Turk J Elec Engin, VOL.12, NO.1 2004, © TÜBİTAK

Mining Classification Rules by Using Genetic Algorithms with Non-random Initial Population and Uniform Operator

Korkut Koray GÜNDOĞAN, Bilal ALATAŞ, Ali KARCI
Fırat University, Department of Computer Engineering,
23119, Elazığ-TURKEY
e-mail: {kkoray, balatas, akarci}@firat.edu.tr

Abstract

Classification is a supervised learning method that induces a classification model from a database and is one of the most commonly applied data mining task. The frequently employed techniques are decision tree or neural network-based classification algorithms. This work presents an efficient genetic algorithm (GA) for classification rule mining technique that discovers comprehensible IF-THEN rules using a generalized uniform population method and a uniform operator inspired from the uniform population method. Initial population is generated by methodically eliminating the randomness by generalized uniform population method. In the subsequence generations, genetic diversity is ensured and premature convergence is prevented by the uniform operator. From the experimental results, it was observed that, this method handled the problems of GAs in the task of classification and guaranteed to get rid of any local solution and rapidly found comprehensible rules.

Key Words: *Data Mining, Classification Rules, Genetic Algorithms, Genetic Algorithm Performance*

1. Introduction

Data mining (DM) is a rapidly evolving art and science of discovering and exploiting new, useful, and profitable relationships in data that is awaking great interest in topics such as decision making, information management, medical diagnosis, performance prediction, and many other applications [1]. Classification is a well-recognized DM task and it has been studied extensively in the fields of statistics, pattern recognition, decision theory, machine learning literature, neural networks, etc. Classification operation usually uses supervised learning methods that induce a classification model from a database. The objective is to predict the class that an example belongs to.

Decision trees are the most frequently used classification methods [2]. The construction of an induction decision tree is divided into two stages. First, creating an initial, large decision tree using a set of training set. Second, pruning the initial decision tree, if applicable, using a validation set. Decision tree with boosting is a method that generates a sequence of decision trees from a single training set by re-weighting and re-sampling the samples in the set [3]. Briefly, these methods generate a tree structure from which the classification is interpreted. There are also another techniques that have been used for classification such as neural networks, naive Bayes, etc.

GAs are stochastic search methods, which have been inspired by the process of biological evolution [4]. Because of GAs' robustness and their uniform approach to large number of different classes of problems, they have been used in many applications. Data mining is also one of the important application fields of GAs. In data mining, a GA can be used either to optimize parameters for other kinds of data mining algorithms or to discover knowledge by itself. In this latter task the rules that a GA finds are usually more general because of its global search nature. In contrast, most of the other data mining methods are based on the rule induction paradigm, where the algorithm usually performs a kind of local search. The advantage of GAs become more obvious when the search space of a task is large.

In this work, a new and an efficient GA is used to perform the task of classification. Several GAs have been designed for discovering classification rules [5, 6, 7, 8, 9]. The presented GA is designed according to some concepts of DM, where the goal is to reach accurate and comprehensible knowledge [9]. Hence, the user can understand the system's results and combine them with his/her knowledge to make a well-informed decision, rather than blindly trusting in the incomprehensible output of a "black box" system. That is why GA's chromosomes encode IF-THEN classification rules similarly to the rules discovered by DM algorithms based on rule induction paradigm.

Two key issues in the proposed approach are eliminating the randomness in the phase of generating the initial population by using an efficient initial population generation method and to protect this quality in each generation of GA by using the uniform operator [10]. These methods are inspired by the uniform population [11, 12, 13], which distributes the initial population in the feasible region in a systematic way.

This paper is organized as follows. Section 2 is detailed description of the proposed method. Section 3 briefly describes the data sets used in the experiments. Section 4 discusses the experimental results. Finally, section 5 concludes the paper.

2. The Proposed Genetic Algorithm

2.1. Encoding

Let n be the number of predicted attributes in the data being mined. Then a chromosome is composed of n genes, where each gene corresponds to a condition containing one attribute. Each ith gene is partitioned into three fields: flag (F_i), relational operator (RO_i), and value (V_i) as shown in Fig. 1. A chromosome corresponds to the entire IF part of the rule and each gene corresponds to one condition in this IF part. The chromosomes do not involve the class predicted by a rule. In a given run of the GA, all chromosomes are searching for rules predicting the same class. The GA is run at least once for each class (value of the goal attribute).

The flag field (F_i) is a binary-valued variable in the range. This variable indicates whether or not the corresponding attribute is involved in the rule. 1 shows that the corresponding condition will be involved in the rule while 0 shows the condition will be removed from the rule. Although each chromosome has a fixed length, the genes are interpreted in such a way that the rule has a variable length. Hence, different chromosomes correspond to rules with different number of conditions.

The relational operator (RO_i) field is a variable related to categorical or continuous range of the ith condition. It indicates the relational operator used in the ith condition. If the attribute is categorical, this field can involve the operators "=" and "≠". If the attribute is continuous, this field can involve the operators "<" and "≥". The value (V_i) field involves one of the values belonging to the domain of the attribute.

Gene$_1$			...	...	Gene$_n$		
F$_1$	RO_1	V_1	...	...	F_n	RO_n	V_n

Figure 1. Chromosome representation.

2.2. Generalized uniform population

All genetic solutions have been done by means of the creating initial population randomly. However this kind of method has some drawbacks. Initial population may be created in the infeasible region, or all the chromosomes in population may be in the nearest neighborhood and far away from a solution, or search of solution may stick to a local solution and we can not get rid of this local solution. Local solution in the task of classification is the rules that have no agreeable generalization ability. In this study, creating initial population is performed in a systematic way inspired by the uniform population method [11, 12, 13].

In this method, it is assumed that the range of genes in the chromosomes has been known. First, two chromosomes are generated according to these ranges all genes of which is in the lower bound and the other of which is in the upper bound of this range. Let these chromosomes be C$_0$ and C$_1$ respectively as shown in Figure 2, and A$_i$s represent set of genes with equa-length, if possible.

$C_0 = A_0$
$C_1 = \overline{A}_0$

Figure 2. Initial chromosomes.

C$_1$ can be thought as the complement of C$_0$and the other chromosomes will be generated based on the complementation. The proposed method will be explained for binary encoding but this method can also be applied to other types of encoding. In uniform population method, 2^r-1 new chromosomes are generated from a chromosome generated randomly. Here, if the population size is fixed, 2^r-2 new chromosomes will be generated in a systematic way. Let |P| be the fixed population size. Then, maximum r values ensuring the inequality

$$2^r \leq |P| \tag{1}$$

is found. Then r is decreased by 1 and new 2^r-2 chromosomes are generated according to this new r value. This process is continued until the remaining number of chromosomes reaches to 2. In the last phase more chromosomes than the actual remaining number of chromosomes may be generated. In this case, only some of them are selected for the initial population to make the population size fixed.

For simplicity, let the population size, |P|, be 54. Then the chromosomes will be generated in this as follows: First two chromosomes will be generated as shown in Fig. 2. Then according to the formula in (1), r will be found as 5 ($2^5 \leq 52$). Thus by using this value of dividing factor r, C$_0$ will be divided into 5 parts. This process is shown in Fig. 3. Division is performed on genes, that is, division points are between two sequential genes. Here 30 (2^5-2) new chromosomes are generated according to C$_0$. The shaded area in each chromosome represents the complement of corresponding genes in C$_0$.

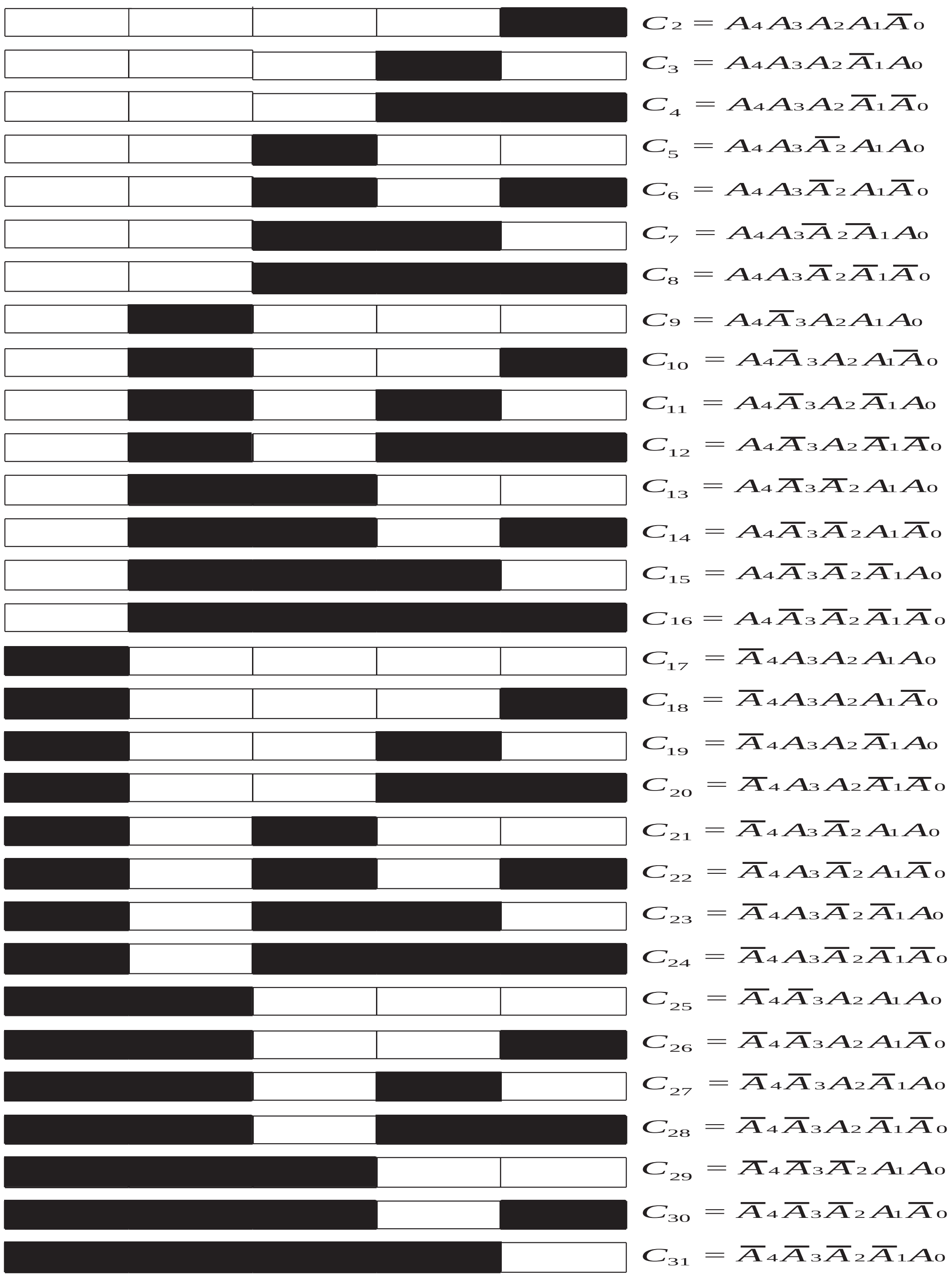
$C_2 = A_4A_3A_2A_1\overline{A}_0$
$C_3 = A_4A_3A_2\overline{A}_1A_0$
$C_4 = A_4A_3A_2\overline{A}_1\overline{A}_0$
$C_5 = A_4A_3\overline{A}_2A_1A_0$
$C_6 = A_4A_3\overline{A}_2A_1\overline{A}_0$
$C_7 = A_4A_3\overline{A}_2\overline{A}_1A_0$
$C_8 = A_4A_3\overline{A}_2\overline{A}_1\overline{A}_0$
$C_9 = A_4\overline{A}_3A_2A_1A_0$
$C_{10} = A_4\overline{A}_3A_2A_1\overline{A}_0$
$C_{11} = A_4\overline{A}_3A_2\overline{A}_1A_0$
$C_{12} = A_4\overline{A}_3A_2\overline{A}_1\overline{A}_0$
$C_{13} = A_4\overline{A}_3\overline{A}_2A_1A_0$
$C_{14} = A_4\overline{A}_3\overline{A}_2A_1\overline{A}_0$
$C_{15} = A_4\overline{A}_3\overline{A}_2\overline{A}_1A_0$
$C_{16} = A_4\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$
$C_{17} = \overline{A}_4A_3A_2A_1A_0$
$C_{18} = \overline{A}_4A_3A_2A_1\overline{A}_0$
$C_{19} = \overline{A}_4A_3A_2\overline{A}_1A_0$
$C_{20} = \overline{A}_4A_3A_2\overline{A}_1\overline{A}_0$
$C_{21} = \overline{A}_4A_3\overline{A}_2A_1A_0$
$C_{22} = \overline{A}_4A_3\overline{A}_2A_1\overline{A}_0$
$C_{23} = \overline{A}_4A_3\overline{A}_2\overline{A}_1A_0$
$C_{24} = \overline{A}_4A_3\overline{A}_2\overline{A}_1\overline{A}_0$
$C_{25} = \overline{A}_4\overline{A}_3A_2A_1A_0$
$C_{26} = \overline{A}_4\overline{A}_3A_2A_1\overline{A}_0$
$C_{27} = \overline{A}_4\overline{A}_3A_2\overline{A}_1A_0$
$C_{28} = \overline{A}_4\overline{A}_3A_2\overline{A}_1\overline{A}_0$
$C_{29} = \overline{A}_4\overline{A}_3\overline{A}_2A_1A_0$
$C_{30} = \overline{A}_4\overline{A}_3\overline{A}_2A_1\overline{A}_0$
$C_{31} = \overline{A}_4\overline{A}_3\overline{A}_2\overline{A}_1A_0$

Figure 3. New chromosomes with r = 5.

Then r is reduced by 1 and the value is 4. 2^4-2=14 new chromosomes will be generated according to C_0. This process is shown in Fig. 4.

In the next step r = 3 and 6 new chromosomes will be generated as shown in Fig. 5, and at the last step r = 2 and 2 new chromosomes will be generated as shown in Fig. 6. In fact, the initially generated two chromosomes, C_0 and C_1, corresponds to the situation where r = 1.

By this method, initial population is distributed in the feasible region uniformly. Chromosomes are not far away from the solution and are prevented to go and remain on stuck to a local solution. Local solutions in the task of classification are the rules that have no agreeable generalization ability.

This method can also be applied to all types of encoding such as binary encoding, floating point-based encoding, string-based encoding [11]. In real-type encoding, a random number can be generated in the half-open interval [0,1) and each gene complemented in the binary encoding are multiplied by this random number.

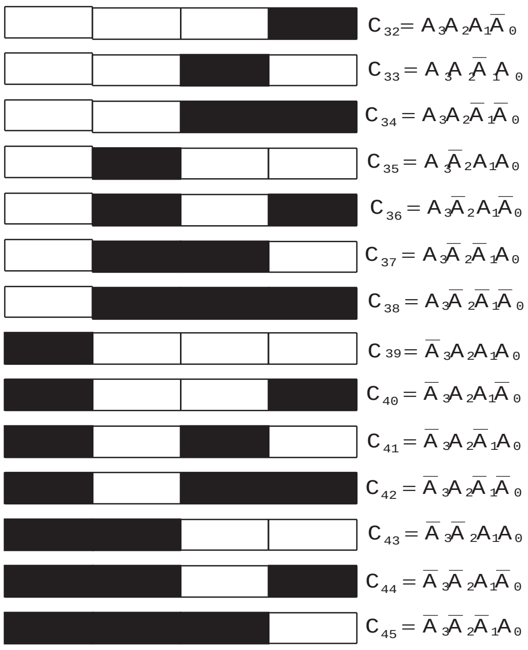
$C_{32} = A_3 A_2 A_1 \bar{A}_0$
$C_{33} = A_3 A_2 \bar{A}_1 A_0$
$C_{34} = A_3 A_2 \bar{A}_1 \bar{A}_0$
$C_{35} = A_3 \bar{A}_2 A_1 A_0$
$C_{36} = A_3 \bar{A}_2 A_1 \bar{A}_0$
$C_{37} = A_3 \bar{A}_2 \bar{A}_1 A_0$
$C_{38} = A_3 \bar{A}_2 \bar{A}_1 \bar{A}_0$
$C_{39} = \bar{A}_3 A_2 A_1 A_0$
$C_{40} = \bar{A}_3 A_2 A_1 \bar{A}_0$
$C_{41} = \bar{A}_3 A_2 \bar{A}_1 A_0$
$C_{42} = \bar{A}_3 A_2 \bar{A}_1 \bar{A}_0$
$C_{43} = \bar{A}_3 \bar{A}_2 A_1 A_0$
$C_{44} = \bar{A}_3 \bar{A}_2 A_1 \bar{A}_0$
$C_{45} = \bar{A}_3 \bar{A}_2 \bar{A}_1 A_0$

Figure 4. New chromosomes with r = 4.

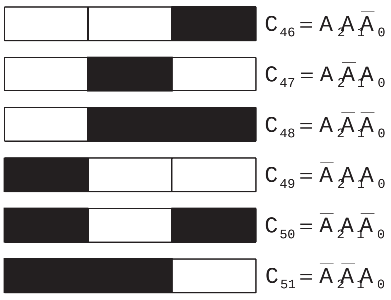
$C_{46} = A_2 A_1 \bar{A}_0$
$C_{47} = A_2 \bar{A}_1 A_0$
$C_{48} = A_2 \bar{A}_1 \bar{A}_0$
$C_{49} = \bar{A}_2 A_1 A_0$
$C_{50} = \bar{A}_2 A_1 \bar{A}_0$
$C_{51} = \bar{A}_2 \bar{A}_1 A_0$

Figure 5. New chromosomes with r = 3.

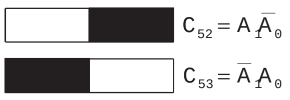

Figure 6. New chromosomes with r = 2.

2.3. Fitness function

The fitness function combines two indicators commonly used in medical domains, namely the sensitivity and the specificity. Table 1 will be used to understand these terms.

Table 1. Calculation of accuracy.

		Disease		
		Positive	*Negative*	
Test	Positive	*True Positive (TP)*	*False Positive (FP)*	*TP + FP*
	Negative	*False Negative (TP)*	*True Negative (FP)*	*FN + TN*
		TP + FN	*FP + TN*	

TP = number of true positive instances
FP = number of false positive instances
FN = number of false negative instances
TN = number of true negative instances

Sensitivity is the probability of the test of finding disease among those who have the disease or the proportion of people with disease who have a positive test result.

$$\text{Sensitivity } = TP/(TP + FN) \tag{2}$$

Specificity is the probability of the test finding NO disease among those who do NOT have the disease or the proportion of people free of a disease who have a negative test.

$$\text{Specificity } = TN/(TN + FP) \tag{3}$$

Finally, the fitness function used for classification is defined as the product of these two indicators:

$$\text{Fitness} = \text{Sensitivity* Specificity} \tag{4}$$

Each run of GA solves a two-class classification problem, where the goal is to predict whether or not the patient has a given disease. When computing the fitness of the rule of the given class, all other classes are effectively merged into a large class, which can be conceptually thought of as meaning that the patient does not have the disease predicted by the rule.

2.4. Uniform operator

An efficient operator, proposed in [10] has been used to obtain high quality chromosomes in each generation of the GA. In each generation, based on the problem, four or more high quality chromosomes from two different best chromosomes are generated; genetic diversity is ensured and early convergence is prevented. For binary encoding, the positions of different bits in these chromosomes are saved and the array size of which is the number of different position bits is generated. Then, from this array, four different arrays are generated similar to uniform population method for r = 2 and the bit values in these arrays are distributed to the different bit positions in the best chromosomes.

For example, let C_0 and C_1 be the best two chromosomes that are expressed with 20 bits in a generation as shown in Table 2. Six bits are different and a 6-bit array is randomly generated. Let the generated array be 001101. For r = 2, the other generated arrays are 001010, 110101, and 110010. This randomly generated array and other three arrays that have been derived from this array are distributed to the different bit positions in one of the best chromosomes. New chromosomes are shown in Table 2. This operator has been performed on the flag of genes. That is why binary encoding has been used for this operator. Surely, this operator can also be applied to all types of encoding such as binary encoding, floating point-based encoding, string-based encoding with small modifications as was done in uniform population method in [11].

Table 2. Best two chromosomes and differences.

C_0	0	1	0	1	1	0	0	1	1	1	0	1	0	1	0	0	0	1	1	1
C_1	0	1	0	1	0	0	1	1	1	1	1	1	0	1	0	0	1	0	0	1
Y	0	1	0	1	*	0	*	1	1	1	*	1	0	1	0	0	*	*	*	1

Table 3. New chromosomes.

A	0	1	0	1	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1	1
B	0	1	0	1	0	0	0	1	1	1	1	1	0	1	0	0	0	1	0	1
C	0	1	0	1	1	0	1	1	1	1	0	1	0	1	0	0	1	0	1	1
D	0	1	0	1	1	0	1	1	1	1	0	1	0	1	0	0	0	1	0	1

2.5. Other genetic operators

Roulette wheel selection and uniform crossover with probability = 100% are used with the elitist strategy. Three mutation operators convenient to the chromosome representation called flag mutation, relational operator mutation and value mutation are used. Mutation rates are 30%. Flag mutation simply takes the inversion of the bit by changing $0 \rightarrow 1$, and $1 \rightarrow 0$. Hence, removal or insertion of the condition in a rule can be regulated.

The relational operator mutation suitably modifies the currently used relational operator in a condition of the rule. This is implemented by replacing the currently used operator with a randomly generated among the valid relational operators depending on categorical or continuous range of the attribute.

The value mutation modifies the values of the genes by replacing the current values with randomly generated ones. If the attribute is categorical, the value mutation simply replaces the current value with another value belonging to the domain of the attribute. If the attribute is continuous a small number is added to or subtracted from the current value in such a way that the lower and upper bounds of the attribute domains are never exceeded.

3. The Used Data Sets

The data sets used in the experiments were obtained from the UCI (University of California at Irvine) – Machine Learning Repository. These data sets have extensively used for classification tasks [14].

3.1. Breast cancer data set

Prognosis of recurrence is important because for about 30% of patients that undergo a breast cancer operation, the illness reoccurs after five years [15]. There are 286 instances, each with 9 attributes with two possible classes of "no recurrence" and "recurrence". All attributes are categorical.

3.2. Dermatology data set

An important problem in dermatology is the differential diagnosis of erythemato-squamous diseases. They all share the clinical features of erythema and scaling with very few differences. The diseases in this group are psoriasis, seboreic dermatitis, lichen planus, pityriasis rosea, chronic dermatitis and pityriasis rubra pilaris. Some characteristics of the diseases are described in [14].

The data set contains 366 instances, each with 34 attributes. In the data set, the family history feature takes on the value 1 if any of these diseases has been observed in the family of the patient, 0 otherwise. The age feature simply represents the age of the patient. Every other feature is given a degree in the range 0–3. Here, 0 indicates that the feature is not present, 3 indicates a great incidence of that feature.

4. Experimental Results

54 chromosomes were generated as the initial population and the GA was terminated when the best fitness did not change continually throughout 10 generations. For each classification rule, the GA was run 3 times, and the rules with generation numbers were obtained. Approximately, this process took 40 generations.

Other set parameters were defined in Section 3. In the proposed method, for categorical attributes, the flags with 0 indicate the lower bounds of the attributes, and $\neq$ operator is assumed as the lower bounds of the genes. The flags with 1 indicate the upper bounds of the corresponding attributes, and $=$ operator is assumed as the upper bounds. For continuous attributes, the flags with 0 indicate the lower bounds of the attributes, and $<$ operator is assumed as the upper bounds of the genes. The flags with 1 indicate the upper bounds of the corresponding attributes, and $\geq$ operator is assumed as the upper bounds.

Each data set was randomly partitioned into two parts in such a way that the portion of examples belonging to each class was kept the same in both the training set (2/3 of the instances) and the test set (1/3 of the instances). Once the rules were mined, they were evaluated on a separate test set. The algorithm cannot cope with the missing values. That is why, in each data set the few instances that contained missing values were simply removed.

The probability of the chromosomes' with many numbers of attributes having fitness value different from 0 is too small. In the same way, the probability of the chromosomes with a little numbers of attributes having fitness value different from 0 is too large. With the proposed initial population generation method, it is guaranteed that chromosomes that have different number of attributes from each other, and that have fitness values different from 0 have been generated. Thus, the restriction in the number of genes in the chromosomes has been removed.

In the random initial population method, it is likely to generate same chromosomes. By the proposed method, this disadvantage has been removed and the chromosomes are uniformly distributed in the feasible region and genetic diversity is guaranteed. Besides, the promising results have been more rapidly found, when compared to results in [9].

From the experimental results, it was demonstrated that, this method coped with the problems of GAs in the task of classification and guaranteed to get rid of any local solution and rapidly found comprehensible rules.

4.1. Results for the breast cancer data set

The rules obtained by the proposed GA are shown in Table 4. Third column of the table shows fitness of the rule computed by GA fitness function, in the training set and in the test set respectively. The generalization of the rules is not so good. This seems to be due to the fact that this is a considerably difficult classification problem and the data is quite noisy.

4.2. Results for the dermatology data set

In the Table 5, 6 rules discovered by GA, one rule for each class, are shown. It can be seen that all the rules discovered from the training data have a very good ability to generalize for the examples in the test set. In all cases the fitness found in the test set is equal to the fitness found in the training set.

Table 4. Discovered rules for the breast cancer data set.

C	Rule	Fitness
1	Inv-nodes<12-14 AND node-caps<>yes AND deg-malig<3 AND irradiat=no	0,518-0,246
2	Age<70-79 AND tumor-size>=20-24 AND inv-nodes <21-23 AND deg-alig>=2	0,426-0,262

Table 5. Discovered rules for the dermatology data set

C	Rule	Fitness
1	Clubbing of the rete ridges>=1 AND elongation of the rete ridges>=1 AND spongiosis<1	0,973-0,973
2	Koebner phenomenon<>1 AND follicular papules=0 AND fibrosis of the papillary dermis<1 AND thinning of the suprapapillary epidermis<2AND munro microabcess<1 AND saw-tooth appearance of retes<1	0,881-0,881
3	Elongation of the rete ridges<1 AND inflammatory monoluclear inflitrate>=1 AND band-like infiltrate>=2	1-1
4	Itching<3 AND polygonal papules<1 AND knee and elbow involvement<1 AND PNL infiltrate<2 AND elongation of the rete ridges <1 AND spongiosis>=0 AND band-like infiltrate<1	0,884-0,884
5	Fibrosis of the papillary dermis>=1 AND sawtoothappearence<2	1-1
6	Itching>=0 AND follicular papules<>0 AND thinning of the suprapapillary epidermis<2 AND saw-tooth appearance of retes <3 AND perifollicular parakeratosis>=1	1-1

5. Conclusion

A GA-based comprehensible rule mining that uses a novel efficient population generation method and an operator has been presented. The GA is proposed as a search strategy to find accurate and comprehensible knowledge within large databases that may be considered as search spaces. The mined rules have been presented in such a way that a user can easily understand the concise set of comprehensible rules. With the proposed method the randomness in generating the initial population step in GAs has been removed and promising results have been obtained using suitable chromosome encoding, fitness function, and uniform operator.

These methods have coped with the problems of GAs and guaranteed to get rid of any local solution that has no generalization ability and rapidly found comprehensible rules. Due to very convenient parallel and distributed architecture of the proposed method we plan a parallel implementation of this method. Further testing on various databases is also in progress to test the robustness of the proposed method.

References

[1] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers Academic Press, 2001.

[2] X. Wang, B. Chen, G. Qian, and F. Ye. "On the optimization of fuzzy decision trees", Fuzzy Sets and Systems, Vol. 2, pp. 117-125, 2000.

[3] Y. Freund, and R.E. Schapire, "Experiments with a new boosting algorithm", in L. Saitta Machine Learning, Proceedings of the Thirteeth International Conference, pp. 148-156, 1996.

[4] Z. Michalewicz, Genetic Algorithms+Data Structures=Evolution Programs, 3^{th} Edition, Springer-Verlag, 1999.

[5] W. Kwedlo, M. Krêtowski, "An Evolutionary Algorithm for Cost-Sensitive Decision Rule Learning", European Conference on Machine Learning, *ECML'01.* Freiburg, Germany. Springer LNCS 2167, 2001.

[6] W. Kwedlo, M. Krêtowski, "Learning Decision Rules using a Distributed Evolutionary Algorithm", Proc. of 8^{th} Workshop Simulation in Research and Development. Gdask, Poland, 2001.

[7] J.L. Álvarez, J. Mata, J.C. Riquelme, "CG03: An oblique classification system using an evolutionary algorithm and C4.5", International Journal of Computer, Systems and Signals, Vol. 2, No. 1, pp. 1 - 15, 2001.

[8] A.A. Freitas, "A Genetic Algorithm for Generalized Rule Induction", Advances in Soft Computing-Engineering Design and Manufacturing, Springer-Verlag, pp. 340-353, 1999.

[9] K.K. Gündoğan, B. Alataş, A. Karcı, Y. Tatar, "Comprehensible Classification Rule Mining With Two-Level Genetic Algorithm", 2^{nd} FAE International Symposium, TRNC, pp. 373-377, 2002.

[10] B. Alataş, A. Karcı, "Genetik Sürecin Düzenlilik Operatörüyle Global Çözüme Doğru Harekete Zorlanması", ELECO'2002, pp. 364-368, 18-22, Bursa 2002.

[11] A. Karcı, A. Arslan, "Uniform population in genetic algorithms" İ.Ü. Journal of Electrical & Electronics, Vol. 2 (2), pp. 495-504, 2002.

[12] A. Karcı, A. Çınar, "Comparison of Uniform Distributed Initial Population Method and Random Initial Population Method in Genetic Search", 15^{th} International Symposium on Computer and Information Sciences, pp. 159-166, Istanbul, Turkey, 2000.

[13] A. Karcı, A. Arslan, "Genetik Algoritmalarda Düzenli Populasyon", GAP IV. Mühendislik Kongresi, pp. 601-607, 2002.

[14] G. Demiröz, H.A. Güvenir, N. Ilter, "Learning differential diagnosis for concept learning", Machine Learning, Vol. 13, pp. 147-165, 1998.

[15] P. Clark, T. Niblett, "Induction in Noisy Domains", Progress in Machine Learning, pp. 11-30, Sigma Press, 1987.