Journal of Computer-Aided Molecular Design, 4 (1990) 271–282
ESCOM

J-CAMD 085

Automated conformational analysis: Algorithms for the efficient construction of low-energy conformations

Andrew R. Leach*, Keith Prout** and Daniel P. Dolata***
Chemical Crystallography Laboratory, 9 Parks Road, Oxford OX1 3PD, U.K.

Received 30 August 1989
Accepted 4 December 1989

Key words: Conformational analysis; Template joining; Artificial Intelligence; PROLOG

SUMMARY

The method of constructing low-energy conformations using template joining can provide an efficient means of searching the conformational space of molecules. The simplest algorithm to perform this task would construct each potential conformation from scratch. However, new algorithms, some of which use techniques from Artificial Intelligence, have been developed which can greatly improve the efficiency of this approach.

INTRODUCTION

WIZARD is a computer system which applies Artificial Intelligence methodology to the analysis of molecular conformations. WIZARD has been described previously [1]. In the WIZARD analysis of a molecule, specified by its atom types and connectivity, important conformational features such as bond orders, stereocentres, rings, aromaticity etc. are recognised and the *conformational units* are determined. A conformational unit is a group of connected atoms about which the system has some knowledge. For example, the conformational units discovered by WIZARD in *N*-allyl-3,6β-dihydroxymorphinan are illustrated in Fig. 1. From a knowledge of the units present and their connectivity a high-level abstract description of the molecule is then constructed, as shown in Fig. 2. A molecular conformation is first suggested at the abstract level by assigning a symbolic conformation to each unit (e.g. 'chair' is assigned to unit 1). Each symbolic conforma-

*Present address: Computer Graphics Laboratory, 926 Medical Sciences, University of California, San Francisco, CA 94143, U.S.A.
**To whom reprint requests should be addressed.
***Present address: Department of Chemistry, University of Arizona, Tucson, AZ 85721, U.S.A.

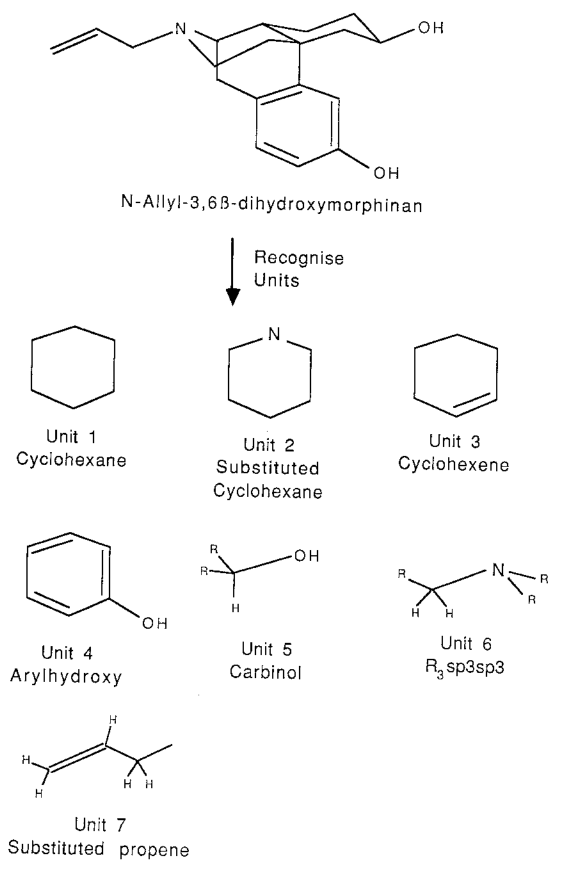

Fig. 1. Units discovered in *N*-allyl-3,6β-dihydroxymorphinan.

tion has an associated *template* file, which contains the atomic coordinates. From these a 3-dimensional structure of the whole molecule is then constructed by joining together the appropriate templates. It may be possible to join a template in more than one orientation; these are given by its *symmetry bindings*. For example, the chair conformation of cyclohexane has two such bindings, as represented by the two structures shown in Fig. 3. In order to search the entire conformational space of the molecule, WIZARD must consider all possible combinations of unit conformations and orientations. Suppose that a molecule has N units, each unit has T_i coordinate template files and that each of these templates has B_{Ti} symmetry bindings. The total number of *subconformations* (template/binding combinations) for the ith unit is given by:

$$S_i = \sum_{1}^{T_i} B_{Ti}$$

Hence the total number of possible conformations, P, for the molecule is:

$$P = \prod^{N} S_i$$

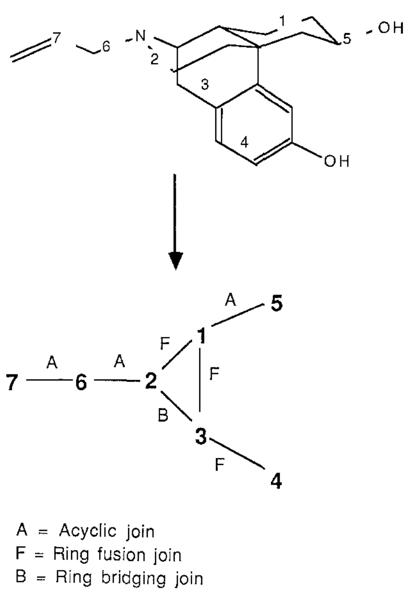

Fig. 2. High-level description of N-allyl-3,6β-dihydroxymorphinan.

Each of these possible conformations is termed a *suggestion* and these suggestions contain the total conformational space of the molecule.

THE INITIAL ALGORITHMS USED TO SEARCH CONFORMATIONAL SPACE

Three distinct processes can be identified in the search operation. The *join-order* algorithm determines the order (stored as the *join-sequence* list) in which the units are to be joined together. The *goal-generation* algorithm is used to produce successive high-level suggestions in which each unit is assigned a symbolic subconformation. The *construction* algorithm takes a high-level suggestion and uses the joining algorithms [2] to build the 3-dimensional conformation it represents.

In the initial method used to perform the search of conformational space the sequence in which the units are joined is calculated before any actual construction occurs and is used for all suggested conformations. The construction starts with the *'crucial unit'*, defined as the most highly connected unit in the molecule. The units which adjoin this crucial unit are then added. When all of the units in this first *shell* have been joined, the units in the second shell (those units separated from the crucial unit by a single unit along the most direct path) are considered. The units in successive shells around the crucial unit are thus considered until all of the units have been added.

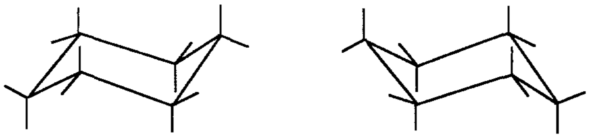

Fig. 3. Symmetry bindings of chair cyclohexane.

This order of joining the units ensures that each unit adjoins (i.e. has atoms and bonds in common with) at least one other unit in the *portion* of the molecule constructed in previous stages. The automatic backtracking provided by Prolog is used by the goal-generation algorithm to generate successive suggestions by varying the subconformations of the units in a systematic fashion, starting with the crucial unit.

As an example of how the process operates, consider a molecule whose unit connectivity is given in Fig. 4. Unit 1 is the most highly connected unit, and is therefore defined as the crucial unit. The first shell contains units 2–5, and so these are the next units to be joined. The second shell contains unit 6 and the third shell unit 7. Hence the order in which the units are joined to construct each suggestion is 1,2,3,4,5,6,7. If the total number of subconformations for unit 1 is S_1, then the first S_1 suggestions considered will have units 2–7 in the first subconformations available to them, while the subconformation of unit 1 is varied. The suggestions from S_1+1 to $2S_1$ will have unit 2 in its second subconformation while unit 1 changes again, and so on until all possible combinations have been described.

Not every suggestion necessarily gives rise to an acceptable conformation. For example, WIZARD generates 99 conformations of *n*-octane; this represents 41% of the 243 suggestions made. The other 59% are all *criticised.* There are a number of reasons why a suggestion may be criticised. Some critics operate before any joining is attempted; for example, the template strain energy critic rejects suggestions for which individual or total template strain energies are too high. During the joining process itself WIZARD uses criticism to reject bad suggestions. Information returned by the joining algorithm is used by the *fit critic* to determine whether a join is satisfactory (i.e. how well the templates fit together). The *vdw critic* criticises each portion as it is constructed on the basis of its non-bonded energy. Both these critics operate after each unit is joined and the suggestion is rejected immediately a problem is detected.

Of the many possible areas for the improvement of the search of conformational space three are identified in this work and improvements suggested:

(1) There is redundancy in the search of conformational space when the conformations of some units change but those of other units remain unchanged. This problem is approached in a new fragment-based construction algorithm.

(2) The order in which the units are joined and the order in which the suggestions are generated.

(3) The construction of molecules which have groups of fused and/or bridged rings is inefficient and has led to the development of an entity-based searching algorithm.

(1) FRAGMENT-BASED CONSTRUCTION ALGORITHM

For many molecules the conformations generated by WIZARD do not vary greatly in the unit subconformations used for their construction. Much of the work involved in joining the units is

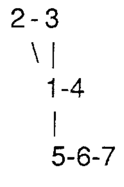

Fig. 4. Unit connectivity graph.

thus repeated a significant number of times. For example, the various conformations of a steroid all have the same structure for the ring skeleton and variations arise solely from the different conformations of the side chains. Rather than building each suggestion unit by unit it would be more efficient to transfer the ring skeleton as a single *fragment* from a previously constructed conformation and use it to build other suggestions. This process of 'cutting out' fragments which contain one or more conformational units should be more efficient than building each suggestion from scratch because the amount of work to perform a join does not increase linearly with the number of atoms involved. For example, the cpu time required to perform two joins between three units is greater than the cpu time required to perform a single join between one of the units and a fragment containing the other two units. It was therefore decided to devise an algorithm which would permit such a transfer of fragments from conformations already constructed. The fragment-based construction algorithm uses fragments taken from the most recently constructed conformation (the *latest generated conformation* or *LGC*). These are recorded in the *current fragment list*.

The first conformation is still built by joining single units, as before, and is then defined as the LGC. The details of the subconformations used to construct it are stored in the Prolog database, and the coordinates in an internal array. As each new conformation is generated, it becomes the LGC and the database and internal array values are updated. To construct a new suggestion, each unit is considered in turn and the subconformation assigned to it is compared with the subconformation of the same unit in the LGC. If a new subconformation is required then the unit is joined separately by reading the coordinates from the appropriate template file, as in the original method. However, if the unit has the same subconformation in the current suggestion as in the LGC (i.e. it is *unchanged*) then it is compared with the units stored in the *current fragment list*. This list contains those units (if any) which come immediately before the current unit in the join-sequence list and are (a) all in unchanged subconformations and (b) form a connected fragment of the molecule. If the current unit is itself unchanged and if it adjoins at least one of the units in this list then it need not be joined at this stage, but is added to the list. The next unit is then considered. The current fragment list is extended in this way until one of three situations arises: (a) the next unit requires a different subconformation from that in the LGC, (b) the next unit, although in an unchanged subconformation, does not adjoin any of the units in the LGC, or (c) there are no more units to be joined. In each of these cases the units contained in the current fragment list are extracted from the LGC and the fragment they define is joined to the previously constructed portion of the molecule. In (a) and (b), in which there are more units to be added, the units which have just been joined are removed from the current fragment list and the construction continues.

The importance of using only *connected* groups of units in the current fragment list can be illustrated by considering the construction of 1,4-di-isopropyl cyclohexane (Fig. 5). Suppose the LGC represents the conformation shown in the figure (with the cyclohexane unit in the chair conforma-

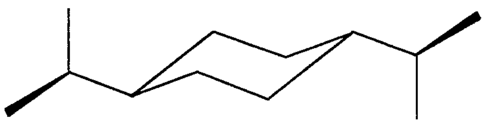

Fig. 5. Di-isopropyl cyclohexane.

tion), and that the current suggestion requires the same subconformations for the isopropyl groups but a twist boat conformation for the cyclohexane unit. It is not feasible to extract the coordinates of both (unconnected) isopropyl groups and join them in a single step to the twist boat cyclohexane, for the atoms in the isopropyl units will still be oriented as though joined to a chair cyclohexane. However, the new conformation can be built in two steps by defining two fragments, each containing one isopropyl group, which are then joined to the new subconformation for the cyclohexane unit in successive joining steps.

As an illustration of how the fragment construction algorithm operates, consider the construction of a molecule whose unit connectivity is as given in Fig. 4. Suppose the next suggestion to be built requires units 2, 3, 4, 6 and 7 to have the same subconformations as they adopt in the LGC and that units 1 and 5 are in different subconformations. The order in which the units are considered is still 1 through 7 as before, and the current fragment list contains no units at the start. Unit 1 requires a new subconformation and so it is loaded from the appropriate template file. Unit 2 is unchanged and so it is added to the current fragment list. Unit 3 is also unchanged, and as it adjoins unit 2 it is also added to the current fragment. Unit 4, although unchanged, does not adjoin either unit 2 or unit 3. Hence it cannot be added to the current fragment. The coordinates of the atoms in units 2 and 3 are extracted from the LGC and the fragment they define is joined to unit 1. The current fragment is now redefined to contain unit 4 alone. Unit 5 requires a new subconformation. Two joins are performed. First, unit 4 (extracting the coordinates from the LGC), and then unit 5 (in its new subconformation). The current fragment is now defined as the empty list. Unit 6 is unchanged and is added to the current fragment. Unit 7 is also unchanged and as it adjoins unit 6 it is also added to the current fragment. There are no more units to be considered and so the fragment defined by units 6 and 7 is extracted from the LGC and joined, to give a conformation of the entire molecule. This example illustrates some of the possible situations which can arise; in general the action to be taken depends upon the status of the current fragment list (empty or containing units), the conformation required for the current unit (unchanged or requiring a new subconformation), whether there are any units still to be considered, and whether any portion of the molecule has been constructed in previous stages. A more complete description is available elsewhere [3].

Results

Thirteen molecules were selected to test the new algorithm (Fig. 6); these form a representative sample of the kinds of molecules currently being analysed using WIZARD. For each molecule the total cpu time required to search the conformational space using the initial algorithms is shown in column 1 of Table 1; the corresponding cpu times using the fragment-based construction algorithm are given in column 2 of Table 1. Time savings are obtained for all the cases considered, with the maximum improvement being some 50%.

(2) THE SEQUENCE OF JOINING UNITS AND THE ORDER OF GENERATING SUGGESTIONS

(a) A new join-order algorithm

There may be many different sequences in which the units could be joined. The only requirement is that each unit always adjoins at least one of the units in the assemblage constructed in pre-

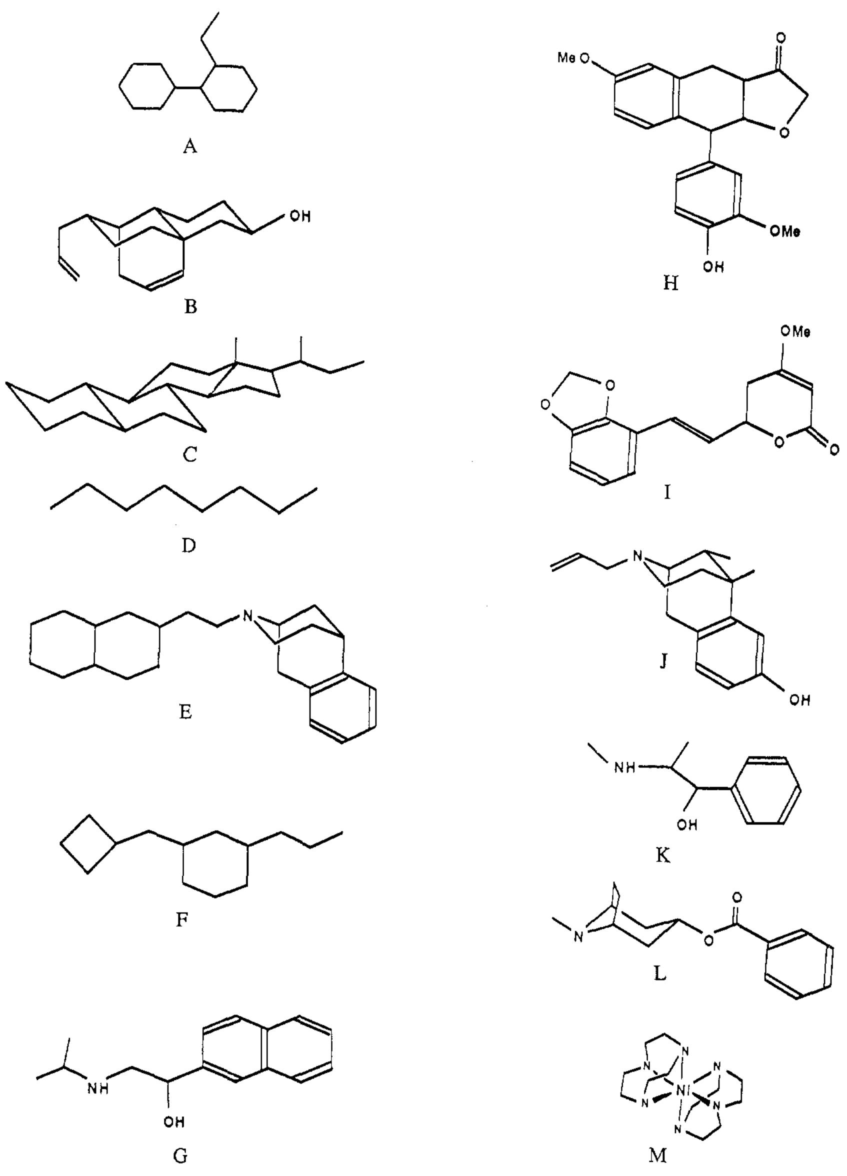

Fig. 6. Molecules used to test the algorithms.

vious steps. The aim is to use an algorithm for determining this order which gives the most efficient search for a wide range of molecules. Two distinct operations can be identified. First, the crucial unit (with which the construction commences) is chosen. Then the order in which the other units are joined is determined. The initial algorithm devised for generating this join-sequence list defines the most highly connected unit as the crucial unit and then adds the other units one shell at a time, in a breadth-first fashion. It was felt that this approach might not be particularly suited to the fragment-based construction algorithm which transfers *connected* fragments of unchanged units. Consider, for example, the construction of the molecule in Fig. 7. Here, the cyclohexane

TABLE 1
A COMPARISON OF THE CPU TIMES REQUIRED TO SEARCH THE CONFORMATIONAL SPACE OF 13 MOLECULES

Molecule	Cpu times (s)					
	1	2	3	4	5	6
A	33.2	30.2	30.0	27.2	23.4	24.4
B	116.1	79.3	85.6	80.0	84.4	20.8
C	199.4	161.5	154.9	181.2	163.5	20.8
D	278.6	182.9	181.7	139.5	138.5	130.8
E	369.1	259.4	250.5	231.7	194.0	24.9
F	326.2	246.0	245.6	186.6	185.0	179.7
G	388.1	271.7	199.7	179.6	140.4	131.3
H	495.5	351.1	250.2	274.2	212.9	78.0
I	94.5	45.8	41.9	42.3	45.0	37.7
J	82.8	63.1	47.4	57.4	38.3	18.1
K	96.4	92.1	70.0	65.4	52.9	48.7
L	63.6	49.3	53.3	53.5	57.7	20.1
M	1093.1	754.2	592.8	632.3	404.1	100.4

1. The originally developed algorithms.
2. The fragment-based construction algorithm.
3. The fragment-based construction algorithm and the entity-based join-order algorithm.
4. The fragment-based construction algorithm and the new goal-generation algorithm.
5. The fragment-based construction algorithm, the entity-based join-order algorithm and the new goal-generation algorithm.
6. The entity-based searching algorithm, the fragment-based construction algorithm, the entity-based join-order algorithm and the new goal-generation algorithm.

unit would be selected as the crucial unit (it is connected to three other units whereas the remaining units have at most two connected units). The cyclopentane unit would be joined next, and then the C-C units, alternately from the two side chains. A suggestion which contains a long sequence of unchanged units will still be constructed one unit at a time because none of these acyclic units adjoins its predecessor in the sequence.

An alternative method was therefore designed to see if an order more amenable to the fragment-based construction algorithm could be determined. This defines the *least* connected unit as the crucial unit. Successive shells of units are then joined until all the units have been added. For many molecules this should mean that larger fragments are transferred. However, this algorithm did not always reduce the search time and sometimes significantly increased it. This reduction in

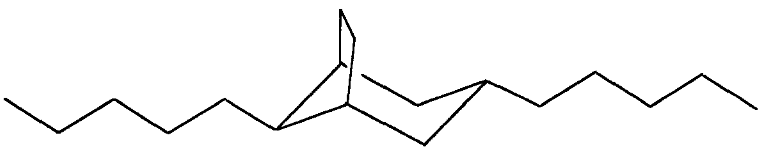

Fig. 7. Molecule in which the fragment joining algorithm is not effective.

efficiency occurred because vdw problems may not be discovered until much later in the construction than when using the original order. When a conformation is criticised because its vdw energy is too high there is often a particularly unfavourable interaction between two units which are separated by a relatively short path in the molecule (one to three units long). By defining the crucial unit as the most highly connected unit and building outwards, suggestions which will be criticised due to such high energy vdw interactions are likely to be discovered (and rejected) early in the construction, so preventing unnecessary work. However, this second algorithm for generating the join-sequence list may not find these vdw problems until much later in the construction.

Consequently, a third method was sought for determining the order of joining the units which would try to incorporate the advantages of the two earlier methods and eliminate their failings. The order is now defined in terms of *conformational entities*. A conformational entity is a group of one or more connected units whose conformations are all linked such that its conformation can be changed without requiring any other entities in the molecule also to change conformation. They were first introduced during the development of the method by which WIZARD relieves strain in criticised conformations [4]. The entities in the molecule are discovered by examining the types of unit connectivity present. A group of cyclic units in which each unit is joined by at least one ring fusion or ring bridging join to another unit in the group is defined as a single entity. Any remaining isolated cyclic units, together with the acyclic units, are defined as single entities. For example, the entities WIZARD finds in *N*-allyl-3,6β-dihydroxymorphinan are shown in Fig. 8. As can be seen, each acyclic unit constitutes a separate entity whereas the four cyclic units which are interconnected by ring fusion or ring bridging joins together constitute one *cyclic entity*.

In the new algorithm to determine the join-sequence list the 'crucial entity' is first discovered;

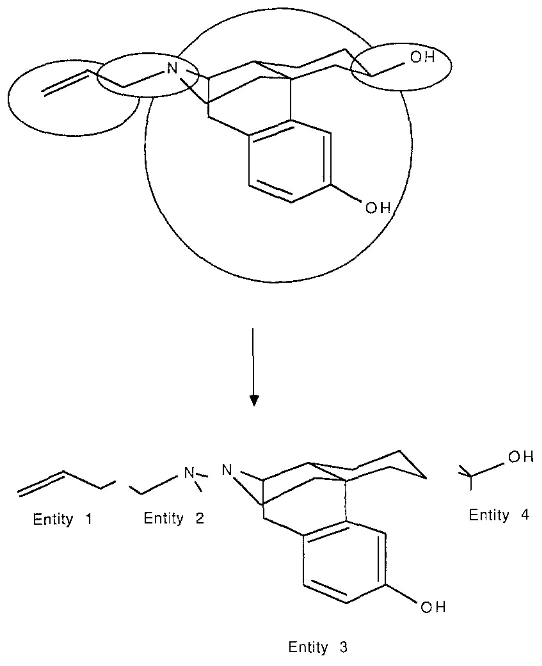

Fig. 8. Entities recognised in *N*-allyl-3,6β-dihydroxymorphinan.

this is defined as the entity which contains the most units. If all of the entities contain just a single unit, the crucial entity is defined as the one which contains the most highly connected unit. The other entities are then added so that the order corresponds to a depth-first search of the entity graph. It was anticipated that this would accommodate the sometimes conflicting requirements of forming large fragments and catching vdw problems as early as possible. Each 'branch' from the crucial entity is considered in turn, and all of the entities on the branch are added before the next branch is considered. For example, in the entity connectivity graph in Fig. 9 the order in which the entities are joined would be [1,2,5,9,10,3,6,4,7,11,8]. The entity sequence is then converted into the equivalent unit join-order list, ensuring that each unit always adjoins at least one previous unit in the list.

(b) A new goal-generation algorithm

The original construction algorithm builds each suggestion completely from scratch, and so the order in which each suggestion is considered is unimportant. However, this is not necessarily the most efficient order to use with the new fragment-based construction algorithm as it is then desirable to have as much similarity as possible between successive suggestions, so that the fragments transferred contain as many units as possible. The fragments are taken from the most recently constructed conformation, and it is therefore preferable to generate the suggestions so that the most highly connected units (e.g. those which are part of a ring system) retain the same conformations for as long as possible. This would then enable such groups to be transferred as a single fragment while the conformations of the other units (e.g. any side chains) vary. In the original goal-generation algorithm the subconformation of the crucial unit is the first to be changed. However, as this unit may be connected to many other units it would be desirable to change its conformation last. The alternative goal-generation algorithm devised therefore changes the subconformations of the units starting with the final unit to be joined.

Results

To determine the effect of the two new algorithms when used in conjunction with the fragment construction algorithm, three sets of experiments were performed using the 13 test molecules in Fig. 6. The results are presented in columns 3, 4 and 5 of Table 1. In column 3 of Table 1 are given the cpu timings using the new join-order algorithm; in column 4 the values using the new goal-generation algorithm; and in column 5 the results using both the new join-order algorithm and the new goal-generation. In each case the fragment-based construction algorithm was employed.

Comparison of column 3 with column 2 shows that the new join-order algorithm did not have much effect on the efficiency of the search for most of the molecules investigated. The greatest im-

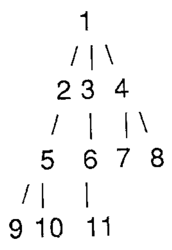

Fig. 9. Entity connectivity graph.

provements in efficiency were for molecules which contained a cyclic entity of the type described above. Similarly, the new goal-generation algorithm alone did not have a significant effect in most of the test cases. However, when the new join-order algorithm was used in combination with the new goal-generation algorithm a more consistent improvement was obtained (contrast the results in column 5 with those in columns 3 and 4).

When using the entity-based join-order method the first units to be joined are those in the entity which contains the most units (i.e. in a cyclic entity, if one is present). It is often found that cyclic entities are limited to a relatively small number of stable structures, and only a small number of the possible combinations of subconformations for their constituent units give non-criticised conformations. This is because many of the combinations do not fit together satisfactorily. Once an acceptable conformation is found for the cyclic entity it would therefore be preferable to transfer it as a single fragment while the conformations of the other units in the molecule vary. Use of the new goal-generation algorithm enables this to occur, for it changes the subconformations of the first units to be joined (i.e. those in the cyclic entity) last.

(3) ENTITY-BASED SEARCHING ALGORITHM

In all of the algorithms described thus far any unsatisfactory combinations of subconformations for cyclic entities continue to be generated even though the suggestions containing them will fail. This is a good example of the 'rediscovery' problem in backtracking algorithms [5]. If the system 'knew' which combinations gave criticised conformations for the groups and which did not, then this information could be used during the search of the molecule's conformational space. Only those suggestions which contained a satisfactory combination of unit subconformations would then be permitted to proceed to construction.

As described in a recent paper on conformational analysis using a truth maintenance system [6] there are two different techniques to solve the rediscovery problem, relaxation techniques and 'caching' techniques. In the relaxation technique the size of the problem domain is reduced by applying some or all of the constraints as filters, either before or during the search. The *entity-based* searching algorithm described below acts in a similar way because there are fewer possible (suggested) conformations of the molecule when the entity description is used. In the caching technique information is stored in the course of solving the problem which is then later used to prevent the re-examination of suggestions which have been shown to contain problems. Work is in progress on a 'learning system' which uses such an approach to improve the efficiency of the search.

The entity-based algorithm for searching conformational space uses a pre-processing step in which the conformational space of each entity is searched, to discover those combinations of unit subconformations which give satisfactory (i.e. uncriticised) entity conformations. These are then stored in the Prolog knowledge base. For entities which only contain a single unit the number of entity subconformations is obviously equal to the number of unit subconformations, but for cyclic entities containing more than one unit, the number of entity subconformations is often much smaller than the number of possible combinations of unit subconformations. When constructing a conformation for the molecule only suggestions for which each entity contains a satisfactory combination of unit subconformations are used. This implies that no suggestion, which will fail because the units comprising the entity do not give a satisfactory (uncriticised) conformation, will ever reach the construction stage.

Results

As the new join-order and goal-generation algorithms gave equivalent or better performance than their original counterparts when used together with the fragment-based construction algorithm, these were utilised when testing the entity-based searching algorithm. Column 6 of Table 1 shows the cpu timings using this combination. As can be seen, this gives even greater improvements in search efficiency, up to 14-fold when compared with the initial algorithms. As would be expected, most improvement was obtained for those molecules which contain at least one of the cyclic entities described above, and the greatest time savings were found for molecules (e.g. molecule E) which have more than one such entity.

CONCLUSIONS

The advantages of WIZARD's logic-based approach in searching conformational space have been outlined previously and the algorithms described above show how AI techniques can be used to further improve the search efficiency. Moreover, by recognising that a molecule's stable conformations are often closely related, it has been possible to make use of the advantages such techniques offer. Of particular importance in this regard are the conformational entities, which further extend the symbolic description of conformation used by WIZARD. The algorithms reported here will permit the analysis of larger and more complex molecules by reducing the amount of computation required to perform the search to reasonable levels.

IMPLEMENTATION DETAILS

WIZARD was developed on a VAX 11/750 minicomputer running the VMS operating system in the Chemical Crystallography Laboratory, Oxford and is written using a combination of PROLOG and FORTRAN. It also runs on other VAX machines under both the VMS and UNIX operating systems. The cpu timings given in Table 1 are for WIZARD running on a VAX 8650 computer. The system is currently being applied to the conformational analysis of molecules containing up to approximately 150 atoms, although the method should be applicable to larger systems. The program can deal with a wide variety of organic molecules and has also been extended to cover some types of transition metal coordination complexes.

A Fortran program (COBRA), which uses a similar approach to the methods described above has now been developed. For further details please contact Dr. David Ricketts at Oxford Molecular, Terrapin House, South Parks Road, Oxford OX1 3UB.

ACKNOWLEDGEMENT

ARL thanks the SERC for a Studentship.

REFERENCES

1 Dolata, D.P., Leach, A.R. and Prout, K., J. Comput.-Aided Mol. Design, 1 (1987) 73.
2 Leach, A.R., Prout, K. and Dolata, D.P., J. Comput.-Aided Mol. Design, 2 (1988) 107.
3 Leach, A.R., D.Phil. Thesis, Oxford University, 1989, pp. 164–190.
4 Leach, A.R., Prout, K. and Dolata, D.P., J. Comput. Chem., 11 (1990) 680.
5 De Kleer, J., Artif. Intell., 28 (1986) 127.
6 Koschmann, T., Snyder, J.P., Johnson, P., Grace, T. and Evens, M.W., J. Mol. Graph., 6(2) (1988) 74.