MOUSE-III: Learning rules of conformational analysis from X-ray data

W. Patrick Walters and Daniel P. Dolata

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

MOUSE-III is learning program that finds rules of conformational analysis from raw crystallographic data. The program perceives molecular features, finds conformational classes in the data and then learns rules that link features to classes. The rules that MOUSE learns are capable of correctly assigning conformations to ring systems that were not used for training with greater than 95% accuracy, when MOUSE was presented with sufficient data. The rules also show a compression of as much as 99% when compared to the raw data. This is accomplished through abstraction and generalization. The algorithm is presented along with a carefully worked example. An example of a learned rule is also presented and analyzed. Some conclusions about the scope and limitations of the learning process are presented.

Keywords: MOUSE, artificial intelligence, learning, rule formation, conformational analysis, ring systems

INTRODUCTION

The process of extracting laws of nature from data has been a central goal of scientists for hundreds of years. The ability of scientists to derive laws typically comes about through a combination of serendipity, inspiration and hard work. In recent years, researchers have sought methods of automating the search for relationships in data. Among the techniques applied to this task are

- multivariate statistics[1]
- neural networks[2]
- machine learning[3]

While the first two methods often provide insight into complex problems, they currently lack the ability to communicate what they have learned in a form that is easily usable by humans. This paper describes the design and results of MOUSE-III, a program that utilizes machine-learning techniques to derive conformational rules for a number of carbocyclic and heterocyclic ring systems. Although this paper reports the use of MOUSE-III on ring systems, the technology is not limited to rings. The application of the MOUSE technology to other systems of chemical interest (e.g., NMR interpretation, peptides) will be reported in future papers.

Programs that produce classification rules are often referred to as concept-learning programs[4–6] and the logical linkage between features of the class and class membership is referred to as the *concept*. Concept-learning programs operate by examining a set of training examples, each of which is assigned to a particular class, and by deriving a rule or set of rules that will correctly assign new examples to these classes. There are many reasons to implement a concept learning program:

- learn new laws of science usable by humans as well as computers
- increase speed in achieving a given task
- reduce the size of a database

MOUSE-III is capable of achieving all three of these objectives. A number of rules generated by MOUSE-III were examined by experienced organic chemists who found the computer-generated rules both innovative and exciting. The generated rules can be used to predict conformations orders of magnitude faster than other conformational analysis techniques such as stochastic search coupled with molecular mechanics. For large data sets, the rules generated by MOUSE-III typically occupy less than 2% of the space occupied by a comparable set of Cartesian coordinates.

Previous publications[7,8] have demonstrated the utility of prior versions of MOUSE in deriving conformational rules from examples. In those cases, the examples were chosen by a "teacher" in order to aid the rule formation process. MOUSE-I was limited in that it was only capable of handling data that were pertinent and correct. In addition, MOUSE-I required that the learning examples be preclassified by the teacher. In version III of the program, it is no longer necessary for a teacher to choose the examples. The program is also capable of discovering classes and deriving rules for class membership in raw, unselected data that may even contain a small number of errors. As in previous versions of MOUSE, the rules are represented as logical expressions that can be understood and used by chemists as well as computers.

Address reprint requests to Prof. Dolata at AI in Chemistry Lab, Dept. of Chemistry, University of Arizona, Tucson, AZ 85721, USA.
Received 20 July 1993; accepted 10 August 1993

METHODOLOGY

MOUSE-III operates by examining a set of crystal structures, all of which have a particular ring system in common. The program randomly selects a portion of the examples and uses these to derive rules for class membership. The remainder of the examples are then used to test the effectiveness of the learned rules.

There are four components to the MOUSE-III program:

(1) PERCEIVE—In the PERCEIVE module, raw data are transformed into a set of properly formatted facts that can be processed by the learning program.

(2) CLASSIFY—The CLASSIFY module performs two functions: it identifies the distinct ring conformations present in the data set and divides the rings into conformational classes (which chemists would recognize as boat, chair, etc.).

(3) LEARN—The LEARN module creates logical connections between the knowledge created by PERCEIVE and the knowledge created by CLASSIFY. This module compares all rings belonging to a given class with the members of all other classes in the training set. The features found in members of that class, and not found in members of other classes, are used to construct rules. These rules are then re-examined and generalized according to the hierarchy in Figure 1.

(4) TEST—In the TEST module all the rings in the test set are classified according to the rules developed in LEARN. The number of correct, incorrect and undefined classifications is reported. Testing is optional with regard to learning, but necessary for our evaluation of the learning process.

Data acquisition and perception

In each set of molecules examined by MOUSE-III, the learning examples consisted of a set of crystal structures containing the ring system of interest. The data sets were limited to rings containing the elements C, O, N and S as ring atoms. Structures that contained transition metals were removed from the data set. The crystallographic coordinates identified each atom only by its atomic number and Cartesian coordinates. In order to generate rules, MOUSE-III requires that atom types include hybridization. These hybridizations were determined using the BABEL[9] program developed in our laboratory as a preliminary step. BABEL utilizes techniques developed by Meng and Lewis[10] to examine bond lengths and angles in order to assign more specific atom types (e.g., sp2 carbon). After atom types were assigned, all exocyclic atoms were removed, producing a set of structures consisting of only ring atoms.

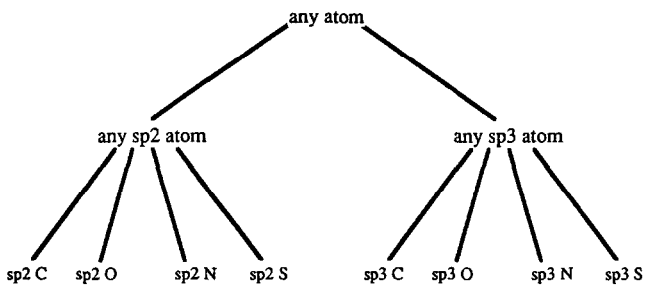

Figure 1. The generalization hierarchy for atom types.

MOUSE-III has taken a simplified approach to predicting ring conformations. It is assumed that the major factor in determining the ring conformation is the hybridization of the atoms that make up the ring. The program currently ignores the presence of repulsive or attractive interactions between functional groups and appendages attached to the ring. In spite of this limitation, the program was able to predict ring conformation with an accuracy ranging from 95–99% in the systems studied. The version of the program that is currently under development (MOUSE-IV) will be able to consider the effects due to side chains.

Ring classification

When the perception process is complete, each ring is represented by a set of Cartesian coordinates and hybridized atom types. In order to generate a set of rules for predicting ring conformation, it is first necessary to divide the rings into classes. These classes will correspond to the predictable conformations. Thus, both the perceived terms (atom types) and the classified terms (ring conformation classes) are determined by MOUSE-III. This represents a significant advancement over previous versions of the program, which required that examples be classified by a "teacher" prior to learning. The rings are classified by finding clusters of conformations and designating the conformation closest to the center of the cluster as the prototypical template for the class. This template provides a computer representation of a typical member of the conformational class. The template can also be used by programs such as WIZARD,[11] which uses a fragment assembly method to perform conformational analysis. The cluster discovery process begins with a one-to-many comparison of each ring with all other rings in the data set. Ring comparison is accomplished through an ordered comparison of dihedral angles for all topologically equivalent mappings of the ring system. All rotational mappings of each ring and its enantiomer are compared with the perspective template. If the sum of the difference in dihedral angles is less than a predetermined threshold, the comparison is considered to be a "match." Each ring is assigned a score corresponding to the number of matches achieved. The ring with the highest score is selected as the first template and all rings that match this template are removed from the data set. The process of template selection and removal of all rings that match the template continues until no unclassified rings remain.

The ring classes discovered by the CLASSIFY module correspond well with conformational classes typically used by chemists to describe ring systems. Figures 2–7 provide representations of the major classes found for the families of rings studied by MOUSE-III.

After all the rings have been classified, the data are transformed into a symbolic representation that consists of a series of PROLOG clauses, each of which represents one ring. Each clause contains a series of slots that represent a specific attribute/value relationship. The ring clauses have one term that contains the class identifier and additional

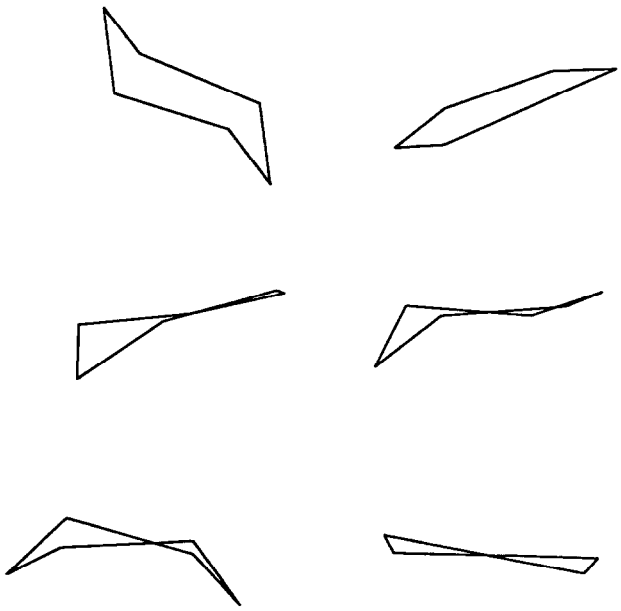

Figure 2. Ring conformations found for six-membered rings.

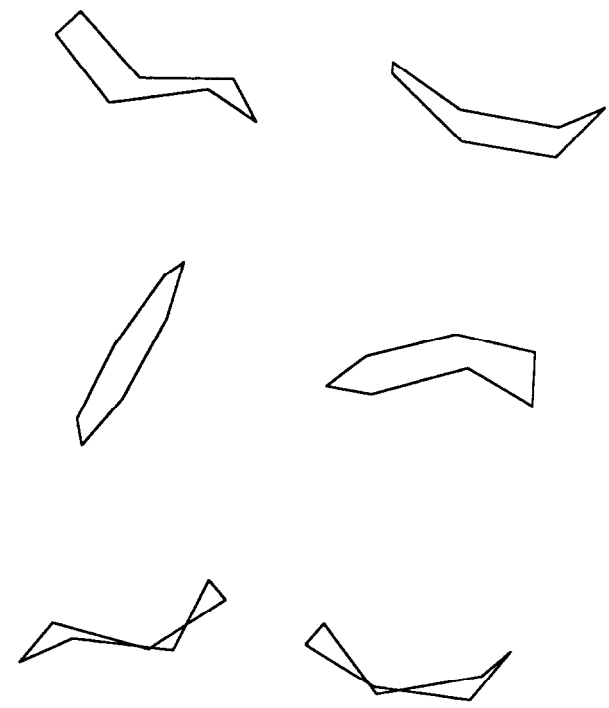

Figure 3. Ring conformations found for seven-membered rings.

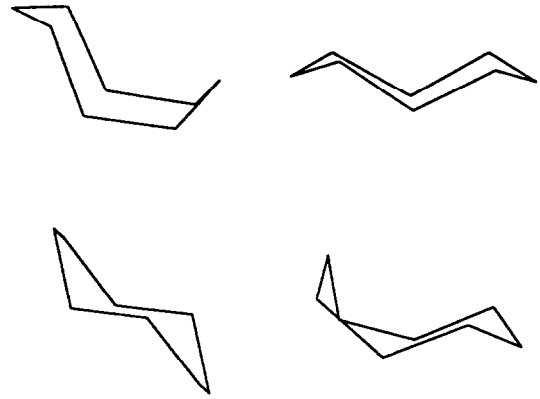

Figure 4. Ring conformations found for eight-membered rings.

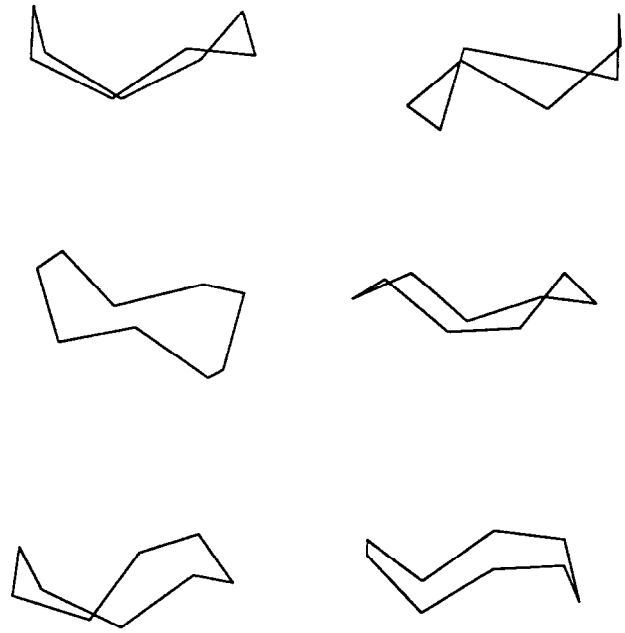

Figure 5. Ring conformations found for nine-membered rings.

terms representing the atom types at each position on the ring template. The ring in Figure 8 would be represented by the clause

ring(class1,O3,C2,C2,C3,C3,C2,C2)

where

C2 = sp2 carbon
C3 = sp3 carbon
O3 = sp3 oxygen

The "class1" designation indicates that the ring is a member of the first class discovered by the CLASSIFY module. On completion of the classification process, these class designations can be mapped to terms such as "chair," which may be more familiar to chemists. Atom numbering

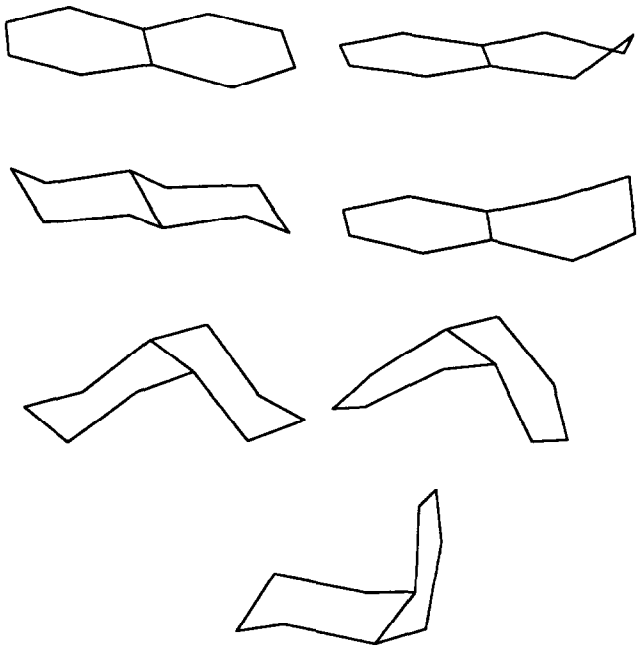

Figure 6. Ring conformations found for decalins.

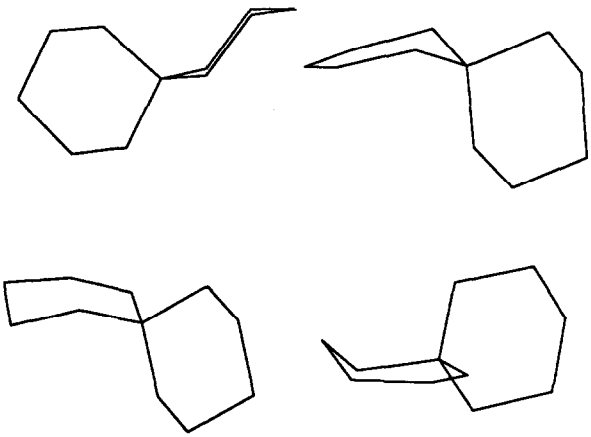

Figure 7. Ring conformations found for spiro-[5.5]undecanes.

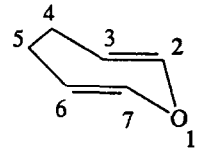

Figure 8. Atom positions used to produce the symbolic representation.

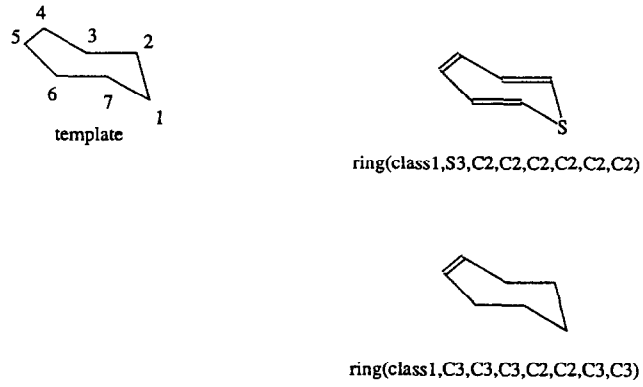

Figure 9. Ring-to-template mapping.

is determined by the mapping of the ring to the class template. Figure 9 shows clauses for three rings that map to the same template.

Rule learning

The objective of the LEARN module is to produce a rule or set of rules that discriminate between examples in one class and examples in other classes. MOUSE-III is capable of carrying out two distinct types of learning—learning what is sufficient for class membership from positive examples and learning what is necessary for class membership from negative examples. In learning from positive examples, MOUSE-III generates a set of rules that describe all observed examples that belong to a particular class. The positive rules must not infer that an example belongs to one class

Let **E** be the set of positive and negative examples of class **C**
repeat
construct a list **L** of attribute/value pairs which describe at least one positive example and no negative examples of class **C**
combine the list **L** and class **C** to form a rule **L** $\rightarrow \in$ **C**
remove from **E** all examples described by **L**
until **E** contains no positive examples of class **C**
Examine the rule set and generalize where possible

Figure 10. The learning algorithm.

when in fact it belongs to another class. In learning from negative examples, MOUSE-III generates a set of rules that describes examples that cannot belong to the class. The removal of these negative examples implicitly describes examples that may belong to the class. The virtues and drawbacks of learning from negative and positive examples will be discussed later.

The algorithm

The basic operation of the learning algorithm is summarized in Figure 10. MOUSE-III attempts to generate a rule that consists of an antecedent containing a set of attribute/value predicates that will be true for positive examples of a given class and will not be true for any negative examples. After such a rule is generated, all examples described by the rule are removed from the data set. Another rule is then generated and the process is repeated until no training examples remain. In generating rules, MOUSE-III attempts to follow the principle of Occam's razor, which states, "given two explanations of the data, all things being equal, the simpler explanation is preferable."[12] The program begins by examining antecedents with a single term (unary rules). If there is not a unary rule that describes at least one positive example and no negative examples, the best unary rules are expanded to all possible binary rules. The process of examining and expanding rules is continued until a rule is found that describes only positive examples. MOUSE-III continues to generate rules until no examples remain in the training set.

After MOUSE-III has generated a complete set of rules, it attempts to generalize the rule set by utilizing a process very similar to the learning algorithm employed by MOUSE-I. A more complete description of the process can be found in reference 8. The generalization routine looks for rules that differ by only one term. On finding such rules, the program creates a new rule that is the disjunction of the two original rules. For instance, the two rules

$$(a = 1) \wedge (b = 1) \wedge (c = 1)$$

and

$$(a = 1) \wedge (b = 1) \wedge (c = 2)$$

can be combined to form the rule

$$(a = 1) \wedge (b = 1) \wedge ((c = 1) \vee (c = 2))$$

MOUSE-III can also employ chemical knowledge to perform a hierarchical generalization on the resulting rules. If MOUSE-III arrives at counting a disjunctive term such as (C3 $\vee$ N3 $\vee$ O3 $\vee$ S3), the program can infer that any sp3 atom will be adequate. The disjunction can then be replaced with the general term *sp3*, which indicates that the position may be occupied by any sp3 atom.

Learning cannot be performed by simple, brute force expansion of every term in every rule. The process of generating rules is highly combinatoric. If we consider an extremely simple system with three attributes labeled A, B and C, and three values labeled 1, 2 and 3, we could generate the rules shown in Figure 11.

In this case, only 63 rules need to be considered. As the number of attributes and values increases, so does the number of possible rules. The number of possible rules can be derived as follows. The number of combinations of r objects

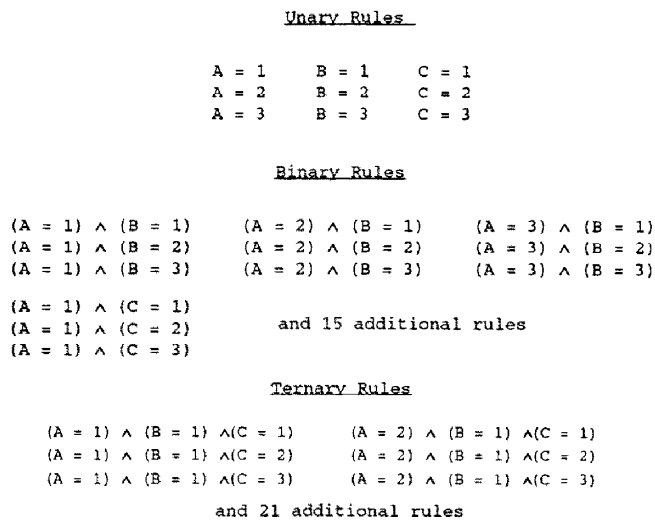

Figure 11. Antecedent clauses for rules.

(items in a rule) selected from a set of n objects (number of ring atoms) is given by[13]

$$\binom{n}{r} = \frac{n(n-1)(n-2)\ldots(n-r+1)}{r!}$$

If each attribute r can assume one of v possible values, the number of combinations is given by v^r, where v is the number of possible atom types. In order to calculate the total number of possible rules, we must consider the unary, binary, ternary ... n-ary rules. This relation is given as

$$\sum_{r=1}^{n} \left(\binom{n}{r} * v^r \right)$$

For a decalin with eight possible atom types, there would be 1.6×10^{11} possible rules. Assuming that each rule could be evaluated in 1 millisecond, the evaluation would require more than five years to complete.

Thus, it is impractical to attempt to evaluate or examine all possible rules. In such cases, a heuristic search is often used[14] to perform a partial search, while maintaining the best chance of success. In order to speed our search through the space of possible rules, we will employ the following heuristic scoring function:[15]

Score = # of positive examples covered–# of negative examples covered

Thus, the rule generation process is essentially increased refinement of the "best rule so far" and is diagrammed in Figure 12. The program begins by examining all possible unary rules. The best unary rule either implies set membership for a subset of the class without error or it incorrectly classifies some of the rings. In the former cases, the rule is saved and the process is begun again. In the latter case, the method assumes that additional terms will improve the predictivity of the best rule so far. The best unary rule is rule b, which was assigned a score of 0. Assume that none of the rules are correct yet. Then the best unary rule b is selected for expansion. The binary rules produced by expanding rule b are designated as ba–be. The binary rules are then evaluated and the process is continued until a rule is generated

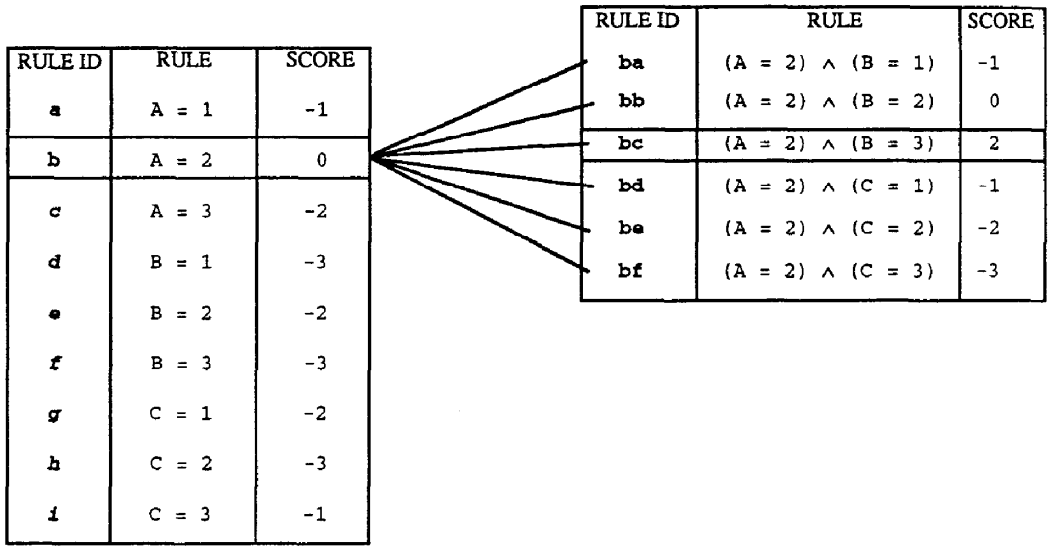

Figure 12. Expansion of unary rule b to all possible binary rules.

that describes only positive examples. The examples described by this rule are then removed from the data set and the process is repeated until no training examples remain.

The "best-so-far" algorithm assumes that the best binary rule will be obtained by refining the best unary rule. It is possible that a less predictive n-ary rule would become the best $(n + 1)$-ary rule if this refinement process were applied to that rule. This sort of inability to guarantee finding the best of all possible sets of rules is an inherent characteristic of any heuristic search through a space that cannot be completely evaluated.

The learning algorithm may be better understood by observing a simple example in which MOUSE-III derives a set of rules for determining whether a ring substituent will prefer the axial or equatorial position on a six-membered ring.[16] The simplicity of this example was chosen for purely pedagogic reasons. An explanation of a larger data set would consume more than the allowable amount of space. The training examples are shown in Figure 13. The four examples with the substituent in an axial orientation will be considered as positive examples while the six examples with the substituent in an equatorial orientation will be considered as negative examples. The examples are shown as attribute/value relationships in Table 1.

MOUSE-III begins by scoring all possible unary rules. The possible unary rules and associated scores are shown in Table 2. The combination in rule e (Pos 5 = OH) achieves the highest score, but still covers negative examples. Thus, it is necessary to expand the rule by examining combinations of rule e with other attribute/value pairs. The binary rules produced by expanding rule e are designated as ea–ed in Table 3.

An examination of the binary rules indicates that two rules (ea and ec) now describe positive examples without describing negative examples. The two rules are retained and the examples (1, 2 and 3) described by the rules are removed from the data set, leaving example 4 as the only positive example. By repeating the process of examining combinations of attribute/value pairs, the program finds that

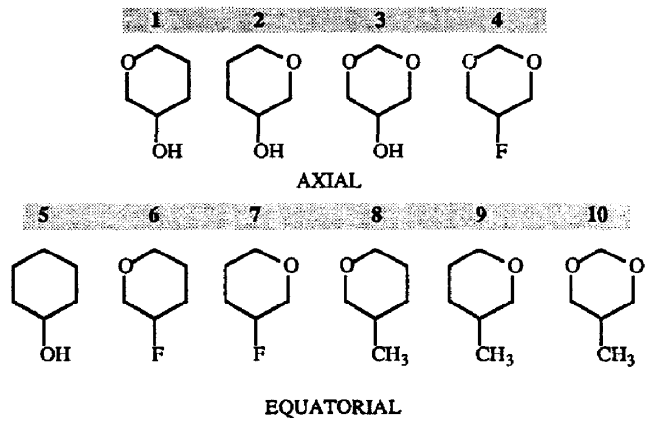

Figure 13. Examples used in the learning demonstration.

Table 1. Attribute/value relationships for the demonstration example.

	Orientation	Position 1	Position 3	Position 5
1	axial	O	CH2	OH
2	axial	CH2	O	OH
3	axial	O	O	OH
4	axial	O	O	F
5	equatorial	CH2	CH2	OH
6	equatorial	O	CH2	F
7	equatorial	CH2	O	F
8	equatorial	O	CH2	CH3
9	equatorial	CH2	O	CH3
10	equatorial	O	O	CH3

Table 2. Possible unary rules.

		Error	Score
a	Pos 1 = CH2	Yes	−2
b	Pos 1 = O	Yes	0
c	Pos 3 = CH2	Yes	−2
d	Pos 3 = O	Yes	0
e	Pos 5 = OH	Yes	2
f	Pos 5 = F	Yes	−1
g	Pos 5 = CH3	Yes	−3

Table 3. Binary rules produced by expanding unary rule e.

		Error	Score
ea	Pos 5 = OH $\wedge$ Pos 1 = O	No	1
eb	Pos 5 = OH $\wedge$ Pos 1 = CH2	Yes	0
ec	Pos 5 = OH $\wedge$ Pos 3 = O	No	1
ed	Pos 5 = OH $\wedge$ Pos 3 = CH2	Yes	0

Table 4. Predictivity of rules learned from positive examples.

System	Training examples	Test examples	Classes	# Correct	# Wrong	Could not classify
six	412	412	6	398 (97%)	14	0
seven	87	87	7	75 (86%)	12	1
eight	44	43	5	33 (75%)	2	8
decalin	343	343	10	339 (99%)	4	0

no single attribute/value pair or combination of two attribute/value pairs covers the positive example without covering a negative example. Thus, the complete example becomes a rule. The four examples are now covered by three rules:

(1) Axial $\leftarrow$ ((Pos 1 = O) $\wedge$ (Pos 5 = OH))
(2) Axial $\leftarrow$ ((Pos 3 = O) $\wedge$ (Pos 5 = OH))
(3) Axial $\leftarrow$ (Pos 1 = O) $\wedge$ (Pos 3 = O) $\wedge$ (Pos 5 = F)

Rules 1 and 2 differ by only one term, thus the two rules can be combined with a disjunction to form the following rule:

Axial $\leftarrow$ ((Pos 1 = O) $\wedge$ (Pos 1 = O) $\vee$ (Pos 5 = OH))

In this small example, two rules represent ten examples for a compression of 80%. As the number of examples increases, the savings can become quite significant. When MOUSE-III was run on large data sets, the rules occupied less than 2% of the space occupied by the examples.

Testing

In the TEST module, the rules developed in the LEARN module are applied to the training examples. The predicted ring conformation is compared to the actual ring conformation. If the predicted and actual conformations are the same, the example is classified as correct. If MOUSE-III is unable to classify the ring based on the current set of rules, this is noted. If the predicted and actual conformations differ, the test example is classified as incorrect. Currently, the test module is used only to determine the quality of the generated rules. Future versions of the program will examine misclassified examples and will attempt to repair inaccurate rules.

RESULTS

MOUSE-III was run on six, separate data sets:

(1) a set of 174 seven-membered rings
(2) a set of 87 eight-membered rings
(3) a set of 686 decalins
(4) a set of 852 six-membered rings
(5) a set of 66 spiro[5.5]undecanes
(6) a set of 31 nine-membered rings

Predictivity of the generated rules

The quality of the rules learned by MOUSE-III can be assessed by examining the accuracy of the predictions for the test examples. The predictivity was measured for the six-, seven- and eight-membered rings as well as the decalins. The nine-membered rings and spiro compounds were not evaluated due to the small number of available examples and the large number of classes in the data set. The lack of quality data is one of the prime motivations for making the next version of MOUSE an explorer program. The explorer program will be capable of generating its own examples, using molecular or quantum mechanics, in order to fill in gaps in the knowledge base.

Table 4 shows the results of testing the rules generated from positive examples. The data indicate that as the number of training examples increases, so does the predictivity of the generated rules. Table 5 shows the results of testing based on rules generated from negative examples. The data indicate that learning from negative examples produces rules that are superior to those learned from positive examples. This phenomenon can be attributed to the fact that there are usually more negative than positive examples of a class. If we have a data set that consists of N examples, which are evenly divided among four classes, then we have N/4 positive examples of each class and 3N/4 negative examples of each class. It can also be seen that as the number of training examples increases, a convergence between what is necessary (the rules learned based on positive evidence) and what is sufficient (the rules learned based on negative evidence) is achieved.

Table 5. Predictivity of rules learned from negative examples.

System	Training examples	Test examples	Classes	# Correct	# Wrong	Could not classify
six	412	412	6	400 (97%)	12	0
seven	87	87	7	85 (98%)	2	0
eight	44	43	5	41 (95%)	2	0
decalin	343	343	10	339 (99%)	4	0

Table 6. Space occupied by rules as compared to examples.

System	Examples	Positive rules	Negative rules
nine	31	7%	15%
spiro	66	4%	6%
eight	87	2%	3%
seven	174	3%	7%
decalin	686	1%	1%
six	824	1%	1%

Compression of the instance space

Table 6 provides a comparison of the sizes of the rules generated by MOUSE-III and the equivalent Cartesian coordinate representations. An examination of the data reveals that the rules occupy only 1–15% of the space occupied by the Cartesian coordinates. The data also indicate that the rule generation process tends to be much more efficient for data sets with large numbers of examples. The quality of rule learning can be judged by the reduction in the amount of information needed to make correct decisions. A set of rules as large as the original raw data is actually not anything more than a restatement of the observed data in detailed terms. On the other hand, accurate classification by a very small set of rules indicates a high degree of learning has occurred. In addition, a small, accurate rule base provides a greater degree of utility to both chemists and computer programs. A large rule base is difficult for any chemist or program to examine. By reducing the size of the data set, we drastically decrease the amount of time required to access the necessary data.

Chemical knowledge learned by MOUSE-III

One of the primary goals in designing MOUSE-III was to build a program that can learn rules that are usable by both chemists and computer programs. This section examines one of the more interesting rules learned by MOUSE-III. A number of results obtained by the program were considered surprising by organic chemists. One of the first surprising results came about during the study of six-membered rings. We had previously expected planar conformations to be important only for aromatic systems. However, MOUSE-III generated the following rule for the planar conformation:

$$\text{class1} \leftarrow ((a = sp2) \wedge (b = sp2) \wedge (d = sp2) \wedge (e = sp2)$$

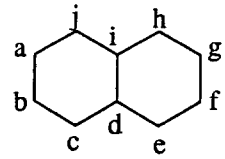

Figure 14. Template mapping for the planar decalin ring system.

According to this rule, a six-membered ring can exist in a planar conformation if it has four sp2 atoms at positions 1, 2, 4 and 5 on the ring. Molecular mechanics calculations carried out with MM2 verified that a minimum does indeed exist for 1,3-cyclohexadiene in this conformation. Future versions of MOUSE will utilize molecular mechanics and semi-empirical techniques as means of verifying the generated rules.

An interesting corollary to this rule arose in the analysis of decalin systems where MOUSE-III produced the following rule for the planar conformation:

$$\text{class2} \leftarrow ((a = sp2) \wedge (b = sp2) \wedge (d = sp2) \wedge (f = sp2) \wedge (g = sp2) \wedge (i = sp2)$$

The positions specified in the rule are equivalent to the positions specified in Figure 14.

Once again, molecular mechanics calculations verified the presence of an energy minimum. It is interesting to note that the rule for decalins is simply a logical "and" of two examples of the previous rule. This suggests a possible future course of exploration by examining the combinations of rules regarding different ring systems to obtain rules for new ring systems.

Manipulation of the rules

The logical combination of rules pertaining to the same class in a ring system has proven to be efficacious in improving performance. For example, the combination of rules based on positive evidence and rules based on negative evidence can be much more powerful than either type of rule by itself. An example of this was observed with the planar conformation of the seven-membered ring. MOUSE-III generated the following rules for this ring system:

$$\text{class1} \leftarrow ((a = sp2) \wedge (b = sp2) \wedge (c = sp2) \wedge (d = sp2) \wedge (e = sp2) \wedge (f = sp2) \wedge (g = sp2))$$

$$\text{class1} \leftarrow ((a = S2) \wedge (b = N2) \wedge (c = N2) \wedge (d = S2))$$

The first rule states that a seven-membered ring will have a flat conformation if it is made up entirely of sp2 atoms. The second rule states that a seven-membered ring will display a flat conformation if it contains two adjacent pairs of sp2 sulfur and nitrogen atoms. MOUSE-III generated this rule because this pattern of atoms was observed in four examples of the planar ring class and no examples of these substitution patterns were found in other classes. A chemist will quickly recognize that this condition is not sufficient to produce a flat ring. The presence of sp3 atoms would distort the ring into another conformation. However an examination of the rules based on negative evidence indicates the presence of the following rule:

$$\neg\text{class1} \leftarrow (a = sp3)$$

This rule indicates that in order for a seven-membered ring to have a flat conformation, it must not contain any sp3 atoms. Positive assertions are combined using disjunctions, while positive and negative rules are combined using conjunctions of the positive terms of the positive rule and negations of the negative rule. By combining this rule learned from negative examples with the previous rule learned from positive examples, we arrive at the following rule:

$$\text{class1} \leftarrow ((a = S2) \wedge (b = N2) \wedge (c = N2) \wedge (d = S2)) \wedge \neg(a = sp3)$$

This combination of what is sufficient (provided by the rules based on positive examples) and what is necessary (provided by the rules based on negative examples) can be extremely powerful.

CONCLUSIONS

MOUSE-III shows that it is possible to write a program that can learn rules of conformational behavior from unprocessed crystallographic examples. The subsequent rules are represented symbolically and are useful to chemists. The rules generated by MOUSE-III can be up to 99% accurate. However, the accuracy of the generated rules is directly dependent on the number of available training examples. With only 44 examples in the training set, the eight-membered rings were the worst examined, with an overall accuracy of only 75% for positive inference. Although the results for learning based on positive examples ranged from mediocre to excellent, the results for learning from negative examples were consistently above 95%. Interestingly enough, this agrees with the premise used by WIZARD, called "don't be stupid." It is often easier and more accurate to avoid making a mistake than to do the right thing directly.

This relationship between the number of examples and the quality of the rules suggests that with fewer than 100 examples, the positive rules will not be that useful. However, it is often important to be able to infer conformations in ring systems when experimental evidence is sparse. This indicates the need for an explorer program that can use MM2, MOPAC or other numerical methods to augment insufficient chemical data.

Learning rules from data has at least a twofold payoff. The rules can be used by chemists as well as conformational analysis programs. Programs such as AIMB[17] (which search crystal databases for analogous compounds) have been quite successful at predicting conformations. One drawback to these techniques is the amount of CPU time required to perform substructure searches of the entire database. In addition, without the benefit of concept generation and negative rule formation, seemingly small changes in the substructure might make substantial differences in the allowable conformations. In concept learning, the rate-determining step can be done once "off-line" (before the conformational analysis is attempted) and the perceived time during the analysis can be vastly reduced.

A problem that always haunts chemists attempting to learn from data is the question of experimental error. A valuable side effect of the rule generation process is that most experimental errors will show up as rules that pertain to only one example of one class. Currently, nothing is done with such singletons, but the explorer program will be able to take note of these cases and examine them more thoroughly.

REFERENCES

1 Linas, J.R. and Ruiz, J.M. In *Computer Aids to Chemistry* (G. Vernin and M. Chanon, Eds.) Ellis Horwood, West Sussex, England, 1986, 200–256
2 Hinton, G.E. In *Machine Learning—Paradigms and Methods* (J. Carbonell, Ed.) MIT Press, Cambridge, Massachusetts, 1990, 185–234
3 Weiss, S.M. and Kulikowski, C.A. *Computer Systems That Learn;* Morgan Kaufmann: San Mateo, California, 1991
4 Michaelski, R.S. *Artif. Intelligence* 1983, **20,** 111–116
5 Mitchell, T.M. *Artif. Intelligence* 1982, **18,** 203–226
6 Quinlan, J.R. *Machine Learning* 1986, **1,** 81–106
7 Dolata, D.P. and Walters, W.P. *J. Mol. Graphics* 1993, **11,** 106–111
8 Dolata, D.P. and Walters, W.P. *J. Mol. Graphics* 1993, **11,** 112–117
9 Shah, A.V., Walters, W.P., Shah, R. and Dolata, D.P. In *Computerized Chemical Data Standards: Databases, Data Interchange, and Information Systems* (R. Lysakowski and C.E. Gragg, Eds.) American Society for Testing and Materials, Philadelphia, Pennsylvania, 1994
10 Meng, E.C. and Lewis, R.A. *J. Comp. Chem.* 1991, **7,** 891–898
11 Leach, A.R., Prout, C.K. and Dolata, D.P. *J. Comp. Chem.* 1990, **11,** 680–692
12 Blumer, A., Ehrenfeucht, A., Haussler, D. and Warmuth, M.K. *Information Processing Letters* 1987, **24,** 377–380
13 Freund, J.E. In *Statistics,* Prentice-Hall, Englewood Cliffs, New Jersey, 1970, 62–69
14 Nilsson, N. *Problem Solving Methods in Artificial Intelligence;* McGraw-Hill: New York, 1971
15 Bratko, I. *PROLOG Programming for Artificial Intelligence;* Addison Wesley: Menlo Park, California, 1990
16 Riddell, F.G. In *The Conformational Analysis of Heterocyclic Compounds,* Academic Press, San Francisco, 1980, 74–75
17 Wipke, W.T. and Hahn, M.A. In *Applications of Artificial Intelligence in Chemistry* (T. Pierce and B. Hohne, Eds.) ACS Symposium Series 306, American Chemical Society, 1986, 136–146