11

Minimizing Structural Risk on Decision Tree Classification

DaeEun Kim

Cognitive Robotics
Max Planck Institute for Human Cognitive & Brain Sciences
Munich, 80799, Germany
daeeun@cbs.mpg.de

Summary. Tree induction algorithms use heuristic information to obtain decision tree classification. However, there has been little research on how many rules are appropriate for a given set of data, that is, how we can find the best structure leading to desirable generalization performance. In this chapter, an evolutionary multi-objective optimization approach with genetic programming will be applied to the data classification problem in order to find the minimum error rate or the best pattern classifier for each size of decision trees. As a result, we can evaluate the classification performance under various structural complexity of decision trees. Following structural risk minimization suggested by Vapnik, we can determine a desirable number of rules with the best generalization performance. The suggested method is compared with C4.5 application for machine learning data.

11.1 Introduction

The recognition of patterns and the discovery of decision rules from data examples is one of the challenging problems in machine learning. When data points with numerical attributes are involved, the continuous-valued attributes should be discretized with threshold values. Decision tree induction algorithms such as C4.5 build decision trees by recursively partitioning the input attribute space [24]. Thus, a conjunctive rule is obtained by following the tree traversal from the root node to each leaf node. Each internal node in the decision tree has a splitting criterion or threshold for continuous-valued attributes to partition a part of the input space, and each leaf represents a class depending on the conditions of its parent nodes.

The creation of decision trees often relies on heuristic information such as information gain measurement. Yet how many nodes are appropriate for a given set of data has been an open question. Mitchell [22] showed the curve of the accuracy rate of decision trees with respect to the number of nodes over the independent test examples. There exists a peak point of the accuracy rate in a certain size of decision trees; a larger size of decision trees can

D.E. Kim: *Minimizing Structural Risk on Decision Tree Classification*, Studies in Computational Intelligence (SCI) **16**, 241–260 (2006)
www.springerlink.com

increase its classification performance on the training samples but reduces the accuracy over the test samples which have not been seen before. This problem is related to the overfitting problem to increase the generalization error[1]. Many techniques such as tree growing with stopping criterion, tree prunning or bagging [25, 21, 24, 5] have been studied to reduce the generalization error. However, the methods are dependent upon a heuristic information or measure to estimate the generalization error, and they do not explore every size of trees.

An evolutionary approach to decision trees has been studied to obtain optimal classification performance [16, 15, 4], since the decision tree based on heuristics is not optimal in structure and performance. Freitas et al. [15] have shown evolutionary multi-objective optimization to obtain both the minimum error rate and minimum size of trees. Their method was based on the information gain measurement; it followed the C4.5 splitting method and selected the attributes with genetic algorithms. They were able to reduce the size of decision trees, but had higher test error rates than C4.5 in some data sets. Recently a genetic programming approach with evolutionary multi-objective optimization (EMO) was applied to decision trees [4, 3]. A new representation of decision trees for genetic programming was introduced [3], where the structure of decision trees is similar to linear regression trees [6]. Two objectives, tree size and accuracy rate in data classification, were considered in the method. The method succeeded in reducing both error rates and size of decision trees in some data sets. However, searching for the best structure of decision trees has not been considered in their works.

It has been shown that EMO is very effective for optimization of multi-objectives or constraints in continuous range [27, 7]. Also EMO is a useful tool even when the best performance for each discrete genotype or structure should be determined [19, 17]. The EMO approach was used to minimize the training error and the tree size for decision tree classification [3, 8]. Also other works using fitness and size or complexity as objectives have been reported [2, 20]. Yet there has been no effort so far to find what is the best structure of decision trees to have the minimal generalization error. Vapnik [26] showed an analytic study to reduce the generalization error, and he suggested the structural risk minimization to find the best structure. It can be achieved by exploring the empirical error (training error) and generalization error (test error) for various structure complexity. We will follow the approach and the tree size will be the parameter to control the structure.

In this work, the EMO with genetic programming for two objectives, the tree size and the training error, is first used to obtain the Pareto-optimal solutions, that is, the minimum training error rate for each size of trees. Then the best tree for each size will be examined to see the generalization performance for a given set of test data. By observing the distribution of the test error rates over the size of trees, we can pinpoint the best structure to minimize the generalization error. In our EMO approach, a special elitism strategy

[1] This is also called test error in this chapter.

for discrete structures is applied. Genetic programming evolves decision trees with variable thresholds and attributes, and an incremental evolution from small to large structures with Pareto ranking is used. The suggested method provides the accuracy rate of classification for each size of trees as well as the best structure of decision trees. The approach will be compared with the tree induction algorithm C4.5. A preliminary study of the approach was published in [18].

11.2 Method

11.2.1 Decision Tree Classification

Decision tree learning is a popular induction algorithm. A decision tree classifies data samples by a set of decision classifiers; the classifiers are located in the internal nodes of the tree and for each instance, the tree traversal depending on the decision of the classifiers from the root node to some leaf node determines the corresponding class. The decision trees can be easily represented as a set of decision rules (if-then rules) to assist the interpretation. Inductive learning methods create such decision trees, often based on heustic information or statistical probability.

One of the most popular learning algorithm is to construct decision trees from the root node to leaf nodes with a top-down greedy method [25, 24]. Each data attribute (with appropriate threshold if the attribute is continuous-valued) is evaluated using the information theory with entropy. This evaluation decides which attribute or what threshold of the selected attribute classifies well a given set of instances. It has been shown that information gain measure efficiently selects one of attribute vectors and its thresholds [23].

Let Y be the set of examples and let C be the set of k classes. Let $p(C_i, Y)$ be the probability of the examples in Y that belong to class C_i. The split probability $p(Y_j)$ in a continuous-valued attribute among m partitions is given as the probability of the examples that belong to the partition Y_j when the range of the attribute is divided into several regions.

Then the information gain of an attribute A over a collection of instances Y is defined as

$$Gain(Y, A) = \frac{E(Y) - \sum_{i=1}^{m} \frac{|Y_i|}{|Y|} E(Y_i)}{\sum_{j=1}^{m} p(Y_j) \log p(Y_j)}$$

where $E(Y) = \sum_{i=1}^{k} p(C_i, Y) \log p(C_i, Y)$ is an entropy function, A has m partitions, and Y_j is one of m partitions for the attribute A.

For a given set of instances, each attribute A has its own threshold τ_i that produces the greatest information gain. This threshold is automatically selected by information gain measurement, and the threshold τ_i is one of the best cut for the attribute A to make good decision boundaries. One decision

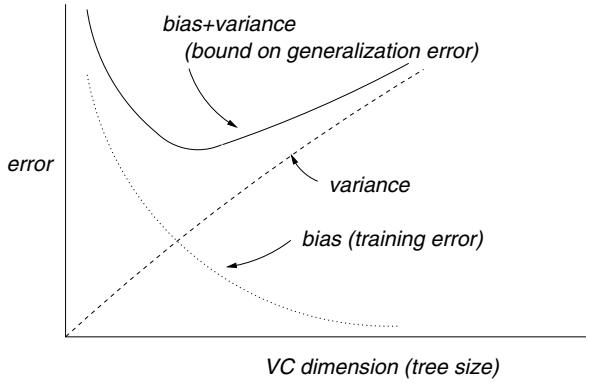


Fig. 11.1. Relationship between training error and generalization error

boundary divides the parameter space into two non-overlapping subsets, depending on τ_i; it has a dichotomy $A > \tau_i$ and $A \leq \tau_i$. However, multiple intervals of the attribute can improve the classification [10].

With the above process of information gain measurement, the decision tree algorithm finds the best attribute and its threshold. More sophisticated algorithms to improve the classification have been developed, but the style of tree induction is not much changed. In our experiments, C4.5 [24] will be used for inductive tree classication.

11.2.2 Structural Risk Minimization

According to the statistical estimation theory by Vapnik [26], while the complexity of a model over a given set of data increases, learning algorithms such as decision trees and neural networks can reduce the approximation error called bias but increase the variance of the model. Much research is concerned with reducing the generalization error which is a combination of bias and variance terms [12, 14, 13]. The generalization error is the rate of errors caused by the model when the model is tested on samples which have not been seen before. Vapnik showed the general bounds for the variance and the generalization error. The generalization error varies with respect to a control parameter of the learning algorithm to model a given set of data; Vapnik [26] mentioned this control parameter as VC (Vapnik-Chervonenkis) dimension. The VC dimension is a measure of the capacity of a set of classification functions.

The number of decision-tree nodes in induction trees can be a control parameter to be related to the generalization error, because increasing the number of nodes can decrease the bias and increases the variance in classification problem. Fig. 11.1 shows the relationship between the training error and generalization error. Here, we are interested in finding a tree structure with minimal generalization error at the expense of increase in training error.

In structural risk minimization, a hierarchical space of structures is enumerated and then the function to minimize the empirical risk (training error, bias) for each structure space is found. Among a collection of those functions, we can choose the best model function to minimize the generalization error. The number of leaf nodes (rules) in decision trees corresponds to the VC-dimension that Vapnik mentioned in the structural risk minimization [26].

The structure of decision trees can be specified by the number of leaf nodes. Thus, we define a set of pattern classifiers as follows:

$$S_k = \{F(x, \beta) | \beta \in D_k\}$$

where x is a set of input-output vectors, $F(x, \beta)$ is a pattern classifier with parameter vector β, D_k is a set of decision trees with k terminal nodes and S_k is a set of pattern classifiers formed by decision trees with k terminal nodes. Then we have

$$S_1 \subset S_2 \subset \cdots \subset S_n.$$

From the method of structural risk minimization [26], we can easily set the VC dimension into the number of leaf nodes and the VC dimension of each pattern classifier is finite. In this chapter, the training error for each set of pattern classifiers, S_k, for $k = 2, ..., n$, is minimized with the EMO method and then the generalization error for the selected pattern classifier is identified. The best pattern classifier or the best structure of pattern classifiers is the one with the minimum generalization error; in the experiments, 10-fold cross validation will be used to estimate the generalization error.

When we wish to have a desirable set of rules over a given set of data, we do not have a prior knowledge about what is the best number of rules to minimize the generalization error. Thus, a two-phase algorithm with the EMO method can be applied to general classification problems. First, we can apply the EMO method to the whole training instances and obtain a set of rules for each size of trees. Then the method of finding the best structure with 10-fold cross validation or other validation process can be applied to the training instances. From this information, we can decide the best VC-dimension, or best size of trees among a collection of rule sets for the original data set. As a result, we can obtain a desirable set of rules to avoid the overfitting problem.

11.2.3 Evolutionary Multiobjective Optimization

We use evolutionary multiobjective optimization to obtain Pareto-optimal solutions which have minimal training error for each size of trees. In the suggested evolutionary approach, a decision tree is encoded as a genotype chromosome; each internal node specifies one attribute for training instances and its threshold. The terminal node defines a class, depending on the conjunctive conditions of its parent nodes through the tree traversal from the root node to the leaf node. Unlike many genetic programming approaches, the current method encodes only a binary tree classification; the only one function set is

a comparison operator for a variable and its threshold, and the terminal set consists of classes determined by decision rules.

The genetic pool in the evolutionary computation handles decision trees as chromosomes. The chromosome size (tree size) is proportional to the number of leaf nodes in a decision tree, that is, the number of rules. Thus, the number of rules will be considered as one objective to be optimized. While an evolutionary algorithm creates a varying size of decision trees, each decision tree will be tested on a given set of data for classification. The classification error rate will be the second objective. The continuous-valued attributes require partitioning into a discrete set of intervals. For simple control of VC dimension, we assume that the decision tree is a binary tree. Thus, the decision tree will have a single threshold for every internal node to partition the continuous-valued range into two intervals. The threshold is one of major components to form a pattern classifier.

We are interested in minimizing two objectives, classification error rate and tree size in a single evolutionary run. In the multi-objective optimization, the rank cannot be linearly ordered. The Pareto scoring in EMO approach has been popular and it is applied to maintain a diverse population over the two objectives. A dominance rank is thus defined in the Pareto distribution. A vector $X = (x_1, x_2, ..., x_m)$ for m objectives is said to *dominate* $Y = (y_1, y_2, ..., y_m)$ (written as $X \prec Y$) if and only if X is partially less than Y, that is,

$$(\forall i \in 1, ..., m, x_i \leq y_i) \wedge (\exists i \in 1, ..., m, x_i < y_i)$$

A *Pareto optimal set* is said to be the set of vectors that are not dominated by any other vector.

$$\{X = (x_1, ..., x_m) | \neg(\exists Y = (y_1, ..., y_m), Y \prec X)\}$$

To obtain a Pareto optimal set, a dominating rank method [11] is applied in this work. Individuals of rank 0 in a Pareto distribution are dominated by no other members and individuals of rank n are dominated only by individuals of rank k for $k < n$. The highest rank is zero, for an element which has no dominator. Fig. 11.2 shows an example of dominating rank method.

In the experiments, tournament selection of group size four is used for Pareto optimization. The tournament selection initially partitions the whole population into multiple groups for the fitness comparison; members in each group are randomly chosen among the population. Inside the tournament group, Pareto score of each member is compared each other and ranked. A higher rank of genomes in the group have more probability of reproducing themselves for the next generation. In our approach, a population is initialized with a random size of tree chromosomes. For each group of four members, the two best chromosomes using a dominating rank method are first selected in a group and then they reproduce themselves; more than one chromosome may have tie rank scores and in this case chromosomes will be randomly selected among multiple non-dominated individuals. A subtree crossover over a copy

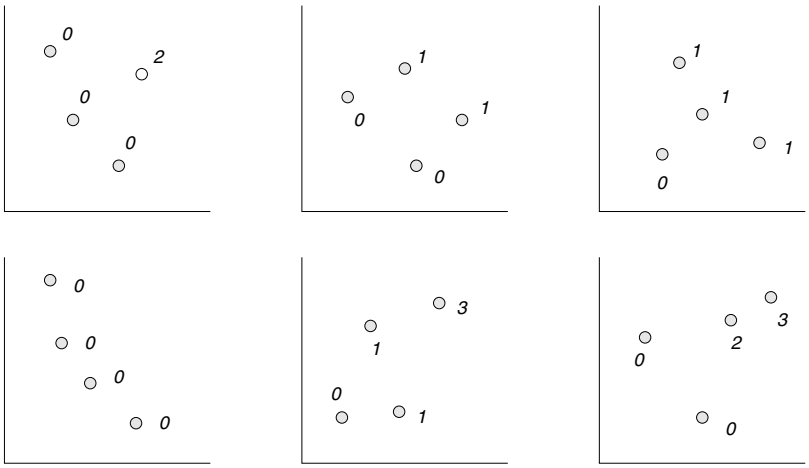


Fig. 11.2. Dominating rank method in a tournament selection of group size four (x-axis and y-axis represent the number of rules and classification error, respectively. each number represents the dominating rank, and a small rank is a better solution since two objectives should be minimized.)

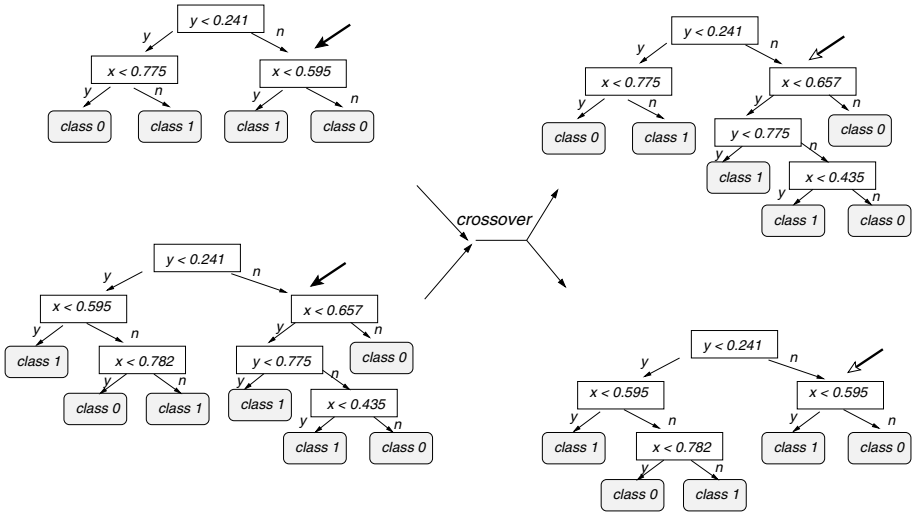


Fig. 11.3. Crossover operator (arrows: crossover point)

of two best chromosomes, followed by a mutation operator, will produce two new offspring. These new offspring replace the two worst chromosomes in the group. The crossover operator swaps subtrees of two parent chromosomes where the crossover point can be specified at an arbitrary branch – see Fig. 11.3.

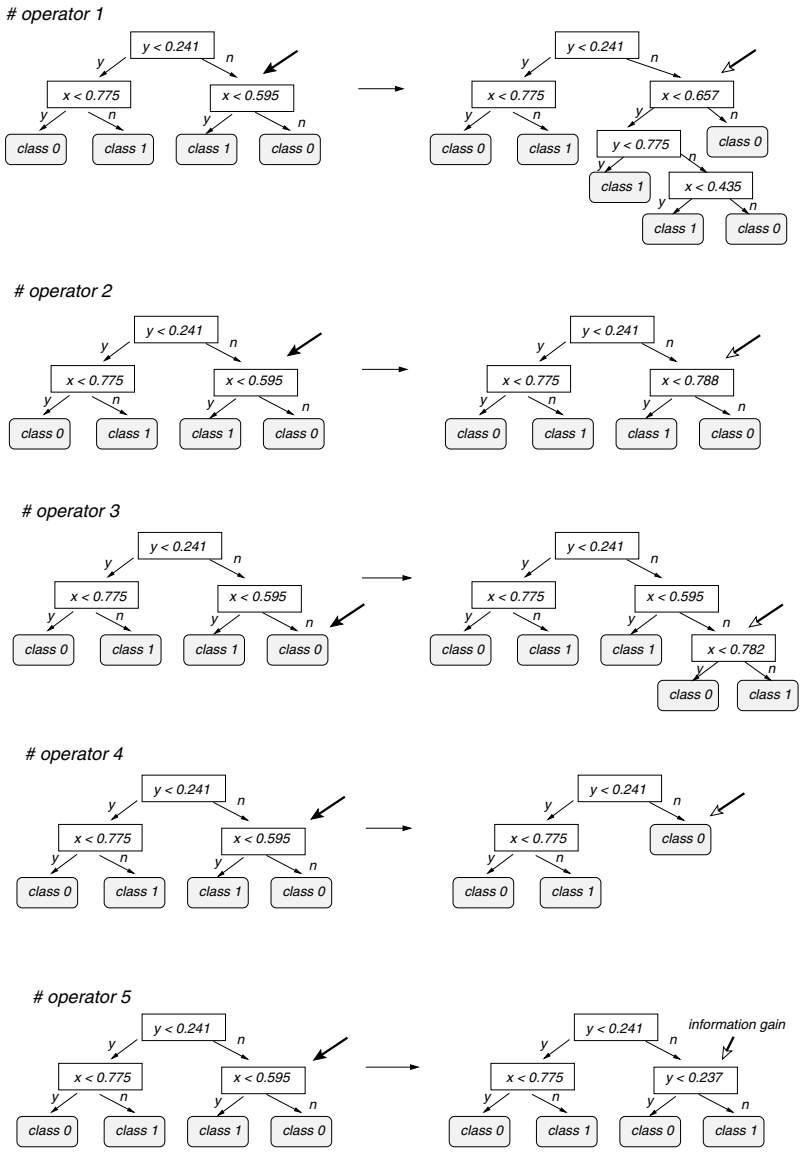


Fig. 11.4. Mutation operators (arrows: mutation point)

The mutation has five different operators as shown in Fig. 11.4. The first operator deletes a subtree and creates a new random subtree. The subtree to be replaced will be randomly chosen in a decision tree. The second operator first picks up a random internal node and then changes the attribute or its threshold. This keeps the parent tree and modifies only one node. The third operator chooses a leaf node and then splits it into two nodes. This will as-

sist incremental evolution by adding one more decision boundary. The fourth operator selects a branch of a subtree and reduces it into a leaf node with random class. It will have the effect of removing redundant subtrees. The fifth operator sets a random attribute in a node and chooses one of the possible candidate thresholds randomly. The candidate thresholds can be obtained at boundary positions[2] by sorting the instances according to the selected variable (the threshold to maximize the information gain is also located at such a boundary position [9]). The last operator has an effect of choosing desirable boundaries based on information gain, but the random selection of the thresholds avoids local optimization only based on information gain. Thus, the last mutation operator[3] accelerates a fast speed of convergence in classification and the other four operators provide a variety of trees in a population. In this work, crossover rate 0.6 and mutation rate 0.2 were used.

In the initialization of the population or the recombination of trees, we have a limit for the tree size. The minimum size of leaf nodes is 2 and the maximum size of leaf nodes is set to 35 or 25; some data set does not need the exploration of as many nodes as 35, because a small number of leaf nodes are sufficient. If the number of leaf nodes in a new tree exceeds the limit, a new random subtree is generated until the limit condition is satisfied. A single run of the EMO method over training examples will lead to the Pareto optimal solutions over classification performance and the number of rules. Each non-dominated solution in a discrete space of tree size represents the minimized error fitness for each number of decision rules. The elitism strategy has been significantly effective for EMO methods [27]. In this work, an elitist pool is maintained, where each member is the best solution for every size of decision trees under progress. For each generation, every member in the elitist pool will be reproduced.

11.2.4 Variable-node C4.5

The well-known tree induction algorithm C4.5 efficiently generates decision trees using information gain. The algorithm can produce a varying size of decision trees by controlling the parameter of a minimum number of object in the branches. When the minimum number of objects in the two branches increases from two to one hundred or more sequentially, we can collect a set of pairs (classification performance, tree size) for each parameter value. That is, we have a collection of pattern classifiers each of which has a different size. Then we extract the best performance as well as the best pattern classifier

[2] We first try to find adjacent samples which generates different classification categories and then the middle point of the adjacent samples by the selected attribute is taken as a candidate threshold.

[3] There is a possibility of using only information gain splitting, but we use this method instead to allow more diverse trees in structure and avoid local optimization.

for each size of trees from the database. We will call this method as variable-node C4.5 to distinguish it from the conventional C4.5 method by default parameter setting. With the variable-node C4.5, the best pattern classifiers are obtained by heuristic measurement, while the EMO tries to find the best classifier for a given size of trees using evolutionary search mechanism.

When the cross validation process is applied, we will estimate the average performance over training set and test set with many trials. It is assumed that the average performance over a given size of trees by variable-node C4.5 will represent the estimation over all the best C4.5 decision trees obtained with a given size of trees. In case that a certain size of trees may be missing by the given splitting method, we calculate the average performance of classification for available tree sizes.

11.3 Experiments

11.3.1 EMO with Artificial Data

The EMO method was first tested on a set of artificial data with some noise as shown in Fig. 11.5; we will show an application of the EMO method to minimize the training error, not generalization error to show how it works, and the experiments minimizing the generalization error will be provided in section 3.2. The data set contains 150 samples with two classes. When the C4.5 tree induction program was applied, it generated 3 rules as shown in Fig. 11.5(a). It produced a 29.3 % training error rate (44 errors). For reference, a neural network (seven nodes in a hidden layer) trained with the back-propagation algorithm achieves a 14.7 % error rate (22 example misclassifications); however, the performance could be improved with better parameter setting. Evolving decision trees with 1000 generations and a population size of 200 by the EMO approach produced Pareto trees. Fig. 11.6 shows an example of the best tree chromosomes. With only two rules allowed, 43 examples were misclassified (28.7 % error rate) as shown in Fig. 11.5(f), and it was better than the C4.5 method. Moreover, six rules was sufficient to obtain the same performance as neural networks with seven hidden nodes. As the number of rules increases, decision boundaries are added and the training error performance improves.

In many cases, the best boundaries evolved for a small number of rules also belong to the classification boundaries for a large number of rules; new boundaries can provide better solutions in some cases. A small number of rules are relatively easily evolved, but a large number of rules needs a long time to find the best trees since more rules have more parameters to be evolved. Large trees tend to be evolved sequentially from the base of small trees or a small number of rules. Thus, incremental evolution from small to large structures of decision trees is operated with the EMO approach. We note that in Fig. 11.5, more rules improve the classification performance for the training data by adding decision boundaries, but some rules support only a small number

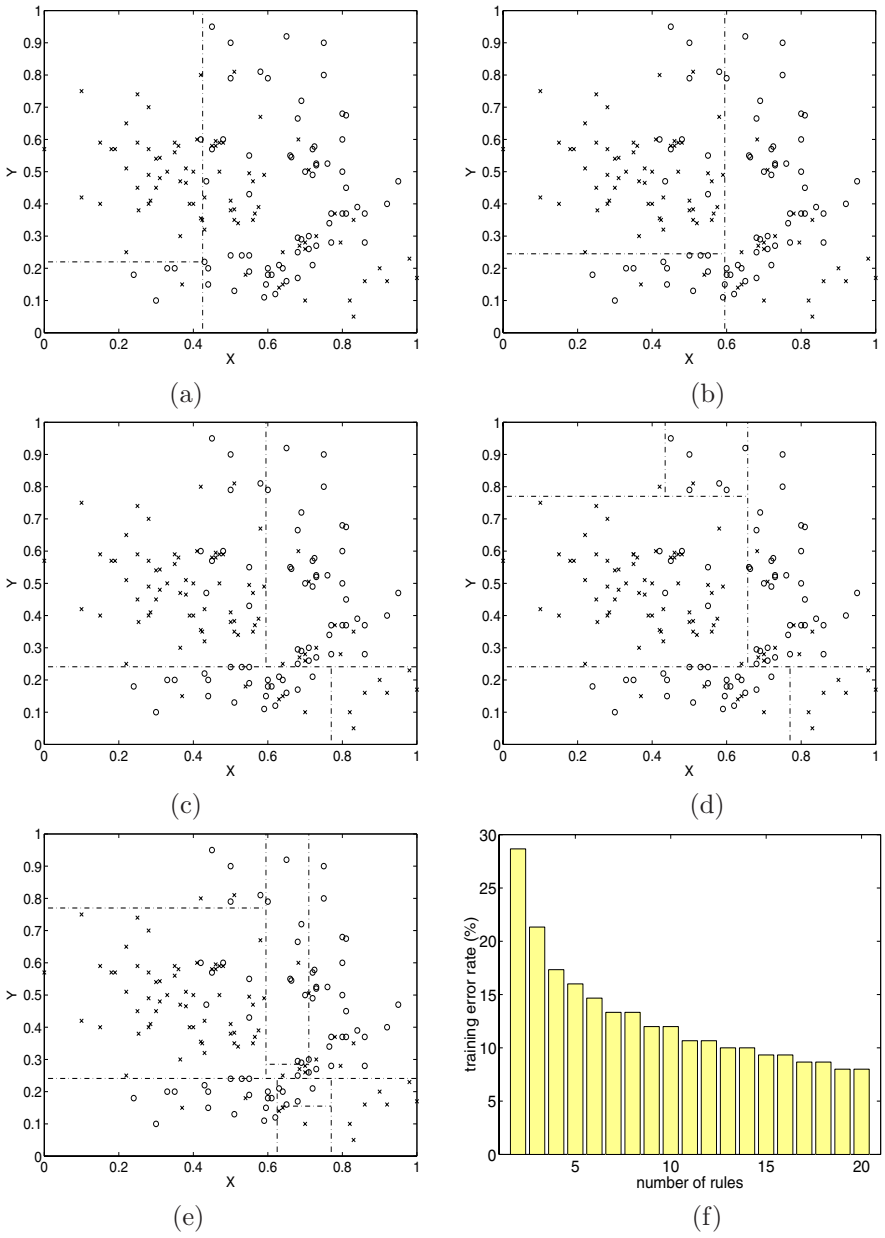


Fig. 11.5. Artificial data and EMO (a) data set and C4.5 decision boundaries (○ : class 0, × : class 1) (b) 3 rules from EMO (c) 4 rules from EMO (d) 6 rules from EMO (e) 9 rules from EMO (f) an example of EMO result

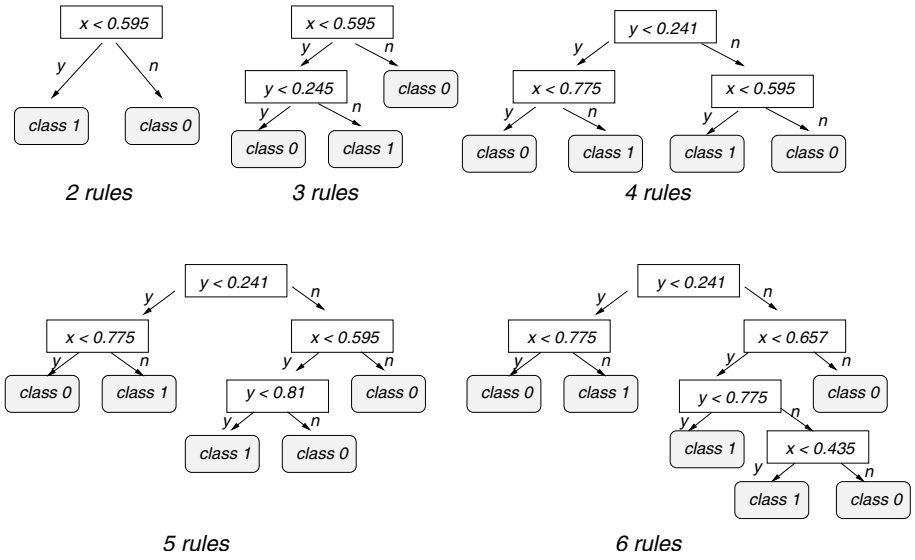


Fig. 11.6. An example of best chromosomes by EMO

of samples. The rules may cause over-specialization problem to degrade the generalization performance. This is the motivation of our work using structural risk minimization.

11.3.2 Machine Learning Data

For the general case, the suggested evolutionary approach has been tested on several sets of data[4] (*iris*, *wine*, *ionosphere*, *ecoli*, *pima*, *wpbc*, *glass*, *bupa*) in the UCI repository [1] and the artificial data in Fig. 11.5. These data sets are mostly for machine learning experiments. Classification error rates are estimated by running the complete 10-fold cross-validation ten times, and we used variable-node C4.5 and the EMO approach as well as C4.5 by default parameter setting. For each size of decision trees, 95% confidence intervals of fitness (test error rate) are measured by assuming t-distribution. For the C4.5 run, both number of rules and error rate will be examined with t statistic. The suggested EMO approach takes a population size of 500 and 1000 generations with tournament selection of group size four for each experiment.

Evolutionary computation was able to attain a hierarchy of structure for classification performance. There exists the best number of rules to show the minimum generalization error as expected from the structural risk minimization. Fig. 11.7(a)-(b) shows that the artificial data have four decision rules as the best structure of decision trees and that the *ionosphere* data have six

[4] Some sets of data include missing attribute values. In that case, the data sample is removed.

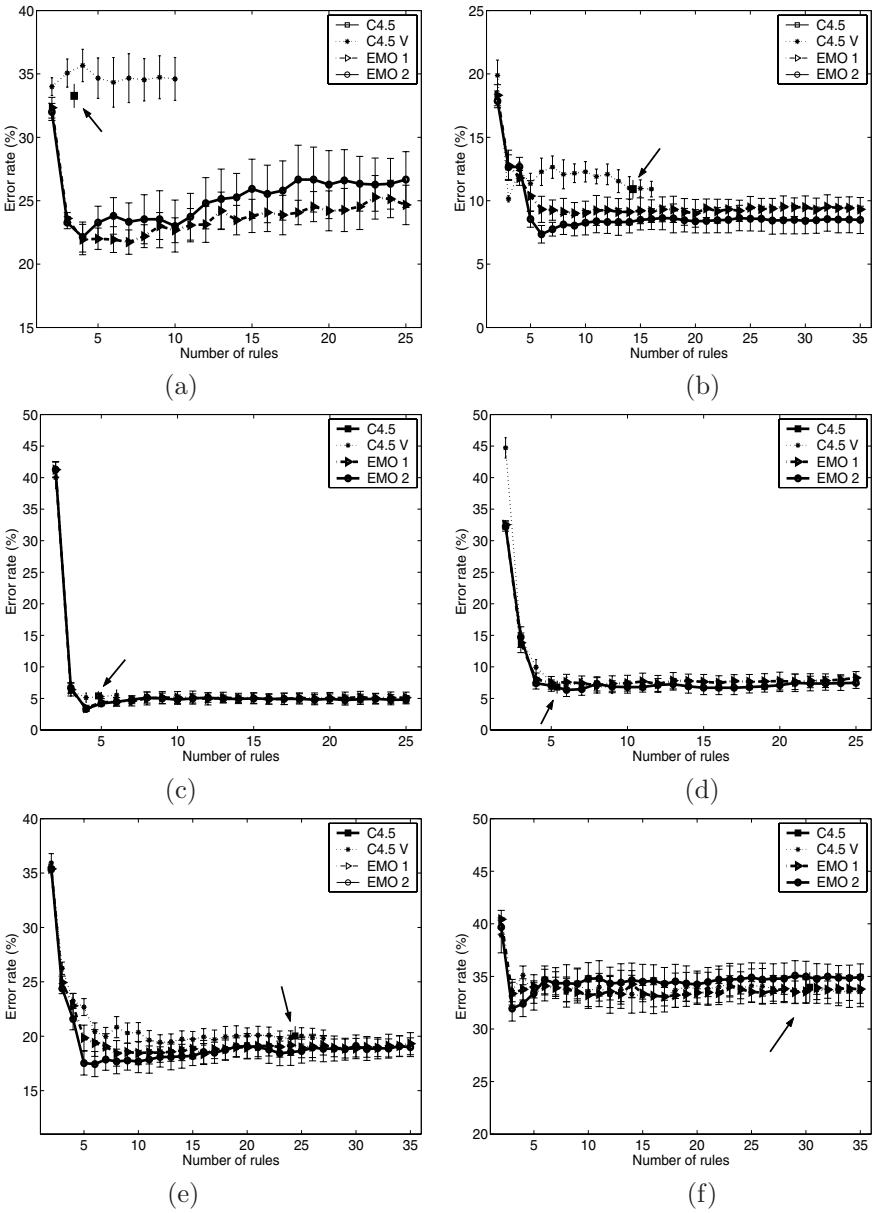
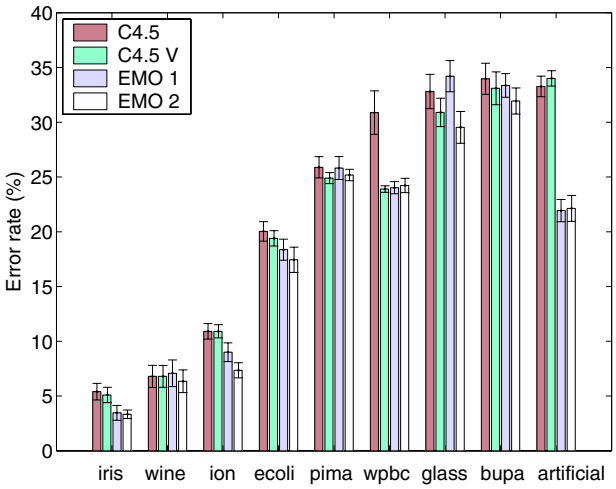
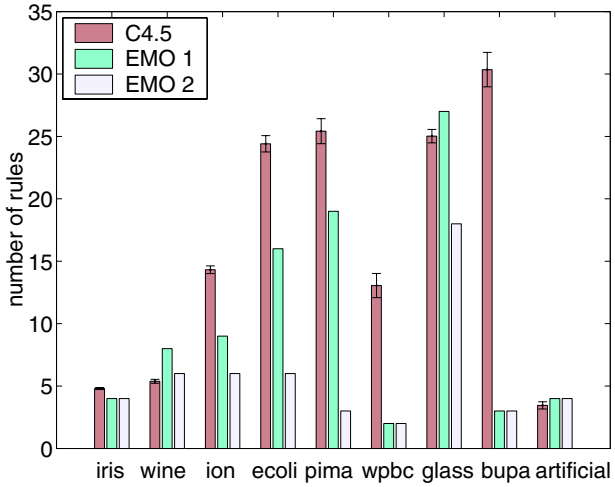


Fig. 11.7. Generalization performance with varying number of rules 1 (arrow: C4.5 with default parameters, *: C4.5 with varying number of nodes, ▷: EMO result with 100 generations, ○: EMO result with 1000 generations) (a) artificial data (b) *ionosphere* data (c) *iris* data (d) *wine* data (e) *ecoli* data (f) *bupa* data



(a)



(b)

Fig. 11.8. Comparison between C4.5 and EMO method (a) error rate in test data with C4.5 and EMO (EMO 1 and EMO 2 represent the EMO running with 100 generations and 1000 generations, respectively) (b) the number of rules with C4.5 and EMO (the number of rules for EMO is determined by selecting the minimum error rate)

rules. If the tree size is larger than the best tree size, then the generalization performance degrades or has no improvement. More generations tend to show a better curve for the best structure of trees. This test validation process can easily determine a desirable number of rules. The EMO even with 100 gener-

Table 11.1. Data classification errors in C4.5 and variable-node C4.5

	C4.5				variable node C4.5	
data	pattern	attr.	error (%)	# rules	error (%)	# rules
artificial	150	2	33.3 ± 0.9	3.5 ± 0.3	34.0 ± 0.7	2
ionosphere	351	34	10.9 ± 0.7	14.3 ± 0.3	10.9 ± 0.6	16
iris	150	4	5.4 ± 0.8	4.8 ± 0.1	5.1 ± 0.7	4
wine	178	13	6.8 ± 1.0	5.4 ± 0.2	6.8 ± 1.0	9
ecoli	336	7	20.0 ± 0.9	24.4 ± 0.7	19.4 ± 0.7	12
pima	768	8	25.9 ± 1.0	25.4 ± 1.0	24.9 ± 0.5	13
wpbc	194	32	30.9 ± 2.0	13.1 ± 1.0	23.9 ± 0.3	2
glass	214	9	32.8 ± 1.6	25.0 ± 0.5	30.9 ± 1.3	9
bupa	345	6	34.0 ± 2.0	30.4 ± 1.9	33.1 ± 1.5	17

Table 11.2. Data classification errors in the EMO method

	EMO (100 gen.)		EMO (1000 gen.)	
data	error (%)	# rules	error (%)	# rules
artificial	21.9 ± 1.1	4	22.1 ± 1.2	4
ionosphere	9.0 ± 0.9	9	7.4 ± 0.7	6
iris	3.5 ± 0.7	4	3.3 ± 0.4	4
wine	7.1 ± 1.2	8	6.3 ± 1.0	6
ecoli	18.3 ± 1.0	16	17.4 ± 1.2	6
pima	25.8 ± 1.0	19	25.2 ± 0.5	3
wpbc	24.0 ± 0.6	2	24.2 ± 0.7	2
glass	34.2 ± 1.4	27	29.5 ± 1.5	18
bupa	33.4 ± 1.1	3	31.9 ± 1.2	3

ations is better than the C4.5 induction tree in the test error rates for these two sets of data, artificial and *ionosphere* data. Variable-node C4.5 does not produce a V-shape curve (in Fig. 11.1) for generalization performance with the VC-dimension, tree size, but instead irregular type of performance curve. Its performance is significantly worse than the EMO performance in most of cases. Fig. 11.7 shows that the variable-node C4.5 is mostly worse in performance than the EMO method for each number of rules, and if we choose the best structure to minimize structural risk, the EMO method outperforms the variable-node C4.5 in generalization performance for all cases except *pima* and *wpbc* data.

We collected the best model complexity and the corresponding performance into Table 11.1-11.2. For reference, we showed the performance of C4.5 by default parameters. Variable-node C4.5 often finds better performance than C4.5 by default parameter setting. Among the collection of varying sizes of trees, there exists some tree better in performance than C4.5 result for most of data sets.

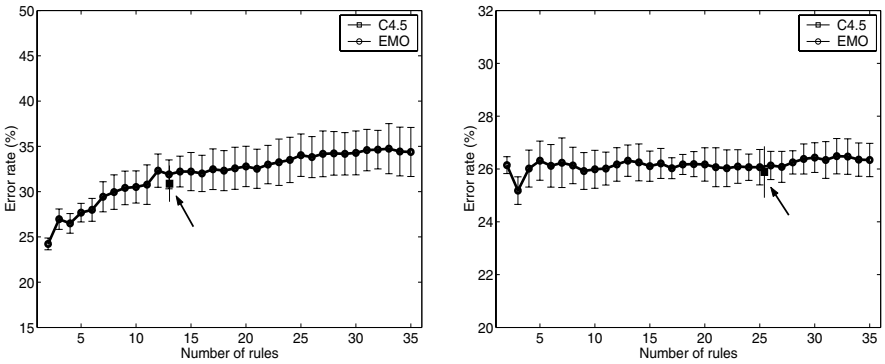


Fig. 11.9. Generalization performance with varying number of rules 2 (arrow: C4.5 with default parameter, ○: EMO result with 1000 generations) (a) *wpbc* data (b) *pima* data

In our experiments, the EMO with 100 generations outperforms C4.5 in the test error rate for all the data except *wine* and *glass*. The EMO with 1000 generations improves both the error rate and the number of rules, and it is better than C4.5 in the test error rate for all the data. Table 11.1-11.2 and Fig. 11.8 show that the EMO is significantly better than C4.5 in error rate with 95 % confidence levels for many experimental data. The EMO method with *wine* and artificial data have a little higher number of rules, but it is due to the fact that the EMO finds an integer number of rules in a discrete space. The other data experiments show that the best number of rules in decision trees by the EMO is significantly smaller than the number of rules by C4.5 induction trees, which is determined by information gain. It confirms that some rules from C4.5 are redundant and thus C4.5 may suffer from an over-specialization problem. In some cases variable-node C4.5 finds a small number of rules, but the rule set has much worse performance in classification than the best rule set by the EMO.

An interesting result is obtained for the *wpbc* and *pima* data (see Fig. 11.9). Two sets of data have a bad prediction performance, regardless of the number of rules. The performance of C4.5 is worse than or similar to that of two or three rules evolved. Investing longer training time on *wpbc* and *pima* data does not improve validation performance. It is presumed that more consistent data are required for the two sets of data.

11.4 Discussion

We evaluated generalization performance under various model complexities, with the pattern classifiers found by the EMO method. An algorithm of min-

imizing structural risk was applied to the Pareto sets of (performance, structure). For the comparison with C4.5, we tried to provide a variety of model complexities using variable-node C4.5, that is, by controlling the parameter of a minimum number of objects in the branches. The performance of C4.5 with varying number of nodes is shown in Fig. 11.7; it is mostly worse than our method in classification performance. With the parameter control, a specific number of decision rules can be missing. Generally it is hard to generate a consecutive number of leaf nodes or a large size of trees with C4.5. It would be a better comparison with the suggested approach if more sophisticated tree induction method with pruning or stopping growing can be tested. As an alternative, other tree induction algorithms that grow trees by one node can be tested and compared with the EMO method. We leave the comparison between the suggested approach and other methods under the same model complexities to future work.

Given any arbitrary data set for future prediction, we can apply the cross-validation process to find the best pattern classifier minimizing structural risk. The data set is divided into training and test set randomly and then the suggested EMO can be applied to the training data. This procedure can be repeated with many trials to obtain the average generalization error. The generalization performance under a variety of structure complexity can determine the best structure or best size of trees. The above method was applied to each training set in the 10-fold cross validation, and the best structure followed the result in Table 11.2 in most of cases. In our approach, the evolved decision trees have axis-parallel decision boundaries. The pattern classification may have a limitation of fitting nonlinear sample patterns, although we can find the best number of classifiers. Applying our EMO approach to linear regression trees or neural network classifiers would produce better classification performance.

If one of the objectives in EMO is discrete, elitism can be easily applied to evolutionary computation by keeping a pool of the best solutions for each discrete genotype. In the EMO approach, all members in the elitist pool were reproduced every generation, where each member corresponds to each size of trees. If chromosomes are linearly ranked by only error rate performance instead of Pareto dominating rank, it fails to produce uniform Pareto-optimal solutions, since the evolutionary run sticks to an one-objective solution. When we tested varying number of members in the elitist pool for reproduction in a new population, we found more members reproduced in the elite pool can significantly improve training performance.

We showed the performance of the EMO with different number of generations, 100 and 1000. In some case, 100 generations are sufficient to find the best model structure. Yet more generations often produce the result that the best model structure shifts to a smaller size of trees. It is believed that better training performance of the EMO can find better model structure to minimize structural risk, but it is still an open question how much training we need for desirable generalization performance.

The computing time of evolutionary computation requires much more time than C4.5. For example, a single EMO run with a population size of 500 and 100 generations over *pima* data takes about 22 seconds while a single run of C4.5 application takes only 0.1 second (Pentium computer). A single EMO run for *iris*, *wine*, *ionosphere*, *ecoli*, *wpbc*, *glass*, and *bupa* took roughly 3 seconds, 6 seconds, 28 seconds, 8 seconds, 12 seconds, 7 seconds, 8 seconds, respectively. Generally the EMO needs much more computing time to find the best performance for every size of trees, but it can improve the classification performance significantly in most of the data sets.

11.5 Conclusions

In this chapter, we introduced an evolutionary multiobjective optimization with two objectives, classification performance and tree size (number of rules), for decision tree classification. The proposed EMO approach searches for the best accuracy rate of classification for each different size of trees. By structural risk minimization, we can find a desirable number of rules for a given error bound. The performance of the best rule set is better than that of C4.5 or variable-node C4.5, although it takes more computing time. In particular, it can reduce the model complexity, the size of trees dramatically. It can also help to evaluate how difficult it is to classify a given set of data examples. Many researchers have used *pima* and *wpbc* in their experiments, but the distribution of error rates over the size of trees implies that these data cannot expect prediction. In addition, we can indirectly determine if a given set of data requires more consistent data or whether it includes many noisy samples.

For future study, the suggested method can be compared with the bagging method [5], which is one of the promising methods to obtain good accuracy rates. The bagging process may also be applied to the best rules obtained from the proposed method. The decision tree evolved in the present work has the simple form of a binary tree. The EMO approach can be extended to more complex trees such as trees with multiple thresholds or linear regression trees. The result can also be compared with that obtained from neural networks.

References

[1] C. Blake, E. Keogh, and C.J. Merz. *UCI* repository of machine learning databases. In *Proc. of the Fifth Int. Conf. on Machine Learning*, 1998. 252

[2] S. Bleuler, M. Brack, L. Thiele, and E. Zitzler. Multiobjective genetic programming: Reducing bloat using SPEA2. In *Congress on Evolutionary Computation*, pages 536–543. IEEE Press, 27-30 May 2001. 242

[3] M.C.J. Bot. Improving induction of linear classification trees with genetic programming. In *Genetic and Evolutionary Computation Conference*, pages 403–410. Morgan Kaufmann, 2000. 242

[4] M.C.J. Bot and W.B. Langdon. Application of genetic programming to induction of linear classification trees. In *Proceedings of the 3rd European Conference on Genetic Programming*, pages 247–258, 2000. 242
[5] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. 242, 258
[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth International Group., 1984. 242
[7] C.A. Coello Coello, D.A. van Veldhuizen, and G.A. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic, 2002. 242
[8] E.D. de Jong and J.B. Pollack. Multi-objective methods for tree size control. *Genetic Programming and Evolvable Machines*, 4(3):211–233, 2003. 242
[9] U.M. Fayyad. *On the induction of decision trees for multiple concept learning*. Ph.D. dissertation, EECS department, University of Michigan, 1991. 249
[10] U.M. Fayyad and K.B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of IJCAI'93*, pages 1022–1027. Morgan Kaufmann, 1993. 244
[11] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proc. of the Fifth Int. Conf. on Genetic Algorithms*, pages 416–423. Morgan Kaufmann, 1993. 246
[12] S. Geman. Neural networks and the bias/variance dilemma. *Neural computation*, pages 1–58, 1992. 244
[13] P. Geurts. *Contributions to decision tree induction: bias/variance tradeoff and time series classification*. Ph.D. thesis, University of Liege, Belgium, 2002. 244
[14] P. Geurts and L. Wehenkel. Investigation and reduction of discretization variance in decision tree induction. In *European Conference on Machine Learning, LNAI 1810*, pages 162–170. Springer Verlag, 2000. 244
[15] A.A. Freitas G.L. Pappa and C.A.A. Kaestner. Attribute selection with a multiobjective genetic algorithm. In *Proc. of the 16th Brazilian Symposium on Artificial Intelligence*, pages 280–290. Springer-Verlag, 2002. 242
[16] K.B. Irani and V.A. Khaminsani. Knowledge based automation of semiconductor manufacturing. In *SRC Project Annual Review Report*, The University of Michigan, Ann Arbor, 1991. 242
[17] D. Kim. Evolving internal memory for T-maze tasks in noisy environments. *Connection Science*, 16(3):183–210, 2004. 242
[18] D. Kim. Structural risk minimization on decision trees using an evolutionary multiobjective optimization. In *Proc. of European Conf. on Genetic Programming, LNCS 3003*, pages 338–348. Springer-Verlag, 2004. 243
[19] D. Kim and J. Hallam. An evolutionary approach to quantify internal states needed for the woods problem. In *From Animals to Animats 7*, pages 312–322. MIT Press, 2002. 242
[20] W. B. Langdon. Data structures and genetic programming. In P.J. Angeline and K.E. Kinnear, editors, *Advances in Genetic Programming 2*, pages 395–414. MIT press, Cambridge, MA, 1996. 242
[21] J. Mingers. An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 4(2):227–243, 1989. 242
[22] T. M. Mitchell. *Machine Learning*. McGraw Hill, 1997. 241
[23] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. 243
[24] J.R. Quinlan. Improved use of continuous attributes in *C4.5*. *Journal of Artificial Intelligence Approach*, 4:77–90, 1996. 241, 242, 243, 244

[25] J.R. Quinlan and R. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80(3):227–248, 1996. 242, 243
[26] V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, 1995. 242, 244, 245
[27] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. Ph.D. thesis, Swiss Federal Institute of Technology, 1999. 242, 249