Structure Generation by Reduction: A New Strategy for Computer-Assisted Structure Elucidation

BRADLEY D. CHRISTIE and MORTON E. MUNK*
Department of Chemistry, Arizona State University, Tempe, Arizona 85287

Received June 17, 1987

A problem common to computer programs for structure elucidation is the efficient and prospective use of the input information to constrain the structure generation process. The input may consist of potentially overlapping substructure requirements and alternative substructure interpretations of spectral data. Other useful information may be structural features that must not be present in the output structures. All of these may interact in a complex manner that is impossible to determine by use of a bond-by-bond structure assembly algorithm. A new method is described called structure reduction. In contrast to structure assembly, this method begins with a set of all bonds and removes inconsistent bonds as structure generation progresses. This results in a more efficient use of the input information and the ability to use potentially overlapping required substructures. Several examples illustrate the application of our computer program COCOA, which uses this method to solve real-world structure elucidation problems.

INTRODUCTION

The heart of any automated system that aims to determine unknown molecular structures is a program that accepts specific types of structural information and produces the completed structures. Examples include Stanford's GENOA,[1] Sasaki's CHEMICS,[2] Bremser's ACCESS,[3] Dubois's DARC-EPIOS,[4] and our earlier programs ASSEMBLE,[5] and COMBINE.[6] All of these programs share a common method of structure generation with some variations. At the start of structure generation, the problem state is a collection of the unconnected atoms. Some of the programs begin with some atoms preconnected, using knowledge about required, nonoverlapping substructures. Using a depth-first search algorithm, all possible combinations for connecting the remaining atoms (the problem states) are explored, and the result is a list of all possible structures consistent with the input information. Because this algorithm sequentially adds bonds to the problem state, we call this method *structure assembly*.

The search space (the number of problems states that are examined) of a structure generation problem and the amount of computer time necessary to explore them all increase exponentially with the number of bonds to be added. The input information serves to constrain the process (by reducing the size of the search space) and slow down the exponential growth. Efficient use of the input information is important if any but the simplest molecules are to be successfully handled by the program.

One important type of information is a list of required substructures. This is a common way to express the results of interpretation of spectral and chemical data. Structure assembly can begin with the required substructures as the initial problem state *only if* these substructures are known to be nonoverlapping. If the initial problem state does not contain a required substructure, then it must be generated during structure assembly. Since structure assembly adds bonds to the problem state, the presence of a required substructure cannot generally be determined until a structure is completed. Elimination of invalid structures at this point does not reduce the size of the search space. Structure assembly algorithms thus have an inherent difficulty using potentially overlapping required substructures.

Several methods have been employed to overcome this problem. ASSEMBLE can accept nonoverlapping and one-atom overlapping required substructures as the starting point for structure assembly. These one-atom overlaps (which must be explicitly defined by the user) are not used as part of the initial problem state but serve to constrain the connections chosen during structure assembly. It is the user's responsibility to determine which required substructures do not overlap. A more general solution used by GENOA is to preprocess the required substructures by determining all possible nonoverlapping combinations. Each combination is a nonoverlapping starting point for the assembly process, as in ASSEMBLE.

The complementary constraint to the required substructure is the forbidden substructure. Forbidden substructures must not be present in any of the output structures. By use of the structure assembly algorithm, forbidden substructures can be detected during the process as they are formed, so the difficulties encountered with required substructures do not occur here. Both GENOA and ASSEMBLE use forbidden substructure constraints efficiently. However, testing at every step for forbidden substructures that cannot possibly be built from the present state can result in an unnecessary increase in execution time.

Some types of spectral data cannot usually be assigned to specific substructures. One example is ^{13}C NMR, in which each peak can correspond to any one of several different substructures. One way to express this is as a list of uniform alternative substructures for each peak. The structure generation program must then select among these substructures. The DARC-EPIOS, ACCESS, and COMBINE programs use this method. Neither DARC-EPIOS nor COMBINE can use forbidden or required substructure constraints. ACCESS offers options to observe the progress of structure generation and to guide the program in a sensible direction.

A list of alternative substructures is one way to express ambiguous information. Other types of ambiguous information need to be expressed in different ways. For example, the presence of an aromatic ring might be determined but without information on the pattern of substitution or even if it is a benzene ring or some aromatic heterocycle. Because of the necessity of starting with the required substructures or a derivation of the required substructures structure assembly programs are limited in their capability of accepting ambiguous substructure requirements.

Combinations of constraints can lead to an increase in program inefficiency. For example, consider a problem with the molecular formula $C_4H_{10}O$ and forbidden substructures OH and OCH_3. The only valid structure is diethyl ether. At some point in the process, structure assembly programs such as GENOA and ASSEMBLE might construct the fragment $CH_3-CH_2-CH_2-$. Since none of the forbidden substructures have been built, this is a valid substructure, and structure assembly will continue. Of course, no valid structures will be produced. The important point is that the *combination* of the molecular formula and substructure constraints has determined

0095-2338/88/1628-0087$01.50/0

the outcome of this problem. With another methylene group to work with, the valid structure of *n*-propyl ethyl ether could be built. Without the substructure constraints, *n*-propyl methyl ester and 1-butanol could be built.

All of the programs previously mentioned have as minimum constraints the molecular formula and structural theory, so this interaction problem can occur with each one. As the number and types of constraints increase, the interaction problem becomes more severe. In extreme cases, ASSEMBLE can spend almost all of the execution time unproductively because of this problem. Since real-world structure elucidation problems must generally be expressed as a complex collection of many different types of constraints, this interaction problem must be addressed if the structure generation program is to be effective.

Of course, any structure generation algorithm using a depth-first search algorithm will make incorrect choices. A more intelligent program would reduce the number of incorrect choices. The cost for this is an increase in time needed to examine each problem state. Since the size of the search space increases exponentially with the size of the problem (in this case, the average number of bonds that must be added to the initial problem state to generate a complete structure), increasing the intelligence of the program is most effective when dealing with larger structures. With the more difficult problems, the increase in time spent at each problem state is more than made up for by the decrease in the total problem states examined. In other words, it helps reduce the "steepness" of the exponential curve.

None of the programs described use required symmetry as a constraint during structure assembly. Our system for interpretation of 2-D NMR spectral data does use symmetry in constructing fragments,[8] but ASSEMBLE checks only completed structures for the required symmetry.[9] In both cases, the symmetry is expressed as symmetrical carbon atoms, and the required symmetry is obtained from the ^{13}C NMR spectral data.

Current structure generation program capabilities fall far short of the requirements needed to solve today's structure elucidation problems. The following features are central to any meaningful increase in program performance: (1) the ability to use both forbidden and potentially overlapping required substructures prospectively without a required preprocessing step; (2) the ability to accept a variety of ambiguous structural information through the use of a flexible substructure representation; (3) the ability to use alternative substructure requirements, such as may result from ^{13}C NMR interpretations; (4) a general method to allow for the efficient interaction between all types of constraints; (5) the ability to use the required symmetry as a constraint during the structure generation process. Since most of these goals are impossible to realize via the structure assembly method, we have developed a new method for structure generation.

DISCUSSION AND RESULTS

Fundamentals of the New Method. The problem of using required substructures in structure assembly can be expressed by using set theory and Venn diagrams, as shown in Figure 1. The universal set is the set of all bonds that could be considered in building a structure. Subsets include the bonds describing forbidden substructures, required substructures, any partial structure during assembly, and any completed structure. The set of a completed structure must contain the sets of all required substructures. Since the set of a partial structure grows as each new bond is added, the containment of required substructures may not occur until the structure is completed. However, the undesired presence of a forbidden substructure can be detected immediately upon the containment of its set.

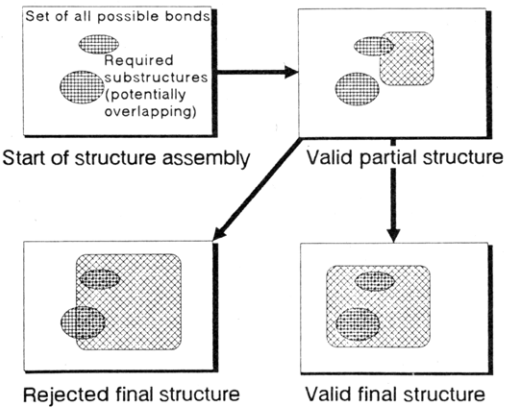


Figure 1. Structure generation by assembly.

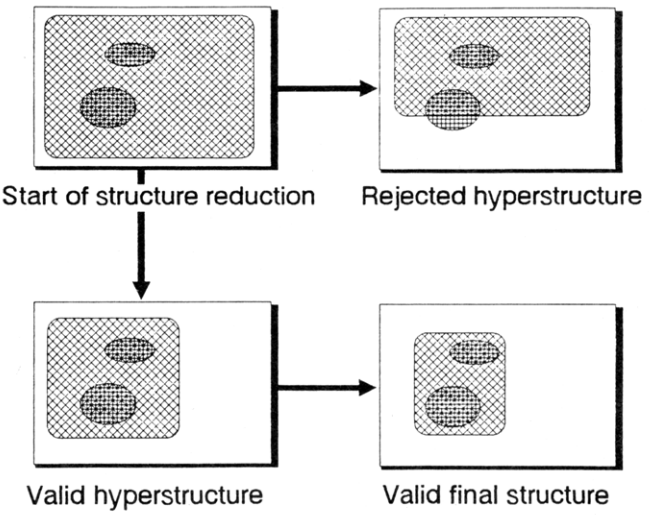


Figure 2. Structure generation by reduction.

Our new method begins with the universal set as the starting point for structure generation. Bonds are removed until a valid structure remains. This process is illustrated with Venn diagrams in Figure 2. Since the working structure contains more bonds than the final structure, it will be referred to as a *hyperstructure*, in contrast to a partial structure of the structure assembly method. Also in contrast to structure assembly, we will refer to the step-by-step removal of bonds from a hyperstructure as *structure reduction*. Initially, the hyperstructure contains all of the required substructures. As bonds are removed, the continued containment of each required substructure is tested. The structure reduction process is thus guided prospectively by the required substructures. It is important to note that the degree of overlap between the individual required substructures does not play a role in this method—any amount of overlap can occur without modification of the method, simply by testing sequentially for the containment of each required substructure.

Representation. Our specific representation for a hyperstructure is a bonding adjacency matrix (BAM). Each nonhydrogen atom is divided into separate bonding sites. Each potential single-, double-, and triple-bond connection is represented by one bonding site. An sp^2 carbon with no attached hydrogens has two "single-bond" bonding sites and one "double-bond" bonding site. There is no "aromatic" type; aromatic rings are represented as alternating single and double bonds. For each bonding site, a set contains other bonding sites it may connect to. Together, all these sets of bonding-site possibilities define the BAM. There is a one-to-one pairing of bonding sites in a complete structure. This reflects one of the most elementary concepts of structural theory—every bond connects two atoms.

Bonding Site	Connections	After Matching	Complete Structure
1 CH3-	2 3 4	CH3- 2 3	CH3- 2
2 CH2-	1 4	CH2- 1 4	CH2- 1
3 -	1 4	- 1 4	- 4
4 OH-	1 2 3	OH- 2 3	OH- 3

Figure 3. Hyperstructures of ethanol.

An example of a starting BAM for a structure reduction problem using the component atoms of ethanol is shown in Figure 3. This is obtained by placing every bonding site in each bonding-site set and then removing connections between the same atom. Thus, the methyl atom (1) cannot connect to itself (1) but can connect to either bonding site of the methylene atom (2 and 3) or the hydroxyl atom (4). The first bonding site of the methylene atom (2) cannot connect to either itself or the other bonding site of the methylene atom (2 or 3).

At each stage of the structure reduction process, a series of tests, such as the substructure constraints, is performed by examination of the BAM. Each test is responsible for returning an error message if the BAM fails the test. In addition, most of the tests remove bonds that would produce an error if they were chosen. This common use of the BAM results in an indirect interaction of all of the tests. As a result, program efficiency is greatly increased. Following are descriptions of each of these tests.

Matching. A close examination of the first BAM in Figure 3 reveals that the two bonds of the methylene group (bonding sites 2 and 3) must collectively bond to the methyl and hydroxy bonds. Since this is a one-to-one relationship, the methyl and hydroxy bonds must not attach to any bonds other than the methylene bonds. If the methyl bond were to attach to the hydroxy bond, the methylene bonds would be left with no available attachments. The methyl–hydroxy bond may therefore be removed from the BAM, as in the second BAM of Figure 3.

The requirement that each bonding site have a unique attachment may be expressed as a matching problem. Matching is an important concept in graph theory and set theory.[10] Of importance here is the König–Hall theorem.[11] First, an arbitrary set of bonding sites is selected (set A). The next step is to calculate the union of possible attachments of the bonding-site members of set A. This is the "image" of set A, or ΓA. Finally, the number of members of set A (the "cardinality" of set A, or $|A|$) is compared to $|\Gamma A|$. The König–Hall theorem states that a matching is possible if and only if, for every set A (i.e., all possible subsets of the set of bonding sites), $|\Gamma A| \geq |A|$. If a set A can be found for which $|\Gamma A| < |A|$, then the hyperstructure can be rejected.

Returning to the ethanol example, there are no sets for which $|\Gamma A| < |A|$. There is a set $A = \{2,3\}$ ($\Gamma A = \{1,4\}$) for which $|\Gamma A| = |A|$. If one of the members of ΓA were removed, an invalid hyperstructure would result. All members of ΓA must therefore attach to members of A. This is the mathematical basis for removing the 1,4 attachment.

As another example using a larger molecule, consider the starting BAM of 1,3,5-trimethylbenzene (mesitylene), shown in Figure 4a. It has been edited only to eliminate loops, connections between incompatible bonding sites (single vs double), and connections between methyl groups (which result in a disjoint ethane molecule). Now presume that information is available which precludes vicinal hydrogens. Elimination of connections between hydrogen-bearing atoms gives the BAM of Figure 4b.

Now a search for König–Hall sets gives $\Gamma\{5,7,9\} = \{12,15,18\}$. As a result, $\{12,15,18\}$ is forced to $\{5,7,9\}$, which gives the BAM shown in Figure 4c. Another König–Hall set is $\Gamma\{1,2,3,4,6,8\} = \{10,11,13,14,16,17\}$. Elimination of the connections of $\{10,11,13,14,16,17\}$ that are not elements of $\{1,2,3,4,6,8\}$ results in the BAM of Figure 4d. The effect of all these operations, which is visually apparent from Figure

a

Bonding Site	Connections
1 CH3-	4 6 8 10 11 13 14 16 17
2 CH3-	4 6 8 10 11 13 14 16 17
3 CH3-	4 6 8 10 11 13 14 16 17
4 CH-	1 2 3 6 8 10 11 13 14 16 17
5 =	7 9 12 15 18
6 CH-	1 2 3 4 8 10 11 13 14 16 17
7 =	5 9 12 15 18
8 CH-	1 2 3 4 6 10 11 13 14 16 17
9 =	5 7 12 15 18
10 C-	1 2 3 4 6 8 13 14 16 17
11 -	1 2 3 4 6 8 13 14 16 17
12 =	5 7 9 15 18
13 C-	1 2 3 4 6 8 10 11 16 17
14 -	1 2 3 4 6 8 10 11 16 17
15 =	5 7 9 12 18
16 C-	1 2 3 4 6 8 10 11 13 14
17 -	1 2 3 4 6 8 10 11 13 14
18 =	5 7 9 12 15

b

Bonding Site	Connections
1 CH3-	10 11 13 14 16 17
2 CH3-	10 11 13 14 16 17
3 CH3-	10 11 13 14 16 17
4 CH-	10 11 13 14 16 17
5 =	12 15 18
6 CH-	10 11 13 14 16 17
7 =	12 15 18
8 CH-	10 11 13 14 16 17
9 =	12 15 18
10 C-	1 2 3 4 6 8 13 14 16 17
11 -	1 2 3 4 6 8 13 14 16 17
12 =	5 7 9 15 18
13 C-	1 2 3 4 6 8 10 11 16 17
14 -	1 2 3 4 6 8 10 11 16 17
15 =	5 7 9 12 18
16 C-	1 2 3 4 6 8 10 11 13 14
17 -	1 2 3 4 6 8 10 11 13 14
18 =	5 7 9 12 15

c

Bonding Site	Connections
1 CH3-	10 11 13 14 16 17
2 CH3-	10 11 13 14 16 17
3 CH3-	10 11 13 14 16 17
4 CH-	10 11 13 14 16 17
5 =	12 15 18
6 CH-	10 11 13 14 16 17
7 =	12 15 18
8 CH-	10 11 13 14 16 17
9 =	12 15 18
10 C-	1 2 3 4 6 8 13 14 16 17
11 -	1 2 3 4 6 8 13 14 16 17
12 =	5 7 9
13 C-	1 2 3 4 6 8 10 11 16 17
14 -	1 2 3 4 6 8 10 11 16 17
15 =	5 7 9
16 C-	1 2 3 4 6 8 10 11 13 14
17 -	1 2 3 4 6 8 10 11 13 14
18 =	5 7 9

d

Bonding Site	Connections
1 CH3-	10 11 13 14 16 17
2 CH3-	10 11 13 14 16 17
3 CH3-	10 11 13 14 16 17
4 CH-	10 11 13 14 16 17
5 =	12 15 18
6 CH-	10 11 13 14 16 17
7 =	12 15 18
8 CH-	10 11 13 14 16 17
9 =	12 15 18
10 C-	1 2 3 4 6 8
11 -	1 2 3 4 6 8
12 =	5 7 9
13 C-	1 2 3 4 6 8
14 -	1 2 3 4 6 8
15 =	5 7 9
16 C-	1 2 3 4 6 8
17 -	1 2 3 4 6 8
18 =	5 7 9

Figure 4. Hyperstructures of 1,3,5-trimethylbenzene: (a) starting hyperstructure; (b) after elimination of vicinal hydrogen connectivities; (c) after matching {5,7,9} to {12,15,18}; (d) after matching {1,2,3,4,6,8} to {10,11,13,14,16,17}.

4d, is that there are no connections between any of the substituted aromatic carbon atoms.

A special case arises when $A = \Gamma A$. Although $|A| = |\Gamma A|$, if $|A|$ is odd, no valid structures can result (an internal pairing of the members of A must leave one member unpaired). If this is detected during the match test, the hyperstructure is rejected.

For structure elucidation problems of even moderate size, applying the König–Hall theorem to each of the $2^N - 2$ proper subsets of N bonding sites is impractical. Two passes are made through the BAM, the first building sets from members with identical images and the second building sets from members with overlapping images. This catches the vast majority (although not all) of the König–Hall subsets. Since it only

Original BAM

Class	Bonding Site		Connections	Connected Classes (CH_3 only)
A	1	CH_3-	3 4 5 8	{B, C, D}
A	2	CH_3-	5 6 8	{C, D}
B	3	CH_2-	1 6 7 8	
B	4	-	1 6 7 8	Intersection = {C, D}
C	5	CH -	1 2	- Classes with
C	6	-	2 3 4 8	less than 2 bonds - {D}
C	7	-	3 4	
D	8	OH -	1 2 3 4 6	Result {C}
				BAM row numbers {5, 6, 7}

Recalculated BAM

Class	Bonding Site		Connections
A	1	CH3-	5
A	2	CH3-	5 6
B	3	CH2-	6 7 8
B	4	-	6 7 8
C	5	CH -	1 2
C	6	-	2 3 4 8
C	7	-	3 4
D	8	OH -	3 4 6

Figure 5. Hyperstructure reduction by use of the symmetry test.

makes two linear passes through the BAM, it also operates very quickly.

Disjoint Test. The disjoint test looks at the BAM to determine if there is a set of atoms with no connections to the remaining atoms. This is performed by a breadth-first search of the connected atoms. The BAM is not changed by this procedure—only an error message is returned. This is the shortest and simplest of the BAM tests.

Multiple Edge Test. For each multivalent atom, every subset of two or more of its bonding sites is examined. The images of these bonding sites of this subset to other bonding sites are added together (logical OR). Next, these connecting bonding sites are translated to the corresponding connected atoms. The number of connected atoms must be greater than or equal to the size of the subset (otherwise a multiple edge would eventually be forced). Note that as double and triple bonds are explicitly represented as different types of bonding sites, combination of two or three single bonds to form a multiple bond is not allowed.

Symmetry Test. Through the use of the ^{13}C NMR data and composition generation (discussed below), a knowledge of symmetrical carbon atoms is available. This knowledge can be used to remove potential bonds from the BAM that cannot lead to a structure with the correct symmetry.

All atoms are first partitioned into classes on the basis of the required symmetry. Carbon atoms that must be symmetrical are placed in the same class. Heteroatoms of the same type and hybridization are also placed in the same class—since no knowledge is available, they are assumed to be symmetrical as otherwise valid bonds might be eliminated. The logic of this method is that symmetrical atoms must each bond to the same classes in order to remain symmetrical.

Next the classes containing more than one atom are examined. For each atom, the connected classes are tabulated. The intersection of these connected classes is calculated. Any class with a smaller number of bonds than the number of atoms in the class being examined is also removed from the calculated intersection. Finally, by removing potential bonds from the BAM, each atom of the class is forced to bond only to this intersection.

An example of this algorithm is shown in Figure 5. The structure elucidation problem is isobutyl alcohol, which has a pair of symmetrical methyl groups. The first BAM is that of a hypothetical hyperstructure partway through the structure reduction process. The symmetrical methyl groups are both placed in class A. The other atoms, being of different types, are each placed in a unique class.

The intersection of the connected classes of class A is a set containing classes C and D. Since there are two atoms in class A and class D has only one bond, class D is removed from the calculated set. This leaves class C (the methine carbon) as the only class that class A may bond to. This set is translated into the corresponding bonding sites to which the members

BAM Bonding Site		Connections	Substructure Bonding Site		Correspondence Set
1	CH3-	5	a	CH_2-	{3, 4}
2	CH3-	6	b	CH-	{7}
3	CH2-	7 8	c	CH-	{7}
4	-	7 8	d	CH_2-	{3, 4}
5	CH -	1			
6	-	2			
7	-	3 4			
8	OH -	3 4			

Figure 6. Search for CH_2–CH–CH_2 substructure in isobutyl alcohol.

of class A are forced. The final BAM shows the result. Application of the match test will reduce this BAM further.

Substructure Constraints. Two types of substructure constraints are used: required and forbidden. Like the BAM, each substructure is stored as a list of bonding sites. Since each bond of the substructure is fixed, the substructure bonding sites are paired. Other substructure information includes the types of atoms and bonds.

For each substructure bonding site, a set is maintained listing all possible corresponding bonding sites of the BAM. These correspondence sets are edited on the basis of connectivity of the BAM and the substructure.

A pair of match tests is used to prevent, for example, a CH_3–C group from satisfying a *gem*-dimethyl required substructure. First, the bonding sites of the substructure are matched to the bonding sites of the BAM. This is followed by a matching of the atoms of the substructure to the atoms of the hyperstructure. This latter matching is essentially the same as the cardinality violation check of the Sussenguth substructure search algorithm.[12] For each of these substructure match tests, the two-pass match method described earlier is used.

As an example, consider a search for a CH_2–CH–CH_2 group in the BAM of Figure 6. This BAM is the result of the application of the match test to the final BAM of Figure 5. Listed in Figure 6 are the correspondence sets for each of the bonding sites of the CH_2–CH–CH_2 group. These sets have been edited on the basis of possible connections of the BAM. Substructure bonding site a, which is the bonding site of the first CH_2, could be either CH_2 bonding site of the BAM, so the correspondence set contains the BAM elements 3 and 4. For the substructure CH bonding sites (b and c), initially all CH bonding sites of the BAM—5, 6, and 7—would be examined, but the first two are quickly eliminated as they cannot support the CH–CH_2 connection of the substructure. Substructure bonding site d, symmetrical with a, has the same correspondence set.

The first subset of substructure bonding sites examined in the substructure match test are the two methylenes, since their correspondence sets are disjoint from the other correspondence sets. Thus, $A = \{a,d\}$ and $\Gamma A = \{3,4\}$. In this case, $|\Gamma A| = |A|$. For the two methines, $A = \{b,c\}$ and $\Gamma A = \{7\}$. Since, for this set, $|\Gamma A| < |A|$, the substructure cannot be present, and the subroutine returns an error message.

The forbidden substructure routine operates in a similar fashion. After the correspondence sets are edited, they are examined for the presence of forced bonds. These are the bonding sites whose images contain only one other bonding site. Since there are no alternatives, these forced bonds must be present in any structures produced from the current problem state. If any one correspondence set contains none of the forced bonds, then the substructure is not yet forced on the BAM, and examination of that specific forbidden substructure ends until the next call to the test. If each correspondence set does contain a forced bond, an attempt is made to assign the individual substructure bonding sites to the forced structure bonding sites. A backup message is returned upon the successful completion of this "mapping" step.

If a forbidden substructure fails one of the match tests (using all bonds, not just the forced bonds), then it cannot be built from the present BAM. This allows the program to

"deactivate" the substructure until backup to a higher level occurs. The correspondence sets are saved, and the forbidden substructure constraint is ignored until the backup occurs. This avoids wasted time on unnecessary substructure searching.

Ring Strain Test. The previous program ASSEMBLE has a ring detection routine that produces a backup message if a strained ring is formed.[5] At each step, all the rings of the partial structures are found and compared to a table of strained rings. The user has some control over the parameters used to decide if a ring is unacceptable.

A routine with a similar function has been developed for our structure reduction program. At each step, any new forced bonds are collected and a spanning tree is updated. Remaining forced bonds are ring closure bonds. All new rings are found by using the Wipke–Dyott ring cluster method.[13] Each new ring is examined for violation of any of the strain rules (described under User Interface, below). All pairs of new rings with other rings in its cluster are examined to find all bicyclic ring systems. These are compared against the bicyclic ring strain constraints to find anti-Bredt and cyclophane ring systems.

As our system does not use an aromatic bond type, aromatic rings are represented as alternating single and double bonds. Each aromatic ring could thus be generated twice during structure reduction, the second time with exchanged single and double bonds. To counter this redundancy, aromatic rings are detected by examination of the new rings found by the ring test. For the purposes of this test, we define an aromatic ring as a ring with a total number of bonds equal to twice the number of double bonds. After a ring has been identified as aromatic, the bond connecting the atoms closest to the top of the BAM is located. If it is a single bond, a backup message is returned. Since all atoms maintain the same positions in the BAM during structure reduction, all redundant rings are found by this method without loss of any valid, nonredundant rings.

As with the other structure reduction tests, bonds are eliminated from the BAM on the basis of potential to violate a constraint. The spanning tree of forced bonds is used together with the BAM to identify other bonds that would close a ring if they became forced bonds. This ring closure bond is processed with the Wipke–Dyott algorithm to find all potential rings, and these in turn are further examined to find all potential bicyclic ring systems. If a potential ring or bicycle violates one of the strain rules or is a redundant aromatic ring, the bond in question is removed from the BAM.

Removal of Redundant Bonds. The match test, as applied to the ethanol BAM in Figure 3, gave a result that can be further reduced to give two alternative final BAMs for ethanol. To prospectively reduce the number of redundant structures produced, the symmetry of the bonding sites is determined at each step, and redundant bonds are eliminated. Continuing with the ethanol example, the two bonding sites of the methylene atom (2 and 3) are determined to be symmetrical. The two bonding possibilities of the methyl group are then identified as redundant. The first one is saved (1 to 2), and the other (1 to 3) is eliminated. Removal of the inverse (3 to 1), followed by reapplication of the match test, gives the final BAM of Figure 3.

This algorithm was found to work poorly for chains of CH_2 atoms. The algorithm stops after enough CH_2–CH_2 connections have been removed to make all the bonding sites topologically nonequivalent, leaving many more bonds than that needed to make a chain of methylene groups. A special routine detects these chains and forces one specific chain on the BAM by eliminating all other CH_2–CH_2 bonds.

Test Control. With two exceptions (the disjoint and redundant bond tests), each test accepts a set of "changed" bonding sites (bonding sites with fewer bonds since the last time the subroutine was called) and only examines that section of the BAM. The exceptions are the disjoint test, which operates quickly anyway, and the redundant bond test.

Those tests that change the BAM (all but the disjoint test) are also responsible for returning a set of changed bonding sites. In this way, each subroutine's output is input for the others. A control routine manages these sets of changes and cycles through each of the tests until no further changes in the BAM occur.

Because of this control routine and the use of changed sets, only the portion of the BAM that changed between problem states is examined. Relatively large hyperstructures can be treated without the prohibitive decrease in speed that might be expected from the information-intense approach of structure reduction. COCOA thus enjoys the efficiency typical of structure assembly programs while maintaining the highly descriptive problem states unique to structure reduction.

Element Groups and Atom-Centered Fragments. Structure reduction, as we have implemented it, cannot begin until the hybridization and number of attached hydrogens have been assigned for each atom. We define this assignment as an *element group*.[14] Given our limits on the types of atoms (carbon, hydrogen, nitrogen, oxygen, divalent sulfur, and the halogens; no charged moieties), there are 26 element groups, and we have added the "super" element group NO_2. These element groups are CH_3, CH_2=, CH≡, NH_2, NH=, N≡, OH, O=, SH, S=, F, Cl, Br, I, NO_2, CH_2, CH=, =C=, C≡, NH, N=, O, S, CH, C=, N, and C (single bonds are omitted for clarity).

An *atom-centered fragment* (ACF) is an element group with one concentric layer of neighboring element groups. Some examples are —CH_2—CH_2—CH_2— and —NH—C(=O)—O—. From our 27 element groups, 13 703 different ACFs can be generated.[14] A large number of these are chemically unstable—removal of these reduces the list to about 5100.[14] A simple interpretation program, separate from our structure reduction program, currently accepts the molecular formula and ^{13}C and ^{1}H NMR spectral data and produces a list of possible ACFs for each atom.

The structure reduction program begins by generating compositions of element groups compatible with the molecular formula of the unknown using a combinatorical algorithm.[15] Each composition assigns a specific element group to each atom. Since the number of hydrogens and hybridization of each carbon atom can be determined from the ^{13}C NMR data, and element group composition generation is tested prospectively against the substructure constraints, this is seldom a difficult problem.

After a composition of element groups has been selected, the BAM is set up and structure reduction begins. The selection of ACFs serves as the basis for removal of bonds. When an ACF is selected for a specific element group (which is the central element group of the ACF), its bonding sites are forced to connect to the bonding sites of the appropriate neighboring element groups. For example, if the selected ACF for a methylene element group is CH_3–CH_2–O–, then the first bonding site of the methylene element group would be forced (by removing any alternatives) to connect to a methyl group, and the second bonding site would be forced to a divalent oxygen.

After ACFs have been selected for every atom, specific bonds are selected for atoms that still have multiple bonding possibilities. The tests described above are performed after each ACF and bond is selected.

Thus, the overall structure of our structure reduction program is three exhaustive depth-first tree searches: (1) element group selection (composition generation); (2) ACF selection;

and (3) bond selection. Since use of ACFs is central to our program, we have named it COCOA, for COnstrained COmbination of ACFs.

User Interface. To run a problem, the user begins by entering the molecular formula and spectral information (^{13}C and ^{1}H NMR data by peak). The interpretation program reads this information and produces the list of possible ACFs. Future interpretation programs will also produce substructure constraints. Finally, COCOA is run, which reads the output of the interpretation program and then prompts the user for substructure constraints. Substructure constraints are entered by using a linear form that is handled by a recursive descent parser.[15] The Bachus-Naur form for the substructure constraints is

```
Constraint ::= alt-constraint {"|" alt-constraint}
Alt-constraint ::= fragment {";" fragment}
Fragment ::= atom-type [p-exp][bond fragment]
P-exp ::= "("fragment")"
Atom-type ::= ring-ident | atom-group
Atom-group ::= [ring-ident ":"][begin-ovlp] atom
  [end-ovlp]["H" num-of-h]
Atom ::= "C" [cshift] |"N"|"O"|"S"|"F"|"Cl"|"Br"|
  "I"|"NO2"|"A"|"G" digit
Num-of-h ::= {"0".."3"}
Begin-ovlp ::= "<"
End-ovlp ::= ">"
Bond ::= ["-"|"="|"+"|"."]
Ring-ident ::= digit
Digit ::= "1".."9"
Cshift ::= a real number (corresponding to a specific
  13C NMR shift)
```

Some examples will help illustrate the various features. An *n*-butyl group may be entered as CH3–CH2–CH2–CH2, CH2–CH2–CH2–CH3, CH2(–CH3)–CH2(–CH2), or CH3CH2CH2CH2 (implied bonds may be of any type, but the parser will determine from the atom types that all bonds must be single). Rings are formed by using a ring label and then a later reference to the label; an epoxide would be entered as 1:C–C–O–1. Disjoint fragments are entered by separating the fragments with a semicolon. Alternative constraints are separated by the "or" (vertical bar) character. The "A" atom matches any type of atom. The user may specify a list of element groups to use for a substructure atom by using the "G" atom; after entering the constraint, the user is prompted for this information.

Although overlap between constraints is allowed to any degree, overlap within a constraint is generally forbidden. This is necessary to prevent, for example, a dimethylcyclohexane constraint from being satisfied by a structure with just one cyclohexyl methyl group. This feature can be overridden by the use of the overlap flags ("<" and ">"). Enclosed atoms may overlap. This is useful, for example, in defining a constraint based on ^{1}H NMR coupling. Two methylene carbons showing only geminal coupling could be entered as the disjoint constraint ⟨AH0⟩–CH2–⟨AH0⟩;⟨AH0⟩–CH2–⟨AH0⟩. This constraint will now be satisfied by C(quat)–CH2–O–CH2–C(quat), for example. Entering the two methylenes as two separate constraints could produce structures with just one isolated methylene. In the substructure test algorithm, substructure bonding sites marked as potentially overlapping are not used in the substructure match test and may receive nonunique assignments during the mapping step.

If the carbon atoms of a substructure constraint can be related to the ^{13}C NMR spectral data (as is the case with constraints derived from 2-D NMR), this information can be entered in the constraint by following the "C" symbol with the chemical shift. This improves the efficiency of the substructure test by greatly reducing the matching possibilities.

Table I. Comparison between ASSEMBLE and COCOA

	structures	CPU times (s) Prime ASSEMBLE	Prime COCOA	Harris ASSEMBLE	Harris COCOA
(1) no additional constraints	140	355	1295	85	123
(2) F: CH2—CH2	81	472	754	103	69
(3) F: CH3—C=O	62	558	792	108	76
(4) both 2 and 3	36	628	470	118	44

There are some nonsubstructure constraints that may be entered. The SYMMETRY constraint, if the user requests it, checks the topological symmetry of each generated structure against the symmetry required by the ^{13}C NMR data. Structures that have too much symmetry are rejected (structures with not enough symmetry are prospectively eliminated by the symmetry test). It is possible for the topological symmetry to exceed the real symmetry, since the topology does not consider stereochemistry or hindered rotation around double bonds. For this reason, this symmetry constraint is an option that must be invoked by the user to take effect.

The user has the ability to modify the values used for the ring strain test. The default values are

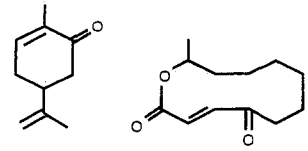
MECYP	ON	methylene cyclopropane
TRIP	8	triple bond in ring
ALLE	9	allene bond in ring
SP2	5	ring of sp^2 atoms
MN0	4	[*m.n.*0] system, size = $m + n$
MN0DB	6	[*m.n.*0] w/ double bond, size = $m + n$
MNP	4	[*m.n.p.*] system, size = $m + n + p$
BREDT	8	[*m.n.p*] w/ bridgehead double bond, size = $m + n + p$
PHANE	10	[*m.n.p*] w/ two bridgehead double bonds, size = $m + n + p$

All numbers give the smallest size that will not be rejected. The user can examine this list and modify the values or disable all or part of it.

As each constraint is entered, it is stored in a buffer where it is available for further examination and editing by the user. When the user gives the END command, the buffer is written to a file that can be recalled and edited in a subsequent run. The constraints are processed, the ACF short list is read, and structure generation is initiated.

Example Problems. To illustrate the scope of COCOA, following are several examples of structure elucidation problems taken from the primary literature on natural products. For each of these problems, the input for COCOA consisted of the molecular formula, the ACF short list from the interpretation program, the ^{13}C NMR data, and the substructure constraints detailed below.

A comparison to the structure assembly program ASSEMBLE[5] may help quantify some of the advantages of structure reduction. Both ASSEMBLE and COCOA were given the same problem, which was based on carvone (**1**). The problem was



1 **2**

arranged such that ASSEMBLE would run efficiently (i.e., no required substructure constraints were used). Although the two programs accept information in quite different manners, the input for each program was designed so that each program produced the same structures. Several versions of the problem were run, adding substructure constraints to reduce the number of structures. The results are shown in Table I. While the

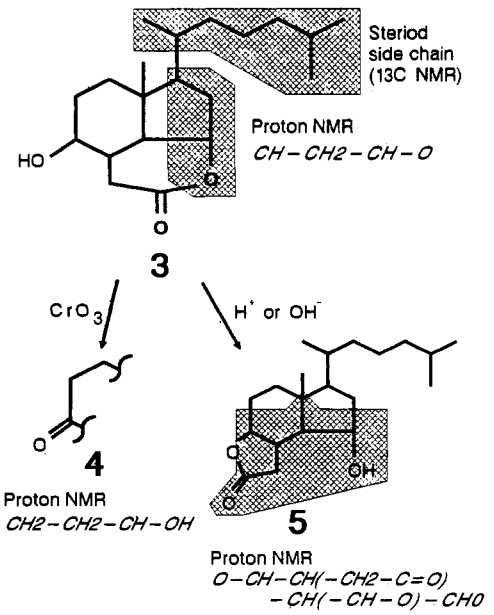


Figure 7. Substructure constraints for aplykurodin.

Harris minicomputer is clearly a faster machine than our Prime 450, the important observation is how the times change as constraints are added. As we anticipated, COCOA handles additional constraints efficiently. However, due to the lack of interaction between constraints, ASSEMBLE uses more time as constraints are added, even though fewer structures are produced.

The next problem is a simple demonstration of overlapping substructures and ambiguous atom types. The natural product is patulolide (**2**),[17] and the constraints used (mostly from ^{1}H NMR interpretation) were (1) CH3–CH–O (a downfield quartet), (2) AH0—CH=CH—AH0 (an isolated AB vinyl pattern), and (3) C(=O)—OH0 (ester or lactone). The AH0 atoms of constraint 2 can be any atom bearing no hydrogens. Structure **2** is the only one produced from this input and the ACF short list. Examination of **2** shows that constraints 1 and 2 overlap with constraint 3.

For a more complex problem such as aplykurodin (**3**),[18] more powerful constraints are necessary (Figure 7). The presence of a steroid side chain was determined by comparison of the ^{13}C NMR data to those of other steroids. This constraint demonstrates the use of the ^{13}C labeling feature. Another constraint is derived from ^{1}H NMR decoupling. Two more constraints are obtained from decoupling experiments on derivatives, as shown in Figure 7. Note that these constraints have been written to reflect the changes made in the derivative; for example, the fragment CH_2—CH_2—C=O was found in **4**, but since **4** was obtained by oxidation of the nonketonic natural product **3**, the constraint used in COCOA is CH2—CH2—CH—OH, since this must be the corresponding fragment in **3**. With these constraints, only structure **3** is produced by COCOA.

The Prime 450 CPU times used for these two problems are 3.2 and 178.3 min, respectively.

PROGRAM IMPLEMENTATION

The COCOA program was developed at Arizona State University on a Prime 450 minicomputer running under Primos 18.3 and is currently in operation on the Prime and on a Harris minicomputer at The Upjohn Co. It is written in Pascal with a couple of FORTRAN subroutines to perform bit operations on sets. The program is broken into three modules: GETCON, which parses and processes the substructure constraint information (about 1000 lines); TESTBAM, which tests each problem state during ACF and bond selection (about 1000 lines); and the main module COCOA, which contains all of the depth-first search routines and performs the canonical naming of the final structures (about 2000 lines).

The bonding adjacency matrix (BAM) is implemented as a one-dimensional array of Pascal sets. Membership of one bonding site in another's set indicates a possible bond connecting the two bonding sites. The program is currently dimensioned to handle up to 50 non-hydrogen atoms and 128 bonding sites (which would give a 50-atom structure with 15 rings), which in turn requires a minimum set size of 128 from the Pascal compiler (the set size of the Prime and Harris Pascal compilers is 256 and 144, respectively).

To improve the efficiency of the program, a FORTRAN subroutine was written to quickly return sequential members of a Pascal set. This NEXTMEM routine examines large sections of the set [32 members (bits) on the Prime, 24 on the Harris], and if the section is nonzero, the first member of the section is found by a binary search using masks. On successive calls, previously returned members are masked off.

ACKNOWLEDGMENT

We thank the National Institutes of Health (NIGMS) and The Upjohn Co. for their generous financial support. Time on the Harris computer was kindly provided by The Upjohn Co.

REFERENCES AND NOTES

(1) Carhart, R. E.; Smith, D. H.; Gray, N. A. B.; Nourse, J. G.; Djerassi, C. *J. Org. Chem.* **1981**, *46*, 1708.
(2) Abe, H.; Okuyama, I. F.; Sasaki, S. *J. Chem. Inf. Comput. Sci.* **1984**, *24*, 220.
(3) Bremser, W.; Fachinger, W. *Magn. Reson. Chem.* **1985**, *23*, 1056.
(4) Dubois, J. E.; Carabedian, M.; Ancian, B. *C. R. Seances Acad. Sci., Ser. C* **1980**, *290*, 383.
(5) Shelley, C. A.; Munk, M. E. *Anal. Chim. Acta* **1981**, *133*, 507.
(6) Lipkus, A. H.; Munk, M. E. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 9.
(7) Sasaki, S.; Fujiwara, I.; Abe, H. *Anal. Chim. Acta* **1980**, *122*, 87.
(8) Christie, B. D.; Munk, M. E. *Anal. Chim. Acta* **1987**, *200*, 347.
(9) Shelley, C. A.; Munk, M. E. *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 247.
(10) Orê, O. *Graphs and Their Uses*; Random House: New York, 1963.
(11) (a) König, D. *Mat. Fiz. Lapok.* **1931**, *38*, 116. (b) Hall, P. *J. London Math. Soc.* **1935**, *10*, 26.
(12) Sussenguth, E. H., Jr. *J. Chem. Doc.* **1965**, *5*, 36.
(13) Wipke, W. T.; Dyott, T. M. *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 3.
(14) Munk, M. E.; Lind, R. J.; Clay, M. E. *Anal. Chim. Acta* **1986**, *184*, 1.
(15) Nijenhuis, A. *Combinatorial Algorithms*; Academic: New York, 1978.
(16) Gonzales, R. C.; Thomason, M. G. *Syntactic Pattern Recognition*; Addison-Wesley: London, 1978.
(17) Sekiguchi, J.; Kuroda, H.; Yamada, Y.; Okada, H. *Tetrahedron Lett.* **1985**, *26*, 2341.
(18) Miyamoto, T.; Higuchi, R.; Komori, T. *Tetrahedron Lett.* **1986**, *27*, 1153.