Short-term learning in conformational analysis

Daniel P. Dolata and W. Patrick Walters

AI in Chemistry Lab, Department of Chemistry, University of Arizona, Tucson, AZ, USA

A method for learning short-term rules of conformational analysis is introduced. The technique works by discovering problems during the building of a conformation in Cartesian coordinate space, and builds an abstract critic suitable for reasoning in abstract symbolic space. The methods not only afford speed increases ranging from 1.0- to 2.3-fold in WIZARD (analysis completed in 100% to 43% of original run time), but can be modified to provide similar increases in other programs that use "template joining" and distance geometry. These methods also provide the basis for a long-term learning project.

Keywords: automated learning, artificial intelligence, conformational analysis

INTRODUCTION

A computer is a machine that executes some program to obtain a result. There is a rough hierarchy of program types:

- Hardwired
- Programmable
- Teachable
- Apprentice
- Explorer

Hardwired systems can be very efficient at their given tasks, but they cannot easily be changed to adapt to new information or new tasks. *Programmable* systems contain a small hardwired section that interprets or runs a provided program. The program can be modified more easily than a hardwired algorithm. However, the modification requires a trained programmer who not only knows the programming language but who also possesses the domain-specific knowledge needed to properly modify the algorithm. In complex domains, this combination is often difficult to find in a single individual. Thus we find teams of programmers and domain experts working together. This can be fairly successful, but there is a well-known problem with the so-called "knowledge transfer barrier."

One method to help eliminate this barrier is to create a *teachable* system.[1] In such a system, the domain expert will teach the program the new knowledge directly, and the program will then modify its own algorithm to reflect the new knowledge. Teachable systems are heavily reliant on the quality of the instruction. Learning errors can arise from false examples, and other problems include insufficient or inconclusive examples, unrelated coincidences without counter examples, etc. Another problem is that the domain expert might not be available or interested in teaching a program. Thus, it is desirable to create an *apprentice* system[2] that can observe examples which were not designed as teaching examples and learn without step-by-step guidance. The apprentice program is limited in that the only examples it can draw might be inconclusive or incomplete. A program which can recognize these deficiencies and devise and solve problems on its own to create good teaching examples is called an *explorer* system.

Our goal is to create such an explorer system for the domain of conformational analysis. This proposed program will learn new general concepts of conformational analysis which will be useful to chemists and programs alike. However, this goal cannot be reached without some intermediate steps and exploratory efforts. This paper reports a preliminary step, which was to provide our conformational analysis program, called WIZARD, with a short-term learning capability. This ability does not provide generalized rules. Instead, it adaptively learns and becomes more proficient during the solution of a single task. The following paper[3] will describe how the short-term data obtained from this work can be generalized to form the basis of a long-term learning system.

BACKGROUND—WIZARD

WIZARD[4] is a conformational analysis program based on the AI techniques of theorem proving.[5] This method begins with a set of axioms of conformational analysis and a set of structural postulates, and applies them to a given molecule to derive a set of conformations. These conformations are theorems which *logically follow* from the axiomatic theory.[6] The conformational axioms control the inferences along generally accepted conformational principles. One of the key axioms is the Building Block Axiom, which states:

> A connected set of atoms and bonds will exhibit the same conformational behavior when they are a subset of a larger molecule in the absence of a deforming force, as they do when they are a subset of a smaller molecule.

Address reprint requests to Dr. D.P. Dolata at AI in Chemistry Lab, Department of Chemistry, University of Arizona, Tucson, AZ 85721, USA.
Received 20 June 1992; accepted 28 July 1992

The structural postulates contain knowledge about such connected sets of atoms and bonds (which we call *units*) which exhibit predictable behaviors. An example of such a unit is the butyl unit, and the postulates contain knowledge about the stable forms such as anti (but_an), + gauche (but_gp) and − gauche (but_gm).

When WIZARD is presented with a molecule for analysis, it begins by identifying the units in the molecule (e.g., butyl or methyl) and categorizing the relationships between the units (e.g., adjoining). The units must show sufficient overlap to control all degrees of freedom. If pentane were analyzed as a methyl group attached to a butyl group, there would be no control over the torsion angles of the methyl group with regard to the rest of the chain. Thus WIZARD analyzes *n*-pentane in terms of two $R_3C\text{-}CH_3$ units and two disubstituted ethyl units as shown in Figure 1. This gives complete control over all degrees of freedom in the molecule.

WIZARD then proceeds to build successive conformations by combining individual conformational templates for each unit. For example, a butyl unit has three subconformations: gauche +, gauche −, and anti. If WIZARD were to simply combine each conformation of each unit without intermediate evaluation of the results, the resulting process could be described by the tree shown in Figure 2.

In this tree, the first level of subconformations are obtained by taking each stable conformation for unit 1. In this example there are three such conformations. The next level is obtained by combining each stable conformation for unit 2 with each stable conformation for unit 1, and so forth until the entire molecule is constructed. While this process is theoretically possible, it is generally computationally intractable. For example, if a molecule has 10 units, each of which has 3 stable conformations, then the number of final conformations would be 3^{10} (59,049). Even if it only took 1/10 of a minute to generate and evaluate each conformation, this would still require over 4 days of CPU time. Many of the conformations obtained in this fashion will not be stable, and some will be downright "silly" (such as those that contain interpenetrating atoms). Uncritical use of this algorithm to generate *n*-pentane from *n*-butane fragments would lead to conformations such as the one shown in Figure 3, where two atoms are far too close to each other. Many of these can be eliminated through criticism. The algorithm can also be made much more efficient by working at various levels of abstraction.

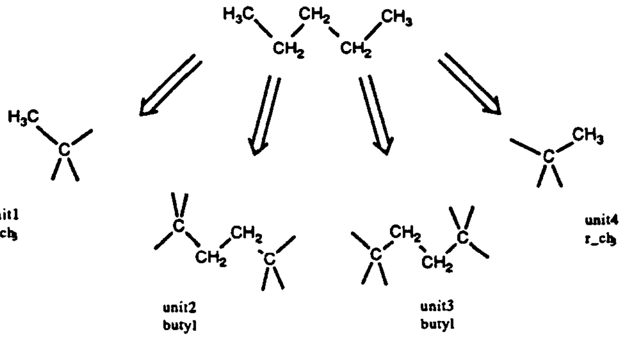


Figure 1. WIZARD's analysis of the "units" in n*-pentane.*

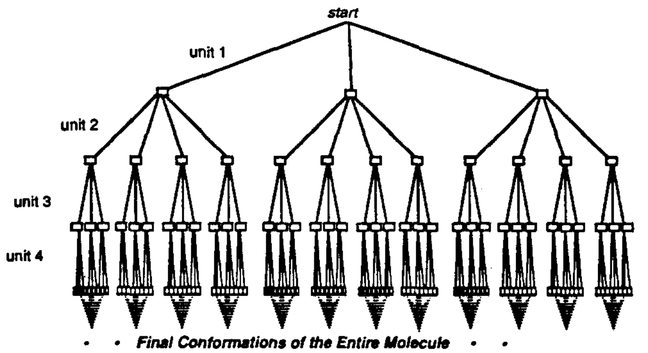


Figure 2. Implicit search tree for WIZARD's synthesis of the subconformations of a molecule.

The knowledge about units and their conformations is stored at several levels of abstraction corresponding to symbolic (e.g., gauche +) and instantiated (e.g., Cartesian coordinates). WIZARD begins working in abstract space by creating a symbolic suggestion. Such a symbolic suggestion is shown below. It corresponds to a common description used by chemists: "gauche + gauche −." Chemists often work in similar abstract spaces, using detailed geometries only when needed. An advantage to working in abstract space is that an abstract suggestion can be generated in a very short period of time, and it can be criticized in an equally short time. Working in Cartesian space takes substantially more time, since there are many more individual objects to keep track of, the calculations must be done in floating number arithmetic, etc. If a potential conformation can be eliminated at the abstract level, we can save substantial amounts of work at the detailed level. WIZARD utilizes a set of abstract critics to see if the deforming forces mentioned in the Building Block Axiom are going to be provably present in the conformation. One such critic corresponds to the pentane rule; a pentyl chain with an adjoining gauche + gauche − pair (butyl_gp, butyl_gm) will be unstable due to hydrogen–hydrogen repulsion, as shown in Figure 3.

If WIZARD fails to criticize the symbolic suggestion, it then proceeds to instantiate the conformation in Cartesian space. It does this by taking templates from a library and joining them together.[4] As it is building this detailed version of the conformation, it applies another set of critics *at each construction step*.[4] The critics include another check for van der Waals exclusion (in case an abstract critic missed something; see below), tests to see how well the templates fit together, tests for excessive electrostatic repulsion, etc.

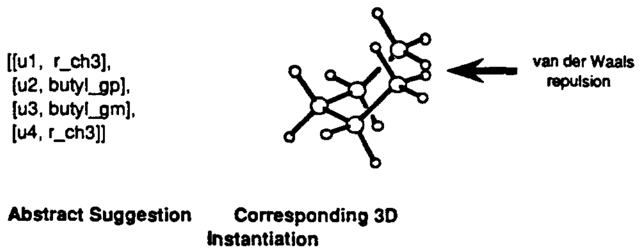


Figure 3. Suggestion and three-dimensional instantiation for an unstable conformation of n*-pentane.*

These critics are important, since a node high in the search tree (i.e., corresponding to the joining of two or three units) may be the parents of hundreds or thousands of child conformations. Conformational analysis shows that problems are inheritable; if the parent is bad, then all of the children will be bad. By discovering these problems early, much fruitless work can be avoided.

Only those conformations which proceed from a suggestion in abstract space to a fully instantiated model in Cartesian space *without criticism at any stage* are presented to the user. The fate of the criticized suggestions are varied; they can be discarded or WIZARD can attempt to *resolve* the problems.[4] The job of short-term learning will be to provide WIZARD with the ability to learn what went wrong with this specific conformation and to avoid making similar mistakes during the rest of the analysis.

SHORT-TERM LEARNING

Since WIZARD works at several levels of abstraction, it would be possible to utilize any of them for the learned description. There are several reasons to work in abstract space:

- The increase in efficiency will be most noticeable if we can prune bad suggestions as early as possible, i.e., at the abstract stage.
- It can be easier to apply a learned critic at a higher level of abstraction.
- The learned critic will be more useful for long-term learning.

These reasons can be demonstrated with the aid of the following example. These are two descriptions of the methyl end of a conformation of a fatty acid which exhibits van der Waals strain due to interpenetrating hydrogens (the pentane rule as shown in Figure 3):

Example 1—Abstract Level:
[unit 1, r_ch3] adjoins [unit2, butyl_gm] adjoins [unit3, butyl_gp] adjoins [unit4, butyl_an], *etc.*

Example 2—Detailed Level:
[atom1, H, X_1, Y_1, Z_1] [atom2, H, X_2, Y_2, Z_2] [atom3, H, X_3, Y_3, Z_3] [atom4, C, X_4, Y_4, Z_4] *etc.*

It can be seen that it will be quicker to apply a simple van der Waals critic by searching an abstract suggestion list for the sublist [any_alkyl][butyl_gm][butyl_gp][any_alkyl] than it would be to perform $(N^2\text{-}N)/2$ distance calculations on N atoms. This is especially true since most units comprise 6 to 20 atoms. Of course, this gain in speed does have some potential drawbacks. Since the abstract suggestions are less detailed than the instantiated suggestions, there is a loss of exact predictive ability. This will show up in one of two ways: abstract suggestions will be criticized when the corresponding detailed conformations would not be, or vice versa. Since we are interested in finding *all* stable conformations, and since working at the abstract level is used to gain efficiency, we have chosen to allow some potentially criticizable suggestions to pass at the abstract level, knowing that they will be caught at the detailed level.

The level of description is also critically important in the eventual goal of our overall project, namely long-term learning.[3] It is tautological that symbolic programs derive their power from the ability to manipulate symbols. If a program has a restricted symbol set, then it will have restricted reasoning power. Imagine trying to learn the pentane rule from a series of examples as shown above: in type-1 examples, minor variations in bond angle, bond length, and torsion angle will disappear during the abstraction step. The term *gauche* + can include a variety of torsional angles from +50° to +70°. Thus a series of examples will give a rule of the form [any_alkyl][butyl_gp][butyl_gm][any_alkyl], even if there is slight variation in the exact angles. This is essentially the method and language commonly used by human chemists to describe this rule. Combining a series of examples of type 2 would yield something that resembled a common volume map, which is not easy to use in learning general principles and is not easy for human chemists to use. Having demonstrated that it is best to perform short-term learning in abstract space, we will show how WIZARD performs such learning.

During the construction process WIZARD criticizes each substep. When a subconformation is criticized, the underlying problem can be broadly classified into one of two classes: sudden problems and aggregate problems. In sudden problems, the critic recognizes that the problem has arisen suddenly, while our definition of aggregate problems is that they change by less than 50% in each step. For example: WIZARD estimates the amount of van der Waals strain in each construction step. Let us assume that the critic is set to reject subconformations which have more than 20 kcal of estimated strain. If the van der Waals strain estimate rises from 4 kcal to 40 kcal, this is considered to be a sudden problem. Sudden problems are generally due to a small number of strong interactions between two (or more) features. A sudden problem might arise from the interpenetration of two hydrogens (e.g., the "pentane violation" above); subconformers consisting of two and three units would show no problems during construction, but the addition of the last unit would show a dramatic sudden increase in vdW repulsion.

An aggregate problem is one which has been present all along, and has steadily crept towards a critical level. If the estimated vdW strain of a subconformer rose from 19 kcal to 21 kcal after addition of a new unit, this would be considered to be an aggregate problem. Aggregate problems often have no single root cause, but are the summation of a larger number of minor problems.

Because of the causal difference between sudden and aggregate problems, they are handled in different fashions. Since a sudden problem generally arises from a clearly defined small set of interactions, WIZARD attempts to find what that interaction is and to find a minimal set of abstractly defined units that contain the interacting portions and the path between them. This minimal path is used as a critic. Aggregate problems cannot generally be partitioned and so they are treated as a whole entity, and the entire abstract suggestion is used as a learned critic. The difference is shown in Figure 4.

Figure 4 represents a small fraction of WIZARD's search tree for conformations of the given molecule. Each subconformation would have from 3 to 10 children, but only 1 or 2 are shown for the purposes of this example. Subconformation *A* is criticized due to the van der Waals

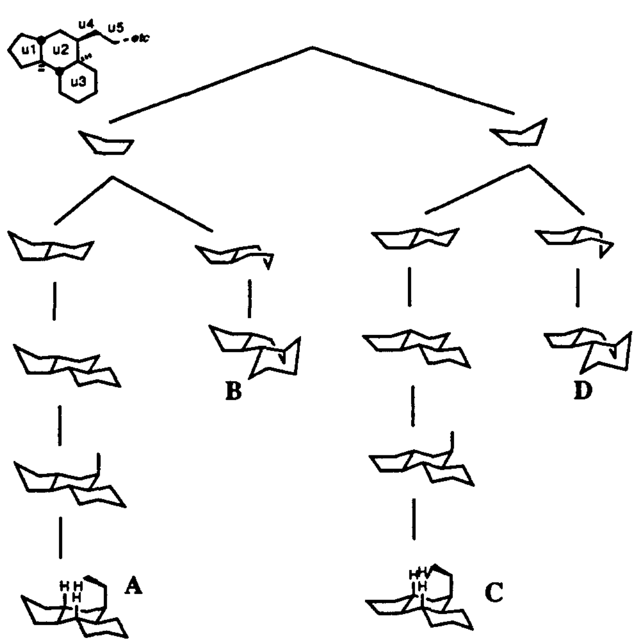


Figure 4. As the tree is explored, subconformations will be built which are similar enough to predictably exhibit similar problems.

repulsion between the labeled hydrogens. This is a sudden problem, since the subconformation which immediately proceeded *A* did not exhibit substantial van der Waals problems. WIZARD then discovers the minimal set of units which will describe the problem.

WIZARD is written in a mixture of PROLOG and a procedural language such as Fortran (WIZARD-II) or C (WIZARD-III). The prolog predicate criticize_vdw calls a procedure which estimates the van der Waals forces through simple distance calculation. This procedure also keeps track of the worst offenders. Thus WIZARD can call upon that knowledge to determine which atoms are at fault. In this example, atoms H_1, H_2, and H_3 are implicated. Figure 5 shows how WIZARD begins with the last unit added, and finds the minimal path of units which contains all of the offending atoms.

The minimal abstract suggestion corresponding to that minimal path is then extracted from the abstract suggestion for the entire subconformer, and is used as a short-term critic. This critic can be applied at the abstract suggestion stage before any geometrical instantiations are performed. In Figure 4, the abstract suggestion which leads to subconformation *C* is criticized, and none of the work required to build it need be performed. In highly branched or constrained molecules, this form of learning can lead to substantial CPU savings, since a problem found in one branch of the search tree (e.g., *A*) can be applied to many other branches (e.g., *C*). Since the shortest path extraction algorithm is simple, benefits of adding this feature far outweigh the costs. In very complex molecules we have seen fivefold increases in speed. However, speed improvements of 1.2- to 1.5-fold are more common in most alkaloids and terpenoids.

Aggregate problems are not amenable to the same treatment. An aggregate problem is shown in Figure 4 in subconformer *B*. The two ring fusions between units 1, 2, and 3 are not particularly well matched. The first is the fusion of the base of the envelope conformation of cyclopentane (dihedral ~0°) to the staggered part of a boat (dihedral = −56°). The second fusion consists of joining a chair (dihedral = 54°) to the nearly eclipsed portion of a twist boat (dihedral = 30°). While the second fusion has clearly pushed the problem over the limit, it is impossible to find any subportion of the subconformer which is uniquely at fault. So the abstract suggestion for the entire subconformation must be used as a critic. But because of the nature of the search, the only abstract suggestions which will match this new abstract critic will lead to subconformer *B*. Although subconformation *D* suffers from an analogous fault, these paths will not be criticizable by the knowledge gained from *B*. Thus, this form of short-term learning does not produce significant increases in speed. However, a long-term learning program (such as the one reported in the following paper) can generalize these critics for use in future searches.

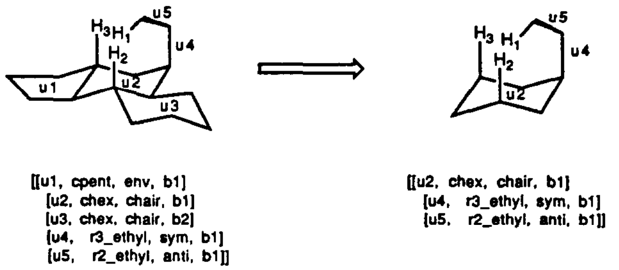


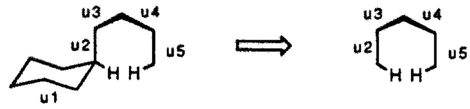
Figure 5. Finding the minimal critical path for a problem.

Some of the results are shown in Table 1. Pentane does not show any increase in speed (in fact, the speed decreased by about 1%). There are only 4 units in pentane, CH_3-R, X-CH_2-CH_2-Y, X-CH_2-CH_2-Y, and R-CH_3. Only the central two have any degrees of freedom, i.e., anti (a), gauche + (g +) and gauche − (g −). In pentane, the entire molecule is necessary to exhibit the critical behavior, so each criticized conformation will be entirely unique. Out of the 9 possible conformations (aa, ag +, ag −, etc.), only 2 (g + g −, g − g +) are criticized. When the first conformation (for example, g + g −) is criticized an abstract critic is learned, but it will never be applicable. Since WIZARD doesn't recognize symmetry, WIZARD cannot infer that g − g + will be equivalent to the previously learned critic g + g −. Short-term learning will only show an improvement in behavior when the learned critic is a subset of the total molecule. For example, if the terminal methyl were replaced by a cyclohexyl ring, then the various suggestions for the molecule such as [*ax*-chair, g +, g–], [*eq*-chair, g +, g–], [twist-boat, g +, g–], etc., would be criticizable. This is shown progressively in the next examples.

In the case of *n*-butylcyclohexane, WIZARD finds four units with degrees of freedom: a cyclohexyl unit (u1), a X_2-CH-CH_2-Y unit (u2), and two X-CH_2-CH_2-Y units (u3 and u4). The critic learned from the conformation shown in Figure 6 will be able to eliminate other conformations which differ in the conformation of unit 1. The short-term learning facility was able to use this critic to eliminate 17 conforma-

Table 1. Increase in speed due to short-term learning

Molecule	Number of sudden problems	Number of aggregate problems	Increase in speed (CPU normal/CPU learning)
n-pentane	2	0	1.0
n-butylcyclohexane	4	0	1.3
2,4-dimethylhexane	4	0	1.8
3,5-dimethylheptane	4	0	1.9
1-methyladamantane	4	0	1.2
1-ethyl-3-methyladamantane	4	0	1.8
1-butyl-3-methyladamantane	8	0	2.3
cyclazocine	10	12	1.3
progesterone	17	32	1.4
cocaine	23	8	1.2



```
[[unit1, chex, chair, b1]
  [unit2, r3_ethyl, sym, b1]
  [unit3, r2_ethyl, g+, b1]
  [unit4, r2_ethyl, g-, b1]]
```

```
[[unit2, r3_ethyl, sym, b1]
  [unit3, r2_ethyl, g+, b1]
  [unit4, r2_ethyl, g-, b1]]
```

Figure 6. The first criticized conformation of 1-(n*-butyl)-cyclohexane provides an abstract critic which doesn't depend upon the conformation of the cyclohexyl ring since the H atom is determined by both unit 2 (which is retained) and unit 1 (which can be discarded).*

tions at the abstract level. This afforded a 1.2-fold increase in speed. The molecules 2,4-dimethylhexane and 3,5-dimethylheptane showed more significant speed increases of 1.8 times and 1.9 times, respectively.

A series of increasingly complex adamantane derivatives shows a similar trend towards increasing efficiency with increasing complexity. There are two foci for learning in 1-alkyl-3-methyladamantane. Without learning, WIZARD repeatedly made false starts on building the adamantyl cage. In addition, there are the same sort of van der Waals interactions between the rings and the side chains that were found in *n*-butylcyclohexane. WIZARD was able to learn a new abstract critic from the first example that was criticized at the instantiated level, and to use that new abstract critic to avoid repeating the mistake. Methyladamantane showed a 1.2-fold increase in speed with learning, while 1-ethyl-3-methyladamantane showed a 1.8-fold increase in speed. In 1-butyl-3-methyladamantane the resulting contraceptive pruning of bad examples at the abstract stage affords an overall 2.3-fold increase in speed.

Small molecules which contain highly branched acyclic portions and cage structures provide the greatest opportunities for increases in speed due to short-term learning. More "common" alkaloids and terpenoids show significant, but smaller, improvements. Molecules such as cocaine, progesterone, and cyclazocine show from 1.2- to 1.4-fold increases in speed when short-term learning is used.

CONCLUSIONS

The short-term learning facility has two significant advantages:

- It provides an increase in speed from 1.0- to 2.3-fold, averaging 1.52-fold (65% of the original time).
- It provides short-term critics which can be utilized as data for a long-term learning program.

The short-term learning facility adds a significant improvement in WIZARD's ability to perform conformational analysis. However, the basic principles behind the concept of short-term learning is not limited to WIZARD. The technique could be used to enhance many existing structural programs. For example, a distance geometry program could be equipped with a critic and learning facility. The distances could be abstracted by converting the exact length representation into a semiabstract set of distance ranges. When a set of distance ranges leads to a criticizable problem, a short-term learning facility could determine which subset of ranges were responsible and put them on a list of forbidden combinations. Similar applications of short-term symbolic learning could be applied to programs which try to build peptide conformations by rotation of multiple phi-psi angles, etc. The key to implementing short-term learning is that the program must be partitioned into the following phases: hypothesizer, optional abstractor (which is needed if the hypothesizer works in detailed space), contraceptive abstract critic, instantiator, learning critic. The learning critic observes the output of the instantiator, and passes knowledge back to the abstract criticizer to contraceptively prevent mistakes.

The conformational short-term critics that WIZARD learns by this technique are not directly applicable to more than one analysis since they contain direct references to atom numbers, unit numbers, and rely on implicit molecular connectivity. However, the critics are formulated in such a fashion that they are suitable for examination by a generalization program, such as the one reported in the following paper. While increasing the speed of a modeling program by anywhere from 0% to 120% may seem like the best result of this technique, the learned critics may well prove to be the most important product of short-term learning. Generalizations of these critics into rules of conformational analysis will extend the knowledge and abilities of both human and mechanical conformational analysts. These rules will outlive both WIZARD and MOUSE, and constitute the final product of our learning project.

ACKNOWLEDGMENT

The authors would like to thank the Arizona Affiliate of the American Heart Association for financial support through grant BG-2-30-90.

REFERENCES

1 Tanimoto, S.L. *The Elements of Artificial Intelligence* Computer Science Press, Rockville, 1987

2 Michalski, R.S., Carbonell, J.G., and Mitchell, T.M. *Machine Learning* Vols. 1 and 2, Morgan Kaufmann Publishers, Palo Alto, 1986

3 Dolata, D.P. and Walters, W.P. MOUSE, A Teachable Program for Learning in Conformational Analysis. *J. Mol. Graphics* 1993, **11,** 106

4 Dolata, D.P. Molecular Modeling By Symbolic Logic. In: *Collected Papers from the 3'rd European Seminar on Computer Aided Molecular Design* IBC Ltd, 1987; Leach, A.R., Dolata, D.P., and Prout, K. Automated Conformational Analysis and Structure Generation: Algorithms for Molecular Perception. *J. Chem. Inf. Comp. Sci.* 1990, **30,** 316; Leach, A.R., Prout, C.K., and Dolata, D.P. The Application of AI to the Conformational Analysis of Strained Molecules. *J. Computational Chem.* 1990, **11,** 680

5 Amble, T. *Logic Programming and Knowledge Engineering* Addison-Wesley, Menlo Park, 1987

6 Stoll, R.R. *Set Theory and Logic* Dover, New York, 1963