of starting pairs.

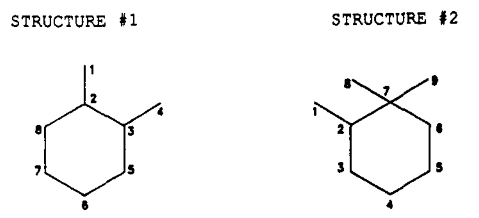
For colored graph matching problems, all the other atom properties like atom types and hybridization are additionally taken into account.

(b) Terminating Conditions. Here, the "terminating conditions" are referred to those conditions which decide that all possible maximal common substructures have been found; further possible starting pairs are rejected in order to save computing time. However, inappropriate terminating conditions may abort the search too early, resulting in loss of some or even all MCSSs. As stated above, the proof of structural identity and substructural isomorphism is well-defined, therefore easy terminating conditions are valid.

For the isomorphism problem, the total number of bonds within one structure is used as a terminating condition. The MCSS candidate containing as many bonds as the query structure represents the final solution. For substructure identification, the number of atoms within the query structure is used as a terminating condition.

In many applications, we are principally more interested in the direct comparison of two very similar compounds, especially those which contain the same parent ring system. In order to achieve a high efficiency for this kind of problem, a special terminating condition is used. A starting pair selected by above-described method 2 may give a MCSS candidate which contains the same parent ring system as the two query structures and which represents at least 80% of the bonds of the larger structure under investigation. It can reasonably be deduced that those structures are very similar and that the MCSS candidate can be used as a final solution.

For a general MCSS problem, it is necessary to process all possible starting pairs found by the methods described previously, which is usually a very CPU-intense task.



(a) DM-Matrix

STRUCTURE #2

		7	2	6	8	9	1	3	5	4
	2	-1								
	1		-1	1	1	1				
	3		-2	-1	1	1				
	8		-2	-2	-1	1				
STRUCTURE #1	4						-1	-1	-1	
	5						-1	-1	-2	
	7						-1	-1	-2	
	6									-1

(b) CM-Matrix

STRUCTURE #2

		7	2	6	8	9	1	3	5	4
	2	1								
	1		1							
	3			1						
	8				1					
STRUCTURE #1	4								1	
	5									
	7									
	6									

Figure 1. DM and CM matrices of structures **1** and **2** with original order (starting pair: 2–7). The information shown belongs to one partial mapping between structures **1** and **2** before any backtracking operation is performed.

(c) Reorder Branches of the Search Trees. Within a search tree, the computation time for detection of the largest MCSS candidate depends upon the order of its branches. Appropriate rearrangement of the sequence of the branches may result in earlier detection of larger MCSS candidates. In order to explain this strategy clearly, first let us briefly discuss the representation of the search process. In the MCSS algorithm, Decreased Matrix (DM) is used to describe the search process, and Corresponding Matrix (CM) stores the correspondences of atoms within the two structures. Figure 1 gives an example (atom 2 in structure **1** and atom 7 in structure **2** are chosen as starting pair). In the DM and CM, the vertical axis corresponds to the smaller structure, and the horizontal axis to the larger structure. As shown in Figure 1, atoms of each structure are ranked according to their distances from the starting atom.

One basic matching condition of atoms from two structures is that they should have the same distances from their starting atoms, respectively. The original values of DM elements are set according to this condition. If atom i of structure **1** and atom j of structure **2** have the same distance value, it is possible for them to correspond to each other, therefore $\mathrm{DM}_{(i,j)}$ is set to 1, otherwise $\mathrm{DM}_{(i,j)}$ is set to 0. For example, the selected starting atoms 2 and 7 have the same distance value 0; $\mathrm{DM}_{(2,7)}$ is initialized with 1. The DM matrix clearly shows that application of this "equal-distance" condition reduces search space dramatically. There are three basic rules to change the DM-matrix and the CM-matrix elements during the search process: (a) If atom i of structure **1** has been successfully matched to atom j of structure **2**—this information is recorded in the CM matrix by setting $\mathrm{CM}_{(i,j)} = 1 - \mathrm{DM}_{(i,j)}$ is changed from 1 into −1 to store the fact that this possibility has been tried. In order to avoid atom j's matching to other previously possible atoms k of structure **1** in the next steps, all elements $\mathrm{DM}_{(k,j)}$ are changed from 1 into −2. (b) If atom i fails to

(a) DM-Matrix

		STRUCTURE #2								
		7	2	6	8	9	3	5	1	4
	2	-1								
	3		-1	1	1	1				
	8		-2	-1	1	1				
	1		-2	-2	-1	1				
STRUCTURE #1	5						-1	1	1	
	7						-2	-1	1	
	4						-2	-2	-1	
	6									-1

(b) CM-Matrix

		STRUCTURE #2								
		7	2	6	8	9	3	5	1	4
	2	1								
	3		1							
	8			1						
	1				1					
STRUCTURE #1	5						1			
	7							1		
	4								1	
	6									1

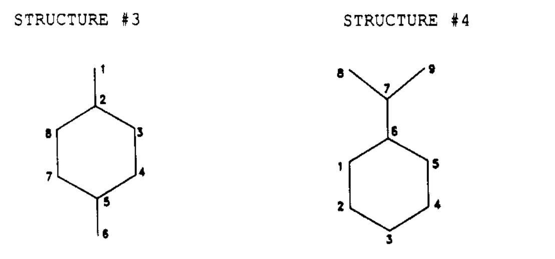
Figure 2. DM and CM matrices of structures **1** and **2** with new sequence of atoms (starting pair: 2–7). The information shown belongs to one partial mapping between structures **1** and **2** before any backtracking operation is performed.

match atom j, $DM_{(i,j)}$ will also be changed from 1 into -1, but $CM_{(i,j)}$ and $DM_{(k,j)}$ remain unchanged. (c) After the search through a branch has been completed, the backtrack operations result in a reinitialization of the corresponding lines of the DM and CM matrix, a necessary preparation for searching a new branch.

As shown in Figure 1, atom 1 in structure **1** is first tried to match atom 2 in structure **2**. Following this search branch, a substructure with five atoms and four bonds is found before any backtracking is performed; the size of the common structural fragment found so far is used as the backtrack condition. With this backtrack condition, no branches were cut off before the next, more-extended substructure containing six atoms and five bonds is found. Then, this new "best" number of bonds is used as a new backtrack condition for further search. It is easily seen from structures **1** and **2** in Figure 1 that the correct MCSS cannot be found until atom 1 in structure **1** is matched against atom 8 or 9 in structure **2**. This example reveals that in the above process most CPU time is spent on a lot of useless searches on bad branches within the search tree.

Now let us consider the rearrangement of the order of branches within the same search tree. The method is based on the following idea: The mapping of two atoms which are connected to a larger number of neighbors will usually result in finding larger substructures than the mapping of two atoms with fewer neighbors. The operations of reordering the branches are performed within each atom group, in which all atoms possess the same distance values. The atom which has the largest number of directly bonded neighbors is ranked first. This can be clearly illustrated using Figure 2. The order 1,3,8 and 4,5,7 of the atoms in structure **1** has been changed into 3,8,1 and 5,7,4, respectively. The equivalent change of atoms 1,3,5 in structure **2** leads to the new sequence 3,5,1. It is interesting to note that this simple procedure of reordering allows immediate detection of the correct MCSS (with eight atoms and eight bonds) using the same starting pair, even before any backtrack operation is performed.

Thus, the backtrack condition immediately obtains its maximal value of 8. Using this value, as long as one atom in structure **1** fails to match any atom in structure **2**, the corresponding branch will be immediately cut off.



(a) DM Matrix

		STRUCTURE #4								
		6	7	1	5	2	4	8	9	3
	2	-1								
	3		-1	-1	1					
	8		-1	-2	-1					
	1		-1	-2	-2					
STRUCTURE #3	4					-1	1	1	1	
	7					-2	-1	1	1	
	5									-1
	6									

(b) CM Matrix

		STRUCTURE #4								
		6	7	1	5	2	4	8	9	3
	2	1								
	3			1						
	8				1					
	1		1							
STRUCTURE #3	4					1				
	7						1			
	5									1
	6									

Figure 3. DM and CM matrices showing the first MCSS between structures **3** and **4** having the new sequence of atoms (starting pair: 2–6).

(a) DM Matrix

		STRUCTURE #4								
		6	1	5	7	2	4	8	9	3
	2	-1								
	1		-1	-1	-1					
	3		-1	1	-2					
	8		-2	-1	-2					
STRUCTURE #3	4					-1	1	1	1	
	7					-2	-1	1	1	
	5									-1
	6									

(b) CM Matrix

		STRUCTURE #4								
		6	1	5	7	2	4	8	9	3
	2	1								
	1				1					
	3		1							
	8			1						
STRUCTURE #3	4					1				
	7						1			
	5									1
	6									

Figure 4. DM and CM matrices showing the first MCSS between structures **3** and **4** having the original sequence of atoms (starting pair: 2–6).

It should be noted that the new order of the search tree will not always find the largest MCSS candidate before any backtracking. As an example, consider the two structures in Figure 3. The search starts with matching atom 3 in structure **3** to atom 7 in structure **4**. Following that branch, a substructure containing six atoms and five bonds is found before any backtracking occurs. This result is as good as the one obtained by using the original sequence. However, with this new order, the correct MCSS was detected much earlier than with the original sequence of atoms; this fact can be easily derived by comparison of Figures 3 and 4.

The rearrangement of branches is performed before backtrack searching, therefore, only a small amount of computation

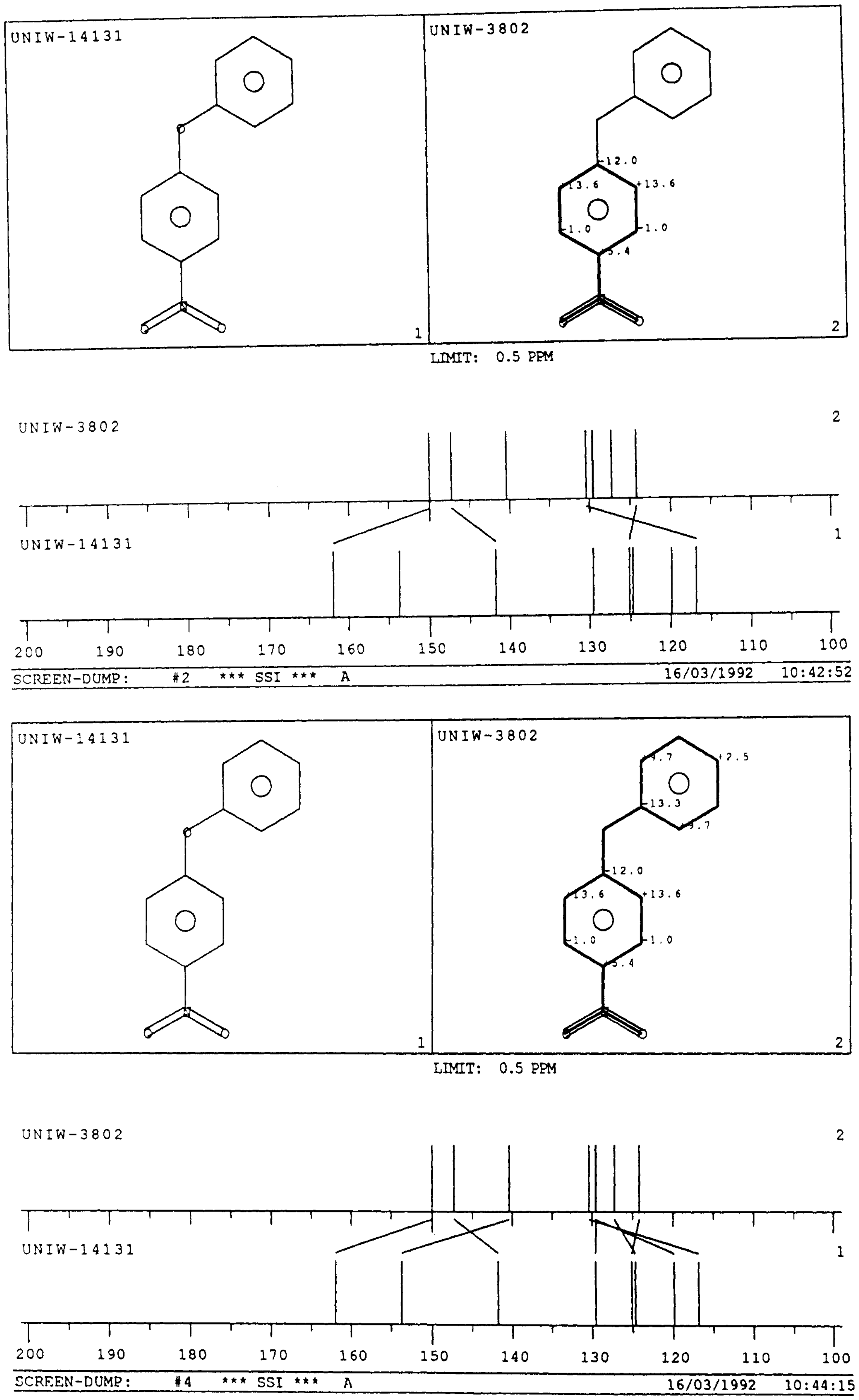


Figure 5. Comparison of the spectra of 4-nitrodiphenyl ether and 4-nitrodiphenylmethane. (Top) Color structure matching. (Bottom) Topological structure matching.

time is needed. The method proposed in this section can be used to guide the backtrack search within one search tree in such a way that larger substructures may be found as early as possible during the search process.

(d) Assigning a Larger Initial Value to the Backtrack Condition. As discussed previously, unlike structure and substructure isomorphism problems, for general MCSS problems, it is impossible to predict how many atoms and bonds MCSSs will have. Therefore, an initial value of 0 is assigned to the backtrack condition. In certain cases, some useless trees may be searched first; in such cases the backtrack condition will not obtain a larger value until a search tree containing a larger common substructure has been processed. Thus, much time is spent upon searching a lot of unnecessary trees.

In this section, we propose a method for setting the backtrack condition to a larger initial value instead of 0. With this method, it becomes possible to avoid useless searches at a very early stage. The principle of this approach is based on the following findings. In the search forest, each first branch of each tree may contain different common substructures with different sizes. Using the technique for reordering branches described above, larger common substructures, even final MCSSs, may appear on the first branches. By means of the backtrack search method, most computation time is spent upon the backtrack process. However, for each search tree, exploring its first branch leads to common substructures before any backtrack operations occurs and usually costs only a little time. Therefore, searching each first branch of each tree, in turn, may find a large common substructure. The number of bonds contained in the largest common substructure found can be used as the initial value for the backtrack condition.

(e) Atom/Bond Property Conditions. Various atom and/or bond properties can also be used to cut off some bad branches, even omit useless trees. However, it must be pointed out that the decision about using these additional limitations depends very heavily upon the structure pair under consideration. Figure 5 shows the comparison of 4-nitrodiphenyl ether (UNIW-14131) and 4-nitrodiphenylmethane (UNIW-3802). The MCSS algorithm discussed here gives the results as drawn in Figure 5, depending on the consideration of atom and bond properties. The display at the top of Figure 5 shows colored structure matching allowing only the comparison of the shift increments within the small nitrobenzene fragment, whereas topological structure matching displays all interesting shift differences giving the information the spectroscopist usually is interested in.

EXPERIMENTAL SECTION

The MCSS algorithm was implemented in FORTRAN-77 under the UNIX operating system on a Silicon Graphics and on an IBM-R6000 workstation. The program contains about 8 000 lines of source code. The databases accessed are the spectral data collection from University of Vienna and SADTLER Research Laboratories.

CONCLUSION

The MCSS algorithm developed here was tested upon a carefully selected set of structure pairs having a large variety of functional groups. The algorithm was applied to a database holding some 80 000 ^{13}C-NMR spectra in order to derive chemical shift increments. This detailed investigation proved that our MCSS algorithm is a very fast technique, especially for similar structure pairs. There is, in principle, no limitation on the sizes of the structures considered; for the current version, structures with a maximum number of 158 non-hydrogen atoms can be handled. The algorithm can also deal with structures which consist of two or more disconnected parts. Furthermore, the MCSS algorithm can be used to perceive topological or colored correspondencies according to the user's request.

The MCSS algorithm has become a central mainstay of several other commands within the CSEARCH–NMR database system.[21,22] Some applications of the MCSS algorithm to the study of ^{13}C-NMR spectra will be presented in the following paper.[21]

ACKNOWLEDGMENT

L.C. thanks the Austrian Academic Exchange Service for financial support. We are grateful to the staff of the University Computing Center for helpful discussion during program development. This project was supported by the European Academic Supercomputing Initiative (EASI).

REFERENCES AND NOTES

(1) Armitage, J. E.; Crowe, J. E.; Evans, P. N.; Lynch, M. F. Documentation of Chemical Reactions by Computer Analysis of Structural Changes. *J. Chem. Doc.* **1967**, *7*, 209–215.
(2) Armitage, J. E.; Lynch, M. F. Automatic Detection of Structural Similarities among Chemical Compounds. *J. Chem. Soc. C* **1967**, *7*, 521–528.
(3) Harrison, J. M.; Lynch, M. F. Computer Analysis of Chemical Reactions for Storage and Retrieval. *J. Chem. Soc. C* **1970**, *10*, 2082–2087.
(4) Lynch, M. F.; Willett, P. The Automatic Detection of Chemical Reaction Sites. *J. Chem. Inf. Comput. Sci.* **1978**, *18*, 154–159.
(5) Vleduts, G. E. Development of a Combined WLN/CTR Multilevel Approach to Algorithmical Analysis of Chemical Reactions in View of Their Automatic Indexing. British Library Research and Development Report No. 5399; London, 1977.
(6) 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.
(7) McGregor, J. J. Backtrack Search Algorithms and the Maximal Common Subgraph Problem. *Software—Pract. Exp.* **1982**, *12*, 23–34.
(8) Levi, G. A Note on the Derivation of Maximal Common Subgraphs of Two Directed or Undirected Graphs. *Calcolo* **1972**, *9*, 341–352.
(9) 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.
(10) Varkony, T. H.; Shiloach, Y.; Smith, D. H. Computer-Assisted Examination of Chemical Compounds for Structural Similarities. *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 104–111.
(11) Barrow, H. G.; Burstall, R. M. Subgraph Isomerism, Matching Relational Structures and Maximal Cliques. *Inf. Proc. Lett.* **1976**, *4*, 83–84.
(12) Tarjan, R. E. Graph Algorithms in Chemical Computation. In *Algorithms for Chemical Computations*; Christoffersen, R. E., Ed., ACS Symposium Series 46, American Chemical Society: Washington, DC, 1977; pp 1–20.
(13) Funatsu, K.; Endo, T.; Kotera, N.; Sasaki, S. Automatic Recognition of Reaction Site in Organic Chemical Reactions. *Tetrahedron Comput. Methodol.* **1988**, *1*, 53–69.
(14) Wipke, W. T.; Rogers, D. Tree-Structured Maximal Common Subgraph Searching. An Example of Parallel Computation with a Single Sequential Processor. *Tetrahedron Comput. Methodol.* **1989**, *2*, 177–202.
(15) Yuan, S.; Zheng, C.; Zhao, X.; Zeng, F. Identification of Maximal Common Substructures in Structure/Activity Studies. *Anal. Chim. Acta* **1990**, *235*, 239–241.
(16) Bremser, W., unpublished results.
(17) Golomb, S. W.; Baumert, L. D. Backtrack Programming. *J. ACM*, **1965**, *12*, 516–524.
(18) 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.
(19) Shelley, C. A.; Munk, M. E. Computer Perception of Topological Symmetry. *J. Chem. Inf. Comput. Sci.* **1977**, *17*, 110–113, and references cited therein.
(20) Chen, L. Thesis, University of Vienna, in preparation.
(21) Chen, L.; Robien, W., following paper in this issue.
(22) Kalchhauser, H.; Robien, W. CSEARCH: A Computer Program for Identification of Organic Compounds and Fully Automated Assignment of Carbon-13 Nuclear Magnetic Resonance Spectra. *J. Chem. Inf. Comput. Sci.* **1985**, *25*, 103–108.