Genetic Programming for the Induction of Decision Trees to Model Ecotoxicity Data

Frances V. Buontempo,[†] Xue Zhong Wang,*[,†] Mulaisho Mwense,[†] Nigel Horan,[‡] Anita Young,[§] and Daniel Osborn[||]

Department of Chemical Engineering and School of Civil Engineering, University of Leeds, Leeds LS2 9JT, U.K., AstraZeneca UK Ltd., Brixham Environmental Laboratory, Freshwater Quarry, Brixham, Devon TQ5 8BA, U.K., and Centre of Ecology and Hydrology, Monks Wood, Huntingdon PE28 2LS, U.K.

Received November 18, 2004

Automatic induction of decision trees and production rules from data to develop structure−activity models for toxicity prediction has recently received much attention, and the majority of methodologies reported in the literature are based upon recursive partitioning employing greedy searches to choose the best splitting attribute and value at each node. These approaches can be successful; however, the greedy search will necessarily miss regions of the search space. Recent literature has demonstrated the applicability of genetic programming to decision tree induction to overcome this problem. This paper presents a variant of this novel approach, using fewer mutation options and a simpler fitness function, demonstrating its utility in inducing decision trees for ecotoxicity data, via a case study of two data sets giving improved accuracy and generalization ability over a popular decision tree inducer.

1. INTRODUCTION

With the ever-increasingly stringent environmental regulation, according to the specialized organic chemicals sector association (SOCSA[1]) in the U.K., environmental management has in recent years become the dominant cost factor for many specialty chemicals manufactured in the U.K., representing around 16% of the total turnover.[2] Among the various parameters that characterize the hazardous environmental impact of chemicals, toxicity is now considered as one of the most important. It was found that although control on biological oxygen demand (BOD), chemical oxygen demand (COD), total organic carbon (TOC), and suspended solids and physical parameters such as temperature, pH, and color have brought a significant improvement in water quality, some aquatic ecosystems have not improved as much as anticipated, presumably because of the presence of toxic substances which may have little contribution to BOD and COD.[3]

As a result the U.K. Environment Agency (EA) has introduced toxicity as a new measure for limiting effluent discharge. The EA approved assays for testing the toxicity of aqueous effluents are young trout, *Daphnia* (water flea), and green algae. Existing methods of toxicity testing used by the SOCSA companies involve subcontracting work to environmental testing laboratories. The tests often take several days to a week and cost on average £2−3K for a normal test and £10K for a full test. More importantly, such tests cannot be carried out before the waste effluents are actually produced. There are two methods that are being explored in order to minimize if not eventually replace bioassays using animals, i.e., rapid toxicity testing using alternatives to animals and computer based prediction using structure−activity relationships (SAR) and quantitative structure−activity relationships (QSAR).

SARs and QSARs are regarded as a very promising technique. QSARs try to map physical, chemical, structural, and biological parameters characterizing a compound to the observed toxicity values, while SARs map characteristics of compounds to the presence of biological activity. QSARs are derived through fitting empirical functions to a set of model data, and their output may then be employed as part of other computer based techniques. Common computational techniques employing SAR and QSAR for toxicity prediction can be broadly classified into three categories: rule-based expert systems, statistical methods, and neural networks. Typical rule-based expert systems are DEREK,[4] Oncologic,[5] and HazardExpert.[6] This approach relies on a collection of rules, which relate a particular chemical structure or substructure to its toxicity. The rules are often collected from domain experts. DEREK makes qualitative rather than quantitative predictions. Statistical linear regression methods represent a quantitative relationship between a numerical measure of toxicity and a set of structure descriptors. Representative commercial systems are TOPKAT and Tsar 3D of Accelrys.[7] Neural networks have also been used in toxicity prediction.[8−15] Despite the fact that the EA has not approved the use of computational toxicity prediction methods for environmental control and the necessity of avoiding inappropriate use of these methods,[16] these tools will improve and will certainly have an increasingly important role to play. Table 1 summarizes the advantages and disadvantages of these traditional approaches and compares them with inductive learning, discussed in more detail below. This overview illustrates the strengths and weaknesses of each methodology, showing the types of results they can produce: qualitative versus quantitative, linear versus non-

* Corresponding author phone: +44 113 343 2427; fax: +44 113 343 2405; e-mail: x.z.wang@leeds.ac.uk.
[†] Department of Chemical Engineering, University of Leeds.
[‡] School of Civil Engineering, University of Leeds.
[§] AstraZeneca UK Ltd.
[||] Centre of Ecology and Hydrology.

10.1021/ci049652n CCC: $30.25
Published on Web 05/12/2005

Table 1. Comparison of Expert Systems, Statistical Methods, Neural Networks, and Inductive Data Mining Methods

	expert systems (ESs)	statistical methods (SMs)	neural networks (NNs)	inductive data mining
advantages	Human expert knowledge can be used. The knowledge is transparent and causal.	Compared with ESs, SMs are data driven methods and therefore are more objective. Give quantitative predictions.	Compared with ESs, NNs are data driven methods and therefore are more objective. Give quantitative predictions. The models are nonlinear. Models are easy to set up and be trained.	It combines the advantages of ESs, SMs, and NNs. It can give both qualitative and quantitative predictions. Both databases and human knowledge can be used together effectively. The knowledge is transparent and causal.
disadvantages	The knowledge is subjective. Experimental data cannot be effectively used. The output is often qualitative.	Compared with ESs, the model is largely a black-box. Human knowledge cannot be used effectively.	Compared with ESs, the model is largely a black-box. Human knowledge cannot be used effectively.	Inductive data mining methods are still at research stage in computer science, particularly in handling attributes which take numerical rather than symbolic values.

linear, and so on. It needs to be pointed out, however, that the comparison is only between the techniques; there is no direct link to a comparison of the tools developed using each method. For evaluations of the performances of commercial tools readers are referred to existing literature.[17−19]

Inductive learning can produce clear, transparent, and understandable decision trees and rules from training data that apply generally. When properly trained there is the potential of matching the accuracy of more opaque methods such as feedforward neural networks.[20] In addition, by comparison with many statistical techniques, this approach is nonparametric, so it makes no assumptions about the underlying distribution of the data.[21] Due to these advantages, inductive data mining or inductive learning has been an active research area in machine learning and knowledge discovery. A number of approaches have been proposed, such as AQ11,[22,23] VersionSpaces,[24,25] CART,[26] Decision Forests,[27] SCAM,[28] and C5.0.[29] However, the majority of these methods have been developed for problem domains where attributes only take discrete or symbolic values. They have proved to perform remarkably well with discrete valued attributes. However, in our previous studies[30] on applying the techniques to problems where attributes take real numbers, it was found that the performance decreases dramatically in terms of accuracy. This has also been confirmed in studies by many other researchers.[31] Despite this, inductive learning or inductive data mining is still a very promising technique for toxicity prediction. Nonetheless, only a few examples of decision tree induction for toxicity are reported in the literature.[8,21,32−36] The packages CASE (Computer automated structure evaluation)[32] and its successor MultiCASE[37,38] use statistical methods to distinguish between fragments present in user data containing between 2 and 10 atoms which are more common to either active or inactive compounds. The former are described as biophores, while the latter are described as biophobes. MultiCASE then develops local QSARs for training set compounds containing given biophores. This approach thereby generates rules for toxicity prediction based on fragment descriptors induced from training data.

Each of the other cited approaches using inductive learning for toxicity prediction classifies the training data into one of two classes: carcinogen versus noncarcinogen and irritant versus nonirritant based on the results of several assays. In contrast, ecotoxicity data usually provides one continuous endpoint, such as an EC50—the concentration effecting 50% of a population, for one species. To apply a classifier, this endpoint data must first be categorized or discretized. Continuous data can be partitioned into ranges in a variety of ways, each having a potential impact on the accuracy and complexity of the models produced.[31,39] Despite the variety of techniques available for discretization, the majority of these are supervised, partitioning the inputs in terms of an already categorical output. Forming ranges of equal width and equal frequency are the only reported unsupervised techniques[31] which are therefore applicable to a continuous endpoint. In addition, the models produced by traditional decision tree generators can be complex,[40] potentially overfitting the training data, thereby not generalizing well. These approaches usually employ a greedy search and so miss potential models, which may be more concise and provide greater generalization ability. Though Lee et al.'s approaches[35,36] employ a beam search providing a partial look-ahead, this will still not cover the whole search space and choosing the beam width can be difficult. Indeed, one step level-look ahead can worsen the quality of trees.[20,41]

Building optimal decision trees is intractable for all but very small data sets with few attributes,[20] yet there are alternatives to the search heuristics usually employed. DeLisle and Dixon[41] developed a new approach, which they call evolutionary programming of trees: EPTree. This employs a genetic algorithm (GA) style search directly upon generations of decision trees, allowing a more thorough investigation of the search space than traditional recursive partitioning. Genetic algorithms have been used in decision tree generation, to decide the splitting points and attributes to be used while growing a tree.[42,43] In contrast, EPTree uses evolutionary programming, sometimes called genetic programming[44] to explore combinations of splitting nodes forming the decision trees themselves. This approach has all the benefits of genetic algorithms, including simultaneous investigation of many solutions and the avoidance of local minima,[45] but does not require parameter encoding into fixed length vectors called chromosomes. Furthermore, the direct action on the decision trees themselves avoids some of the problems usually associated with decision tree induction from genetic programming, such as the tendency to grow overly large trees, known as bloat.[41,46]

We have adopted a variant of EPTree, which is detailed in this paper along with a case study of its application to two ecotoxicity data sets, both with a large number of descriptors and continuous endpoints. This is compared with the performance of Quinlan's See5 (C5.0 for Windows[29]). We show an improved accuracy over See5 for the training data sets and greatly improved generalization ability, measured through the accuracy for hold out test sets. Finally, y-scrambling[47,48] is employed to test the robustness of the results. The endpoint classes are randomly shuffled or permuted, and models rebuild with the same parameters. The accuracy would be expected to be considerably lower for unseen data when y-scrambling has occurred. If the accuracy is still quite good for the shuffled data, either a chance correlation has occurred or the data cannot be modeled well by the methodology. These were run three times to avoid chance correlations. The training and test accuracies for this y-scrambled data were found to be low in each case, indicating the method is robust.

2. THE METHOD

Genetic algorithms find optimal and near optimal solutions to given problems but require encoding of the problem into a fixed sized vector called a chromosome. This requires a genotype (point in search space) to phenotype (point in solution space) map[49] which is employed to assess the fitness of individuals in the population. The genetic programming paradigm was developed from the genetic algorithm approach initially to generate computer programs to solve problems.[50] Later they were used to automatically design electric circuits[51] and for the discovery of suitable architectures for neural networks.[52] In contrast to genetic algorithms, where a problem must be encoded as a fixed length vector, genetic programming can provide solutions of varying size.[53] This is clearly an advantage in the context of decision tree induction, since trees of varying depth can be grown. Traditional genetic programming for decision tree induction requires careful, specialist problem structuring[54,55] and frequently suffers from bloat;[41,46] the trees become very big and therefore lose comprehensibility. DeLisle and Dixon's EPTree[41] provides the first example of the application of genetic programming directly to decision trees, overcoming these problems.

As with DeLisle and Dixon,[41] we generated univariate binary trees, i.e., trees using one attribute for splitting at each node, and each node having precisely two children or being a leaf node. As depicted in Figure 1, the first step generates the required number of trees by randomly choosing a row (here, a compound) and column (descriptor) from the points covered at the node under construction. The left child will take those data points with selected attribute values less than or equal to the value of the chosen row and column, while the right child will take those having a greater value. A check is performed to see if each child node will cover enough data points; the required number was set to 10% of the total training rows. If a child will not cover enough rows, another combination is tried until an acceptable value is found, or until several combinations have been tested. Any child node that becomes a leaf node (i.e. pure: all the rows covered are in the same class, or near pure: less than the required number of compounds in all but one class) is labeled as such,

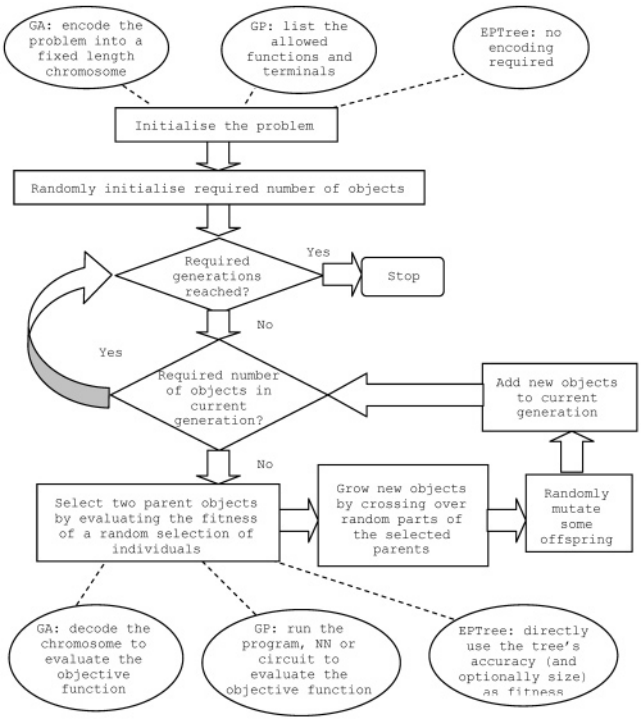

Figure 1. The evolutionary process for optimization, contrasting genetic algorithms (GA), genetic programming (GP) and its variant EPTree, for the induction of decision trees. Note that the main difference lies in the problem encoding: for GA the objects are fixed length vectors called chromosomes, for GP they are structures such as computer programs, circuit diagrams or neural network (NN) architectures defined through the allowable functions and terminals in a node, and for EPTree they are the decision trees themselves.

while the other nodes grow children until all nodes either have two children or are leaf nodes, whence the tree is fully grown and added to the first generation.

Once the first generation is fully populated new trees are grown by crossover: splitting the selected parents at a random node and recombining the parts to form two new trees and then possibly mutating. To select the pair of parent trees that will take part in crossover, tournament selection is employed. A subset of trees from the parent generation is randomly selected. The best tree from this subset is chosen according to the objective or fitness function employed, here simply the accuracy on the training data. This best tree is used as a parent tree for crossover. A second parent is then selected by the same process. The number of trees taking part in the tournament is configurable, though we found 16 to be appropriate for the data sets considered below, when trees numbering 10 times the compounds in the training set were used in each generation. EPTree used the number of leaf nodes and the accuracy in its fitness function. In contrast, our fitness function only uses the accuracy of the trees in the competition. Since we enforce that a node must contain at least 10% of the rows there can be a maximum of 10 leaf nodes. When we included the number of leaf nodes in the fitness function, the population tended to converge to a relatively small set of trees, decreasing the parts of the search space explored, and thereby leading to a slower overall increase in accuracy, and often seeming to get stuck in regions containing less accurate trees of the same size as those produced without using the leaf node count in the fitness function.

In addition, every *n*/*m*th tree was mutated, with a random choice of change of split value (corresponding to choosing a different row's value for the current attribute), choosing a new attribute while keeping the same row, choosing a new attribute and new row, or regrowing part of the tree from any randomly selected node (apart from leaf nodes). Also, when no improvement in accuracy had been seen for k generations, selected as 5 for the data sets below, the trees generated were mutated. If either crossover or mutation gives rise to a node previously classed as a leaf node which is no longer pure or can now usefully be split further, that part of the tree is regrown. If a node becomes a leaf node during either operation, its previous children will not be copied to the new tree. This process is repeated until the required number of trees has been grown for the new generation. Generations are grown up to the requested number.

The option to employ elitism[45] was added. This carries the best user-specified number of trees through to the next generation. We found that elitism did not improve the accuracy or speed of discovery of good solutions, though without it the accuracy of the best tree in the next generation was sometimes lower than in a previous generation. Any tree matching or increasing the best accuracy so far was therefore dropped out to a file when elitism was not employed.

The changes to the original approach, in particular to the mutation options and fitness function, were adopted because they provide an improved speed of the search to find better solutions for our data sets, which have considerably more descriptors than either case study presented by the original paper.[41] In addition the original case studies have two categories for the data, while ours have four. The next section presents the results of See5 and those of our tree induction approach for these two ecotoxicity data sets.

3. RESULTS

3.1. The Data Sets Used. The first data set used comprises 75 compounds which had been studied by Zhao et al.[56] The compounds are a diverse mixture of organics, including chlorobenzenes, nitrobenzenes, anilines, and phenols. The endpoint describes the concentration lethal to 50% of the population; LC50, transformed to Log(1/LC50), of *vibrio fischeri*—a bioluminescent bacterium, along with the values for the octanol/water partition coefficient, CLOGP, and the McGowan characteristic volume, Vx, given in the original paper. The compounds were then optimized using HyperChem (Hypercube Inc., Waterloo, Canada), and other descriptors, including topological, fragment, and physical property descriptors, were calculated using DRAGON.[57] This gives 1497 including include constitutional, topological, walk counts, Burden eigenvalues (BCUT), Galvez topological charge indices, 2D autocorrelation and charge descriptors, aromaticity indices, Randic molecular profiles, geometrical descriptors, radial distribution function descriptors, 3D Molecule Representation of Structure based on Electron diffraction (MoRSE), weighted holistic invariant molecular (WHIM), GEometry, Topology, and Atom Weights AssemblY (GETAWAY), functional groups, and atom-centered fragments. More details are provided in Table 2. Those with zero variation were removed, leaving 1093 descriptors. The endpoint was then partitioned into 4 equal frequency groups, with the split points shown in Table 3, to provide categories in place of the continuously valued endpoint, since preliminary experiments suggested this enabled See5 to produce better models than other splits. The data were split into 60 training compounds and 15 test compounds, each with approximately equal numbers in each of the four categories, by selecting every fifth compound as a test compound.

The second data set comprises 80 organic compounds taken from Cronin et al.[58] The compounds are a diverse organics, covering a range of hydrophobicity, including aldehydes, ketones, benzenes, anilines, and phenols. The endpoint describes the concentration effecting 50% (EC50) of the population of algae *chlorella vulgaris*, by causing fluorescein diacetate (FDA) to disappear. Again, this is given as the logarithm of its reciprocal. The compounds were optimized and 1150 descriptors with zero variation calculated by DRAGON were combined with the octanol/water partition coefficient (LOGKow), energy of the lowest unoccupied molecular orbital (LUMO), and the first-order difference in valence connectivity index (delta1chiv) given in the original paper. The endpoint was also split into four equal frequency groups, with the split points shown in Table 3, and the data were split into 64 training compounds and 16 test compounds with approximately equal numbers in each group, achieved by sorting the data on increasing endpoint. This division follows the approach taken by Cronin et al.[58]

3.2. See5 Results. First, Quinlan's See5 (C5.0 for Windows, Release 1.19) was applied to both data sets, requesting at least 10% of the training data at a node, that is 6 for the first data set and taken as 7 for the second data set. The decision trees generated by See5 are shown in Figures 2 and 3 and the corresponding confusion matrices and accuracies in Tables 4 and 5. For the *vibrio fischeri* data set the accuracy achieved on the training data is 88.3%, but the corresponding results for the test data are low; just 60.0%. For *chlorella vulgaris* the training accuracy is 90.6% and for the test data drops to 75.0%. The accuracies are summarized in Table 10 for comparison with the results from the subsequent tree induction via genetic programming.

3.3. Genetic Programming Results. For both data sets, 100 generations of trees were produced by evolutionary programming using the same training data and the corresponding results found for the same test data as used for See5. We tried no elitism and elitism with between 2 and 16 trees surviving. No elitism tended to find better solutions and did so more quickly. The results are therefore presented for no elitism. Each generation consisted of 600 trees, approximately 10 times the number of data points. It was found that this tended to provide the best performance. Fewer tended to restrict the possible coverage of the search space and more tended to provide no advantage while slowing the growth of each generation. Sixteen trees competed in each tournament, and 66.7% of the trees were mutated for the first data set, with 50% mutated for the second data set. These values were all adopted after recording the accuracy of the best trees and the average accuracy of each generation on the training data. With these parameters, each generation took about 1 s to grow on a Dell Dimension 8200. For the bacterial data set, the training accuracy of See5 was matched with a greater accuracy for the test data by generation 31. An example of such a tree is shown in Figure 4 and its performance in Table 6. For the algae data set the accuracy

Table 2. Information on the Classes of Descriptors Used for Each Data Set

type of descriptor	definition	
constitutional	physical description of the compound; molecular weight, atoms count	
topological	2D descriptors taken from the molecular graph; Wiener index, Balaban index	
walk counts	walk count and self-returning walk counts for various orders	
Burden eigenvalues (BCUT)	eigenvalues of the adjacency matrix, reflecting the topology of the whole compound; the diagonals by weighted by atomic mass, volume, electronegativity, or polarizability	
Galvez topological charge indices	describes charge transfer between pairs of atoms calculated from the eigenvalues of the adjacency matrix; including topological and mean charge index of various orders	
2D autocorrelation	sum of the atom weights of the terminal atoms of all the paths of a given length (lag); Moreau, Moran, and Geary autocorrelations	
charge descriptors	charges estimated by quantum molecular methods; e.g. total positive charge, dipole index	
aromaticity indices	estimated from geometrical distance between aromatically bonded atoms; e.g. harmonic oscillator model of aromaticity	
Randic molecular profiles	derived from distance distribution moments of the geometry matrix including both both molecular profile and shape profile	
geometrical descriptors	conformational-dependent, based on molecular geometry; e.g. 3D Wiener index, gravitational index, distance degree index	
radial distribution function descriptors	obtained from radial basis functions centered at different distances	each either unweighted or weighted by atomic mass, volume, electronegativity or polarizability
3D Molecule Representation of Structure based on Electron diffraction (MoRSE)	calculated by summing atomic weights viewed by different angular scattering functions	
GEometry, Topology, and Atom Weights AssemblY (GETAWAY)	calculated from the leverage matrix, representing the influence of each atom in determining the shape of the molecule, obtained by centered atomic coordinates	
weighted holistic invariant molecular (WHIM)	statistical indices calculated from the coordinates of atoms projected onto 3 principal components from a weighted covariance matrix	as above or weighted by electrotopological state
functional groups	counts of various atoms and functional groups, e.g. primary carbons, aliphatic ethers	
atom-centered fragments	from 120 atom centered fragments defined by Ghose-Crippen; e.g. Cl attached to primary carbon	
various others	unsaturation index; number of nonsingle bonds Hy; a function of the count of hydrophilic groupsaromaticity ratio; aromatic bonds/total number of bonds in a H-depleted atom Ghose-Crippen molecular refractivity fragment based polar surface area	

Table 3. Equal Frequency Ranges for Each Data Set Used To Define the Classes Used in Decision Tree Induction

data set	minimum	class 1 range	class 2 range	class 3 range	class 4 range	maximum
1. Bacteria	0.90	≤ 3.68	≤ 4.05	≤ 4.50	> 4.50	6.32
2. Algae	-4.06	≤ -1.05	≤ -0.31	≤ 0.81	> 0.81	3.10

of See5 was matched by generation 9 again with greater accuracy for the test data, as shown in Figure 5 and Table 7.

For the bacterial data set, other trees with the same maximum accuracy were produced up to generation 37, whence trees with improved accuracy over See5 for the training data were grown which still retained the improved accuracy for the test data. An example tree is shown in Figure 6 and its performance in Table 8. Though 100 generations were grown, no improvement in accuracy was seen after generation 37. For the algae data set, the accuracy of See5 was improved on by generation 30. This was again accompanied by an improvement in accuracy for the test data over See5, though this accuracy is lower than that achieved by trees from generation 9. An example of such a tree is shown in Figure 7 and its performance in Table 9. No further improvement in training accuracy was seen in subsequent generations.

Finally, y-scrambling was used to test the robustness of the genetic programming results. The endpoint classes were randomly permuted three times for each training and test data set, and then experiments were rerun with the same parameters for each shuffled data set. For the first run, the bacteria data achieved 66.7% for the y-scrambled training data and 33.3% for the scrambled test data. For the algae data, a selection of trees with 65.6% for training was induced with 12.5%, 25.0%, and 31.3% for the unseen test data. For the second run, the bacteria data managed 66.7% for training and only 6.6% for testing. For the algae data, a selection of trees with 68.8% for training was induced, one with 6.3% and the rest with 18.8% for the unseen test data. On the last run, the bacteria training data reached 63.3% corresponding to one tree with 20.0% and the rest with 26.7% for the scrambled test data. For the algae data, the training accuracy reached 62.5%, while the test accuracy 25.0%. In each case, the training accuracy is low and the test accuracy even lower, indicating the method has produced robust nonspurious results for the actual data.

Table 10 summarizes the results for See5 and the trees induced by genetic programming for both data sets. The trees induced by genetic programming have improved on the accuracy of those produced by See5 for both the training

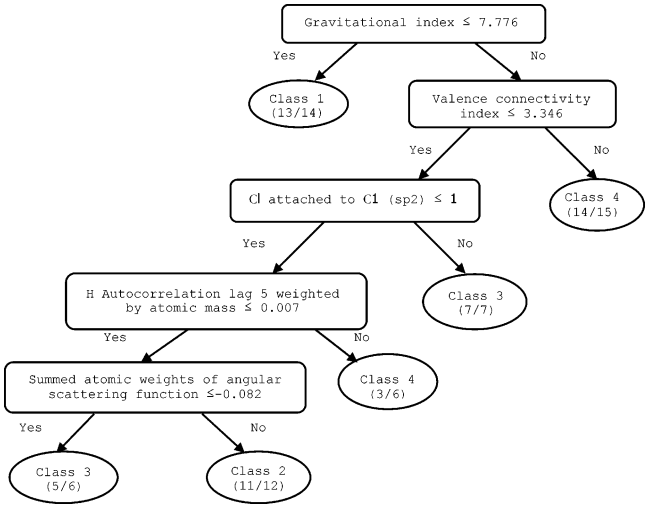

Figure 2. The decision tree produced by See5 (Release 1.19) for the bacteria data. The numbers in brackets at a leaf node indicate the number of training data points correctly classified out of the number of compounds covered at that node. The class ranges are shown in Table 3, and its accuracy and confusion matrix are shown in Table 4.

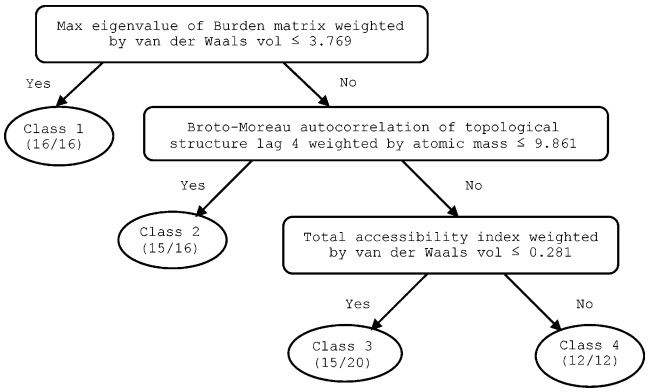

Figure 3. The decision tree produced by See5 for the algae data. Again, the numbers in brackets at a leaf node indicate the number of training data points correctly classified out of the number of compounds covered at that node. The class ranges are shown in Table 3, and its accuracy and confusion matrix are shown in Table 5.

Table 4. Accuracy and Confusion Matrix for the Bacterial Training and Test Data, Corresponding to the Decision Tree Induced by See5 Shown in Figure 2

		predicted class			
training		1	2	3	4
actual class	1	13			
	2	1	11	1	1
	3		1	12	3
	4				17
	correct: 53/60 = 88.3%				

		predicted class			
testing		1	2	3	4
actual class	1	2	1	1	
	2		2	1	1
	3			2	1
	4		1		3
	correct: 9/15 = 60.0%				

and test data, without a significant increase in tree size measured by the number of leaf nodes. This shows that genetic programming is a viable alternative to See5 and can handle a large number of inputs and more than just two classes.

Table 5. Accuracy and Confusion Matrix for the Algae Training and Test Data, Corresponding to the Decision Tree Induced by See5 Shown in Figure 3

		predicted class			
training		1	2	3	4
actual class	1	16			
	2		15	1	
	3		1	15	
	4			4	12
	correct: 58/64=90.6%				

		predicted class			
testing		1	2	3	4
actual class	1	4			
	2		3	1	
	3		2	2	
	4			1	3
	correct: 12/16=75.0%				

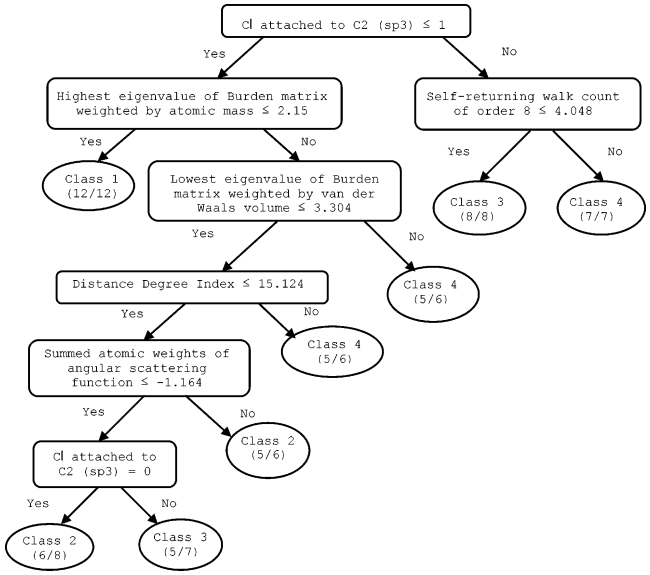

Figure 4. A decision tree produced by generic programming for the first, bacteria, data set in generation 31. It is larger than the tree produced by See5 (Figure 2) but still provides a transparent model. Its accuracy matches that of See5 for the training data while providing an improved accuracy for the test data. Its accuracy and confusion matrix data are shown in Table 6.

4. DISCUSSION

EPTree was shown to improve on the accuracy of decision trees induced by recursive partitioning in S-Plus 6.0 for two data sets. Our case studies show an improved accuracy over See5 for two other data sets. These data sets have different properties to those used in EPTree. First, there is a significant difference in the number of attributes for each pair of data sets. DeLisle and Dixon's[41] data sets consist of 37 and 25 descriptors, respectively. In the second case, the 25 descriptors were selected through simulated annealing. Results were not presented for the whole descriptor set, which would provide an interesting comparison. In contrast, our case study demonstrates that this approach can cope with a large number of correlated descriptors, so that no form of feature extraction is required prior to forming the decision trees. In addition, our data sets have four categories to predict, where those in

Table 6. Accuracy and Confusion Matrix of the Decision Tree Induced in Generation 31 by Genetic Programming Shown in Figure 4 for the Bacterial Data Set[a]

		predicted class			
training		1	2	3	4
actual class	1	12	1		
	2		11	2	1
	3		2	13	1
	4				17
	correct: 53/60 = 88.3%				

		predicted class			
testing		1	2	3	4
actual class	1	3	1		
	2		3	1	
	3		1	2	
	4		1		3
	correct: 11/15 = 73.3%				

[a] It matches the accuracy of the tree induced by See5 (Figure 2 and Table 4) with better generalization ability.

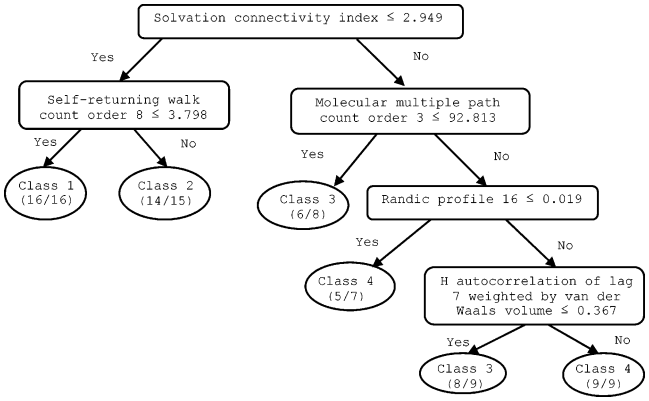

Figure 5. A decision tree produced by generic programming for the second, algae, data set in generation 9. Its accuracy matches that of See5 (see Figure 3) for the training data while providing an improved accuracy for the test data, as shown in Table 7.

Table 7. Accuracy and Confusion Matrix for the Algae Data Set, Corresponding to the Decision Tree Shown in Figure 5 Grown in Generation 9, Each Showing an Improvement over the Tree Produced by See5 (Figure 3 and Table 5)

		predicted class			
training		1	2	3	4
actual class	1	16			
	2		14	1	1
	3		1	14	1
	4			2	14
	correct: 58/64 = 90.6%				

		predicted class			
testing		1	2	3	4
actual class	1	4			
	2		3	1	
	3		1	3	
	4				4
	correct: 14/16 = 87.5%				

the original study have two. Our case study has shown this approach works well for more than two classes.

The rates of mutation used (2/3 and 1/2) are higher than is usually suggested for genetic algorithms.[44] EPTree used an even higher proportion (7/10). There was an increased incidence of repeated decision trees in later generations with

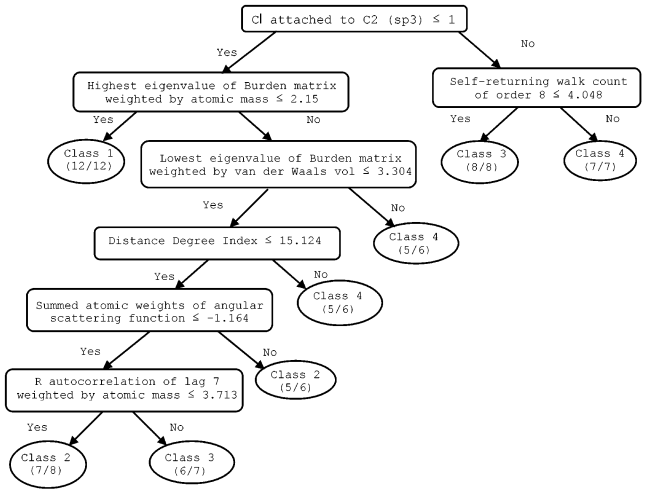

Figure 6. A decision tree produced by generic programming for the first, bacteria, data set in generation 37 showing an improved accuracy over previously grown trees for the training data while maintaining the improvement over See5 for the test data. Further details are shown in Table 8.

Table 8. Accuracy and Confusion Matrix for the Decision Tree Shown in Figure 6 Grown in Generation 37 for the Bacterial Data Set[a]

		predicted class			
training		1	2	3	4
actual class	1	12	1		
	2		12	1	1
	3		1	14	1
	4				17
	correct: 55/60 = 91.7%				

		predicted class			
testing		1	2	3	4
actual class	1	3	1		
	2		3	1	
	3		1	2	
	4		1		3
	correct: 11/15 = 73.3%				

[a] It improves the training accuracy of the tree shown in Figure 4, while maintaining the improvement in accuracy for the test data.

lower mutation rates for our data sets, which is possibly restricting the regions covered in the search space. Increasing the number of mutations performed will help to overcome this, since a purpose of the mutation operator is to avoid getting stuck in local minima.

We have demonstrated an improvement in accuracy over See5 for the training data, together with better generalization ability. The method was further validated by y-scrambling, which could not match the accuracy for the training data and achieved very low accuracies for the test data, indicating robust models were found. In addition, this approach provides a choice of trees with similar accuracy. For example, trees using attributes more easily related to potential modes of action and biological interaction, such as the octanol/water partition coefficient representing adsorption and therefore bioavailability[59,60] can be selected. Furthermore, the results suggest that the greedy search employed in See5 may be overfitting the data since the trees produced by See5 do not give accuracies for the test data comparable with those obtained for the training data.

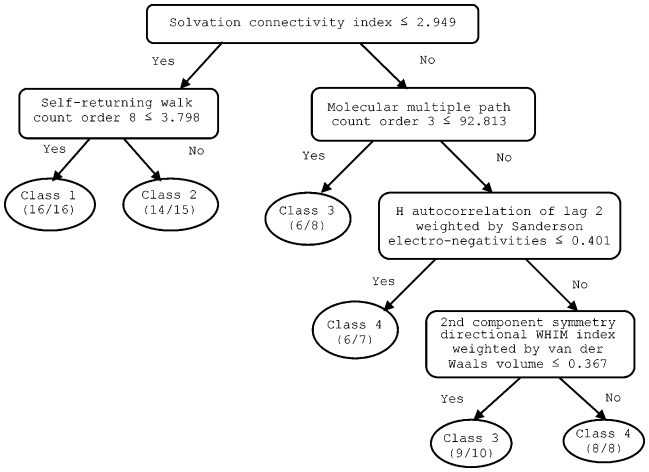

Figure 7. A tree produced by genetic programming in generation 30 for the second, algae, data set, outperforming See5 in terms of the accuracy for the training data and the test data, though the test accuracy is not as high as that achieved in generation 9. Further details are shown in Table 9.

Table 9. Accuracy and Confusion Matrix for the Decision Tree Shown in Figure 7 Grown in Generation 30 for the Algae Data Set, Showing an Improved Training Accuracy and a Decrease in Accuracy for the Test Data from Generation 9, though This Still Improves See5's Accuracy for the Test Data

		predicted class			
training		1	2	3	4
actual class	1	16			
	2		14	1	1
	3		1	15	1
	4			2	14
	correct: 58/64 = 92.2%				

		predicted class			
testing		1	2	3	4
actual class	1	4			
	2		3	1	
	3		1	2	1
	4				4
	correct: 13/16 = 81.3%				

Table 10. Results Obtained by See5 and Decision Trees Induced by Genetic Programming[a]

	details	bacteria data	algae data
See5	tree size	6	4
	training accuracy	88.3%	90.6%
	test accuracy	60.0%	75.0%
tree matching See5's training accuracy induced by genetic programming	generation	31	9
	tree size	8	6
	training accuracy	88.3%	90.6%
	test accuracy	73.3%	87.5%
tree with best accuracy for training data induced by genetic programming	generation	37	30
	tree size	8	6
	training accuracy	91.7%	92.2%
	test accuracy	73.3%	81.3%

[a] Though the latter trees tend to be slightly larger, they are still compact enough to be comprehensible. Furthermore, they show improved accuracy over See5 for both the training and test data.

Future developments should include the use of roulette wheel selection for splitting attributes and values. By weighting the descriptors, this could prefer fragment descriptors to provide even clearer models more closely related to known functional groups or physical properties more immediately related to biological activity. Other future directions should consider a change to the stopping rule, besides node purity and number of points at each node, and ways to prune the trees generated. In addition, ecotoxicity data has endpoints with continuous rather than class values. This paper has presented results for classes determined by an equal frequency split. Work is being carried out to automatically derive appropriate ranges for continuous endpoints.

5. CONCLUSION

The induction of decision trees via generic programming has been adapted from a previously reported study[41] for application to two ecotoxicity data sets of organic compounds, both with a large number of inputs and four classes obtained from equal frequency splitting of the endpoint, in contrast to the previous study using fewer attributes and just two classes. These results were compared with those obtained by the well-known decision tree generator See5 (C5.0 for Windows). Our case studies demonstrate the effectiveness of this approach, which gives decision trees with greater generalization ability than that achieved by See5. In addition, the case studies have shown that decision tree induction via genetic programming can handle a vast number of inputs and more than two classes. Finally, ways to improve this methodology in order to produce further succinct decision trees providing new insights into toxic responses have been suggested.

ACKNOWLEDGMENT

This work was sponsored by EPSRC Crystal Faraday Partnership on Green Technology (grant number 01306000), Brixham Environmental Laboratory of AstraZeneca and the Centre for Ecology and Hydrology. The authors would like to thank Daniel Oliver and Mark Earll of Umetrics UK for contributing their data analysis tool SIMCA-P 8.0 and for providing advice on toxicity modeling using multivariate data analysis methods and QSAR development.

REFERENCES AND NOTES

(1) SOCSA Specialised Organic Chemicals Sector Association, http://www.socsa.org.uk/ 1993.
(2) DEFRA; Environmental Protection Expenditure by Industry: SIC 24: Chemicals and Chemical Products Sector (The Press Release, No. 328/04) ed.; DEFRA: 2002.
(3) SEPA Policy Number 37: Aquatic Discharge Compliance Assessment Directive, http://www.sepa.org.uk/pdf/policies/37.pdf 1999.
(4) LHASA DEREK software, http://www.chem.leeds.ac.uk/luk/.
(5) LogiChem Oncologic software, http://www.logichem.com/.
(6) CompuDrug HazardExpert software, http://www.compudrug.com/.
(7) Accelrys see http://www.accelrys.com/.
(8) Bahler, D.; Stone, B.; Wellington, C.; Bristol, D. Symbolic, neural, and Bayesian machine learning models for predicting carcinogenicity of chemical compounds. *J. Chem. Inf. Comput. Sci.* **2000**, *40*, 906−914.
(9) Basak, S. C.; Grunwald, G. D.; Gute, B. D.; Balasubramanian, K.; Opitz, D. Use of statistical and neural net approaches in predicting toxicity of chemicals. *J. Chem. Inf. Comput. Sci.* **2000**, *40*, 885−890.
(10) Burden, F. R.; Winkler, D. A. Robust QSAR Models Using Bayesian Regularised Artificial Neural Networks. *J. Med. Chem.* **1999**, *42*, 3183−3187.
(11) Gini, G.; Lorenzini, M.; Benfenati, E.; Grasso, P.; Bruschi, M. Predictive carcinogenicity: A model for aromatic compounds, with nitrogen-containing substituents, based on molecular descriptors using an artificial neural network. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 1076−1080.
(12) Kaiser, K. L. E.; Niculescu, S. P. Using probabilistic neural networks to model the toxicity of chemicals to the fathead minnow (Pimephales

promelas): A study based on 865 compounds. *Chemosphere* **1999**, *38*, 3237−3245.
(13) Kovesdi, I.; Dominguez-Rodriguez, M. F.; Orfi, L.; Naray-Szabo, G.; Varro, A.; Papp, J. G.; Matyus, P. Application of neural networks in structure−activity relationships. *Med. Res. Rev.* **1999**, *19*, 249−269.
(14) Niculescu, S. P.; Kaiser, K. L. E.; Schultz, T. W. Modeling the toxicity of chemicals to Tetrahymena pyriformis using molecular fragment descriptors and probabilistic neural networks. *Archives Environ. Contam. Toxicol.* **2000**, *39*, 289−298.
(15) Vracko, M. A study of structure carcinogenic potency relationship with artificial neural networks. The using of descriptors related to geometrical and electronic structures. *J. Chem. Inf. Comput. Sci.* **1997**, *37*, 1037−1043.
(16) ECETOC. QSARs in the assessment of the environmental fate and effects of chemicals, European Centre for Ecotoxicology and Toxicology of Chemicals, 1998.
(17) Benfenati, E.; Gini, G. Computational predictive programs (expert systems) in toxicology. *Toxicology* **1997**, *119*, 213−225.
(18) Benigni, R. The first US National Toxicology Program exercise on the prediction of rodent carcinogenicity: Definitive results. *Mutat. Res.-Rev. Mutat. Res.* **1997**, *387*, 35−45.
(19) Cariello, N. F.; Wilson, J. D.; Britt, B. H.; Wedd, D. J.; Burlinson, B.; Gombar, V. Comparison of the computer programs DEREK and TOPKAT to predict bacterial mutagenicity. *Mutagenesis* **2002**, *17*, 321−329.
(20) Murthy, S. K. Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey. *Data Min. Knowledge Discovery* **1998**, *2*, 345−389.
(21) Worth, A. P.; Cronin, M. T. D. The use of discriminant analysis, logistic regression and classification tree analysis in the development of classification models for human health effects. *J. Mol. Struct.: THEOCHEM* **2003**, *622*, 97−111.
(22) Michalski, R. S. Selection of most representative training examples and incremental generation of VLI hypotheses: the underlying method and description of programs ESEL and AQ11, Computer Science Department, University of Illinois, 1978.
(23) Michalski, R. S.; Chilausky, R. L. Knowledge acquisition by encoding expert rules versus computer induction from examples: a case study involving soybean pathology. *Int. J. Human-Comput. Stud.* **1999**, *51*, 239−263.
(24) Mitchell, T. M. In *Proceedings of IJCAI-87*; Cambridge, Mass, 1977.
(25) Mitchell, T. M. *Machine Learning; International Editions*; McGraw-Hill: New York, London, 1997.
(26) Breiman, L.; Freidman, J. H.; Olshen, R. A.; Stone, C. J. *Classification and regression trees*; Chapman & Hall/CRC: Boca Raton, London, 1993.
(27) Tong, W.; Hong, H.; Fang, H.; Xie, Q.; Perkins, R. Decision forest: combining the predictions of multiple independent decision tree models. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 525−531.
(28) Rusinko, A.; Farmen, M. W.; Lambert, C. G.; Brown, P. L.; Young, S. S. Analysis of a large structure/biological activity data set using recursive partitioning. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 1017−1026.
(29) Quinlan, J. R. *C4.5: programs for machine learning*; Morgan Kaufmann: San Mateo, CA, 1993.
(30) Yuan, B. Ph.D., Leeds University, 2002.
(31) Liu, H.; Hussain, F.; Tan, C. L.; Dash, M. Discretization: An enabling technique. *Data Min. Knowledge Discovery* **2002**, *6*, 393−423.
(32) Klopman, G. Artificial-Intelligence Approach to Structure Activity Studies − Computer Automated Structure Evaluation of Biological-Activity of Organic-Molecules. *J. Am. Chem. Soc.* **1984**, *106*, 7315−7321.
(33) Bahler, D.; Bristol, D. In *Intelligent Systems for Molecular Biology*; Hunter, L., Shavlik, J., Searls, D., Eds.; AAAI/MIT Press: Menlo Park, CA, 1993; pp 29−37.
(34) Glymour, C.; Madigan, D.; Pregibon, D.; Smyth, P. Statistical themes and lessons for data mining. *Data Min. Knowledge Discovery* **1997**, *1*, 11−28.
(35) Lee, Y.; Buchanan, B. G.; Aronis, J. M. Knowledge-based learning in exploratory science: Learning rules to predict rodent carcinogenicity. *Machine Learning* **1998**, *30*, 217−240.
(36) Lee, Y.; Buchanan, B. G.; Mattison, D. M.; Klopman, G.; Rosenkranz, H. S. Learning rules to predict rodent carcinogenicity of nongenotoxic chemicals. *Mutat. Res./Fundam. Mol. Mech. Mutagen.* **1995**, *328*, 127−149.
(37) Klopman, G.; Saiakhov, R.; Rosenkranz, H. S. Multiple computer-automated structure evaluation study of aquatic toxicity II. Fathead minnow. *Environ. Toxicol. Chem.* **2000**, *19*, 441−447.
(38) MultiCASE available at http://www.multicase.com/.
(39) Dougherty, J.; Kohavi, R.; Sahami, M. In *Twelfth International Conference on Machine Learning*; Russell, A. P. a. S., Ed.; Morgan Kaufmann: 1995.
(40) Quinlan, J. R. Simplifying decision trees. *Int. J. Human-Comput. Stud.* **1999**, *51*, 497−510.
(41) DeLisle, R. K.; Dixon, S. L. Induction of Decision Trees via Evolutionary Programming. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 862−870.
(42) Bala, J.; Huang, J.; Vafaie, H.; DeJong, K.; Wechsler, H. In *IJCAI*; Montreal, 1995.
(43) Dutton, D. M.; Conroy, G. V. A review of machine learning. *Knowledge Eng. Rev.* **1997**, *12*, 341−367.
(44) Takagi, H. In *Intelligent Hybrid Systems: Fuzzy Logic, Neural Networks and Genetic Algorithms*; Ruan, D., Ed.; Kluwer Academic Press: Massachusetts, 1997; pp 1−33.
(45) Chiu, T.-L.; So, S.-S. Genetic Neural Networks for Functional Approximation. *Quant. Struct−Act. Relat. Comb. Sci.* **2003**, *22*, 519−526.
(46) Bot, M. C. J. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*; Morgan Kaufmann: 2000; pp 403−410.
(47) Tropsha, A.; Gramatica, P.; Gombar, V. K. The Importance of Being Earnest: Validation is the Absolute Essential for Successful Application and Interpretation of QSPR Models. *QSAR Comb. Sci.* **2003**, *22*, 69−77.
(48) Wold, S.; Eriksson, L. In *Chemometrics Methods in Molecular Design*; Waterbeemd, H. v. d., Ed.; VCH: Weinheim, 1995; pp 309−318.
(49) Hoai, N. X.; McKay, R. I. B.; Essam, D.; Abbass, H. A. In *Genetic Programming: 7th European Conference, EuroGP 2004, Coimbra, Portugal, April 5−7, 2004. Proceedings*; Keijzer, M., O'Reilly, U.-M., Lucas, S. M., et al., Eds.; Springer-Verlag: Heidelberg, 2004; Vol. 3003, pp 67−77.
(50) Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*; The MIT Press: 1992.
(51) GP Genetic Programming organisation, http://www.genetic-programming.org/ 2005.
(52) Pujol, J. C. F.; Poli, R. Evolution of the Topology and the Weights of Neural Networks using Genetic Programming with a Dual Representation, University of Birmingham, School of Computer Science, 1997.
(53) Nikolaev, N.; Slavov, V. In *First European Workshop on Genetic Programming*; Banzhaf, W., Poli, R., Schoenauer, M., Fogarty, T., Eds.; Springer: Europe, 1998; Vol. 1391, pp 49−59.
(54) Loveard, T.; Ciesielski, V. Representing Classification Problems in Genetic Programming. *Proc. Congress Evol. Comput.* **2001**, *2*, 1070−1077.
(55) Marmelstein, R. E.; Lamont, G. B. In *Proceedings of the Third Annual Conference on Genetic Programming*; Morgan Kaufmann: 1998; pp 223−231.
(56) Zhao, Y. H.; Cronin, M. T. D.; Dearden, J. C. Quantitative structure−activity relationships of chemicals acting by nonpolar narcosis − Theoretical considerations. *Quant. Struct.−Act. Relat.* **1998**, *17*, 131−138.
(57) Dragon http://www.disat.unimib.it/chm/Dragon.htm, 2004.
(58) Cronin, M. T. D.; Netzeva, T. I.; Dearden, J. C.; Edwards, R.; Worgan, D. P. Assessment and Modeling of the Toxicity of Organic Chemicals to Chlorella vulgaris: Development of a Novel Database. *Chem. Res. Toxicol.* **2004**, *17*, 545−554.
(59) Cronin, M. T. D.; Schultz, T. W. Pitfalls in QSAR. *J. Mol. Struct.-THEOCHEM* **2003**, *622*, 39−51.
(60) Konemann, H. Quantitative Structure−Activity-Relationships in Fish Toxicity Studies. 1. Relationship for 50 Industrial Pollutants. *Toxicol.* **1981**, *19*, 209−221.

CI049652N