Proceedings of International Conference FACT-2009:
Recent Trends in Computing and Communications, December 18-19, 2009:
K C G College of Technology, Chennai, TN, INDIA:

Discovering Classification Rules by Real Coded MOGA

Dipankar Dutta*
Department of Computer Science and Engineering, University Institute of Technology, The University of Burdwan, Golapbug (North), Burdwan, West Bengal, PIN-713104, India.
Email: dipankar_dutta@rediffmail.com

Paramartha Dutta
Department of Computer and System Sciences, Visva-Bharati University, Santiniketan, West Bengal, PIN-731235, India.
Email: paramartha.dutta@gmail.com

Abstract-**Classification problem is one of the well-studied problems in machine learning. By supervised way we can solve it. First step is the generation of IF-THEN rules at the learning phase from records where class is known. Second step is to use those rules to classify records where class is not known. In this article we are using real coded multi objective genetic algorithms (MOGAs) for generating a set of optimized classification rules. Real-valued attribute ranges are encoded with real-valued genes and we are presenting a new genetic algorithm operator. Rules set are optimized from two objectives – confidence and coverage within the scope of our present scheme. All rules are non-dominated and lie on the pareto-optimal front. Results demonstrate effectiveness of classification rule generation by real coded multi objective genetic algorithm (CRGRCMOGA) by applying it on 8-benchmark data set from UCI machine learning database.**

Index Terms- **Classification, multi objective genetic algorithm, pareto-optimal front, real coded.**

I. INTRODUCTION

Real-world optimization problems are often having several characteristics, which make them difficult to solve up to a satisfactory level. Those characteristics are [4]: 1) Existence of mixed type of variables (such as boolean, discrete, integer and real). 2) Existence of non-linear constraints. 3) Existence of multiple conflicting objectives. 4) Existence of multiple optimum (local and global) solutions. 5) Existence of stochasticities and uncertainties in describing the optimization problem.

Rule generation for classification is one interesting optimization problem having most of the above-mentioned characteristics. We are using genetic algorithms (GAs) for generation of classification rules. GA is a kind of optimization method based on Darwin's natural selection principle. From coding point of view we can categorize GAs into two categories, binary coded GAs (BCGAs) and real coded GAs (RCGAs). In BCGAs, binary numbers codes design variables or genes. A set of genes constructs a chromosome and a set of chromosomes builds a population. In a previous work Dutta and Dutta have applied BCGAs for classification rule generation [5]. However, it is reported that it is difficult to find the high accuracy solution in continuous problems [2]. Also, the search space associated to the objective function is totally different from the one before the coding and the one after the coding. Thus, when users are in search of optimum and the problems are continuous, it is easy to understand the search space of the problem in a real valued space. This is one motivation behind choosing RCGAs. Several RCGAs have been introduced [6], [12]. Crossover and mutation operations in RCGAs are different from BCGAs. In this article, we propose a new GA operator called substitution operator.

In many real world situations there may be more than one objective to be optimized

simultaneously to solve a certain problem called multi-objective optimization problem (MOOP). Contrary to this there are some simpler optimization problems called single objective optimization problems (SOOPs) where only one objective is to be maximize or minimize at a time. Classification rule generation is one MOOP. The multi-objective optimization can be formally stated as following [1]: Find the vector $\mathbf{x}^*=[x^*_1,x^*_2,\ldots,x^*_v]^T$ of v decision variables, which optimizes the objective function vector $\mathbf{f}(\mathbf{x})=[f_1(\mathbf{x}),f_2(\mathbf{x}),\ldots,f_k(\mathbf{x})]^T$ satisfying some equality and inequality constraints. A decision vector $\mathbf{x}^*$ is called Pareto-optimal if and only if there is no $\mathbf{x}$ that dominates $\mathbf{x}^*$, i.e., there is no $\mathbf{x}$ such that $\forall i\in\{1,2,\ldots,k\}, f_i(\mathbf{x})\leq f_i(\mathbf{x}^*)$ and $\exists i\in\{1,2,\ldots,k\}, f_i(\mathbf{x})\leq f_i(\mathbf{x}^*)$ Pareto optimality usually admits a set of solutions called non-dominated solutions. MOGAs are designed regarding two common goals 1) fast and reliable convergence to the Pareto set and 2) a good distribution of solutions along the front. This is in contrast with the approach of aggregating the objectives into a scalar function and solving the resulting SOOP.

Generated classification rules are of the form IF (Antecedent) THEN (Consequence). Antecedent and Consequence are represented as A and C respectively. C is the user specified goal attribute or class label and A is a set of conditions predicting C. Each rule in a rule set matches only a subset of all possible input instances, while the set of rules together is expected to cover the whole input space. Association rule mining is another type of important data mining task, which is also having IF-THEN form. Although they are having identical form there is a big difference between them. In association rule C may have more than one attribute but in classification rule C always involves one attribute.

Rules should be as accurate as possible and they should cover as much records as possible. So two objectives of MOGAs are maximization of confidence or accuracy and coverage simultaneously. We can define confidence by

$$\text{Confidence} = \frac{\text{SUP(A \& C)}}{\text{SUP(A)}} \qquad (1)$$

SUP(A&C) is the number of records that satisfy antecedent A as well as consequent C.

On the contrary SUP(A) is the number of records satisfies all conditions in antecedent A.

We can define coverage by

$$\text{Coverage} = \frac{\text{SUP(A \& C)}}{\text{SUP(C)}} \qquad (2)$$

Where SUP(C) is the number of records having class label as in consequent C.

This paper is organized as follow: Section II describes how MOGAs generate optimized classification rule set. Section III contains experimental results, which were obtained from classification rule sets applied on some benchmark databases and comparison with other methods. Finally, Section IV presents a brief summary and conclusions.

II. PROPOSED APPROACH

As we are using RCGAs, several preprocessing steps needed in BCGAs such as conversion of categorical attributes into continuous attributes and discretization of attributes describes in [8], [13], [7], [5] are not necessary. As we are not using databases having missing values, data cleaning step is also not necessary. Out of filter approach (attributes elimination algorithm is not dependent on data mining algorithm) and wrapper approach (attributes elimination algorithm is dependent on data mining algorithm) the first approach is followed where attributes not relevant for classification are eliminated from database.

A. Storing information about attributes

At the time of loading the database we are storing the name of attributes, all distinct values of attributes in ascending order, types of attributes for later use. Information about attributes for Iris Database and Tic-Tac-Toe database from UCI machine learning repository [16] are shown in Table 1 and Table 2 respectively.

Table 1: Information about attributes of Iris database

Attribute Name	Distinct Values	Type of Attributes
Sepal Length	,4.3,4.4,4.5,4.6,4.7,4.8,4.9,5.0,5.1,5.2,5.3,5.4,5.5,5.6,5.7,5.8,5.9,6.0,6.1,6.2,6.3,6.4,6.5,6.6,6.7,6.8,6.9,7.0,7.1,7.2,7.3,7.4,7.6,7.7,7.9,	Float
Sepal Width	,2.0,2.2,2.3,2.4,2.5,2.6,2.7,2.8,2.9,3.0,3.1,3.2,3.3,3.4,3.5,3.6,3.7,3.8,3.9,4.0,4.1,4.2,4.4,	Float
Petal Length	,1.0,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.9,3.0,3.3,3.5,3.6,3.7,3.8,3.9,4.0,4.1,4.2,4.3,4.4,4.5,4.6,4.7,4.8,4.9,5.0,5.1,5.2,5.3,5.4,5.5,5.6,5.7,5.8,5.9,6.0,6.1,6.3,6.4,6.6,6.7,6.9,	Float
Petal Width	,0.1,0.2,0.3,0.4,0.5,0.6,1.0,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9,2.0,2.1,2.2,2.3,2.4,2.5,	Float
Class Label	,Iris-setosa,Iris-versicolor,Iris-virginica,	String

Table 2: Information about attributes of Tic-Tac-Toe database

Attribute Name	Distinct Values	Type of Attributes
top left square	,b,o,x,	String
top middle square	,b,o,x,	String
top right square	,b,o,x,	String
middle left square	,b,o,x,	String
middle middle square	,b,o,x,	String
middle right square	,b,o,x,	String
bottom left square	,b,o,x,	String
bottom middle square	,b,o,x,	String
bottom right square	,b,o,x,	String
ClassLabel	,negative, positive,	String

B. Encoding chromosome

In this context, there are Pittsburgh approach [15] (several-rules-per-individual encoding) and Michigan approach [9] (one-rule-per-individual encoding) in our work we follow Michigan approach. We are encoding a single classification rule representing one class label only. Only antecedent parts need to be coded into chromosomes. Consequence parts not required to be coded. Reason for this is explained in next subsection. Before first run of GA 10% records of the databases are chosen for constructing initial population. The motivation behind choosing 10% records of the databases for building initial population is to generate population proportional to the database size. Genes representing attributes in these chromosomes are having two parts- minimum value and maximum value as min|max. Initially, we are building chromosome by taking minimum value from one record and maximum value from other record having same class label for all attributes. Out of all attributes, random combinations of attributes will be used for building all chromosome for generations of GA, when we are building population by choosing data from original database One chromosome from Iris database may be 2.1|2.3|6.2|7.6|3.0|3.4| and one chromosome from Tic-Tac-Toe database may be b|o|x|x|b|x|b|x|. Chromosomes are forming antecedent(A) part of the rule.

C. Assigning class label

When we are building population by choosing data from original database, class label of chromosome is same as the class label of original data. After applying GA operators for a chromosome, we are counting the no of instances when A is true for every class and denote it as A_{ij} where i =no of chromosomes and j =no of classes. We are also counting the number of instances of every class and denoting it as C_j where j =no of classes. For every chromosome we are calculating A_{ij} /C_j=(Number of instances where A is true of i^{th} chromosome and j^{th} class)/(Number of instances of j^{th} classes). We are assigning class label as C_i to the chromosome for which class A_{ij} /C_j is highest where i =no of chromosomes.

D. Calculating Objectives

For each chromosome, we count number of instances, 1) where A is true, 2) labeled class or C is true, 3) A and C is true and denote them as A_i, C_i, $A\&C_i$ respectively. Next, we compute confidence $A\&C_i/A_i$ and coverage $A\&C_i/C_i$ and denote it as $Confidence_i$ and $Coverage_i$ respectively.

E. Crossover

Crossover operation in RCGAs is different from that in BCGAs. As described in [3], there are several crossover methodologies like linear crossover, native crossover, blend crossover, simulated binary crossover, fuzzy recombination crossover, unimodal normally distributed

crossover, simplex crossover, fuzzy connectives based crossover, unfair average crossover etc. Single point crossover with crossover probability 0.5 is used in all our experiments. Crossover may take place at any place of chromosomes having "|" i.e., we are using native crossover method in which cross-sites are allowed to be chosen at the variable boundaries.

F. Substitution

Described in [3] there are several mutation operators for RCGA like random mutation, non-uniform mutation, normally distributed mutation, polynomial mutation etc. All these operators will generate numbers randomly or based on some function on values of parameters of parent chromosomes. As we are generating rules with crisp boundaries so boundary of the rule will takes any value of the training data. Therefore we are using native crossover method and in place of mutation operator we are using a new operator named substitution operator. We are substituting a value in chromosome with any value of that attribute from the training data chosen randomly from "all distinct values of attributes" collected at the time of loading the database. Probability of substitution is 0.1.

G. Eliminate unnecessary attribute

Due to crossover and substitution operation some genes representing attributes of some rules may take minimum value and maximum value of that attribute, i.e. genes minimum value and maximum value cover all possible value of that attribute. In these cases we are replacing minimum value and maximum value by "-99.0" as don't care values for eliminating those attributes from rules.

H. Combination and Elite Selection

After every run of GA, we are combining parent chromosomes and child chromosomes. For every class, we are selecting rules, which are lying on the pareto-optimal font using pareto dominance concept.

I. Selection for building population for the next generation of GA

Among all rules survived after elite selection, we are selecting one rule randomly.

If all antecedent attributes of that selected rules having valid values, we are selecting all rules having valid values for all antecedent attributes. From these rules by selecting random combinations of attributes we are building chromosomes for the next generation.

Else if all antecedent attributes don't have valid values then we are selecting attributes having valid values. We are selecting all rules having valid values for same set of attributes and invalid values for same set of attributes as of the first one. From these rules using set of attributes having valid values we are building chromosomes for the next generation.

For every alternative run of GA and if number of rules selected is less than 2 (at least two chromosomes are required for crossover operation) 10% records of the databases are chosen for constructing population of chromosomes for the next generation of GA. This is required to generate chromosomes of various combinations of attributes. The length of the chromosomes are varying from one generation of GA to another generation of GA but for a particular run of GA length of all chromosomes are same, by which we are avoiding many complexities such as designing crossover operator for variable length GAs.

III. RESULTS AND COMPARISON

We are trying to maximize confidence and coverage simultaneously. Each rule has a particular class label so that a rule covers some records of that class. As MOGA generates a set of rules that lie on the pareto optimal front, we should measure how this rule set works on the whole database. For a particular record and a particular rule, for each attribute of the record, we are checking whether the value of the attribute is within the range specified by maximum and minimum value of that attribute in the antecedent part of the rule. For categorical

data, maximum and minimum values are occupying two specific positions in the corresponding "all distinct values of attributes". Any categorical attribute value in a record will be checked whether that value falls within that or not. If for all attributes in the antecedent part of the rule, a record's corresponding attributes values fall within the range then that rule can be used to classify that record. By applying all generated rules on a record if a single rule is found which is satisfying the above-mentioned criteria, we can use that rule for classifying that record. But if more that one rule is found, we are choosing rule with maximum confidence value. If more than one rule is giving same confidence, we are choosing rule with higher coverage. For same confidence and coverage we are choosing a rule randomly. In that manner, we are classifying all data in a particular class by using a particular rule for classification. If there is a match between actual class label and predicted class label, the record is correctly classified; otherwise it is misclassified. So, in the context of the whole database, we can measure accuracy of the rule set produced by MOGA by the following expression of accuracy as per equation (3).

$$\text{Accuracy in } \% = \frac{\text{Number of matches} * 100}{\text{Number of records}} \tag{3}$$

Now it may also happen that for a particular record, no rule is covering that record. In that case, the second objective is not fulfilled. So, in respect of the whole database, we can measure coverage of the rule set by the following expression of coverage as per equation (4).

$$\text{Coverage in } \% = 1 - \frac{\text{Number of records when rule not found} * 100}{\text{Number of records}} \tag{4}$$

We are comparing our method of classification rule generation by real coded multi objective genetic algorithm (CRGRCMOGA) with four classification methods: C4.5 [14], CBA [11], CMAR [10], CRGMOGA [5] in Table 3.

Table 3: Comparison of C4.5, CBA, CMAR, CRGMOGA and CRGRCMOGA on confidence

Database	C4.5	CBA	CMAR	CRGMOGA	CRGRCMOGA
Glass	68.7	73.9	70.1	71.5	75.7
Ionosphere	90	92.3	91.5	89.1	90.9
Iris	95.3	94.7	94	95.3	96.6
Pima	75.5	72.9	75.1	74.9	73.9
Tic Tac Toe	99.4	99.6	99.2	73.6	74.9
Waveform	78.1	80	83.2	75.2	75.3
Wine	92.7	95	95	93.8	91.6
Zoo	92.2	96.8	97.1	100	100

Another objective of CRGRCMOGA is to maximize coverage. Rules generated by CRGRCMOGA in cases of all 8 databases gives 100% coverage.

IV. CONCLUSIONS

In this paper, real coded multi objective genetic algorithm is used for generation of a set of optimized classifications rules. Rules are optimized from two perspectives confidence and coverage. We have also tested rule set from the context of all records in the database, which is giving compatible results in comparison with other methods. We have proposed a new operator called substitution operator in place of mutation operator, which is helping to reduce on of rules in rule set. Discritization methods required in BCGAs can be avoided in RCGAs and thereby the inaccuracies involved with the process. Converting variables into binary and from binary to real values can also be avoided.

REFERENCES

[1] C. A. C. Coello, "A comprehensive survey of evolutionary-based multiobjective optimization techniques," *Knowledge and Information Systems*, vol. 1, no. 3, pp. 269-308, 1999.

[2] L. D. Davis, *Handbook of Genetic Algorithms*. New York: Van Nostrand ReinHold, 1991.

[3] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. Chichester: John Wiley and Sons, 2001.

[4] K. Deb, "Single and Multi-Objective Optimization Using Evolutionary Algorithms," KanGAL, IIT-Kanpur, India, Report KanGAL-2004002, 2004.

[5] D. Dutta, and P. Dutta, "Optimized classification rule set generation using pareto based binary coded elitist multi objective genetic algorithm," *Foundations of Computing and Decision Sciences.*, submitted for publication.

[6] L. J. Eshelman, K. E. Mathias, and J. D. Schaffer, "Crossover operator biases: Exploiting the population distribution," in *Proc. of the 7th Int. Conf. on Genetic Algorithms*, Michigan State University, Michigan, USA, 1997, pp. 354-361.

[7] A. A. Freitas, "A survey of evolutionary algorithms for data mining and knowledge discovery," Postgraduate Program in Computer Science, Pontificia Universidade Catolica do Parana, Rua Imaculada Conceicao, 1155, Curitiba - PR, 80215-901, Brazil, 2000.

[8] J. Han, and M. Kamber, "Data Mining: Concepts and Techniques". 2nd ed., J. Gray, Ed. Morgan Kaufmann, 2006.

[9] J. H. Holland, "Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems," in *Machine Learning: An artificial intelligence approach*, vol. 2, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, Eds. Los Altos, CA, USA: Morgan Kaufmann, 1986, pp. 593-623.

[10] W. Li, J. Han, and J. Pei, "CMAR: Accurate and efficient classification based on multiple class-association rules," in *Proc. of 1st IEEE Int. Conf. on Data Mining*, San Jose, California, USA, 2001, pp. 369-376.

[11] B. Liu, W. Hsu, and Y. Ma, "Integrating classification and association rule mining," in *Proc. of 4th Int. Conf. on Knowledge Discovery in Data Mining*, New York, NY, Aug. 1998, pp. 80-86.

[12] I. Ono and S. Kobayashi, "A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover," in *Proc. of 7th Int. Conf. on Genetic Algorithms*, Michigan State University, Michigan, USA, Aug. 1997, pp. 246-253.

[13] D. Pyle, *Data Preparation for Data Mining*. San Francisco, CA, USA: Morgan Kaufmann, 1999.

[14] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann, 1993.

[15] S. F. Smith, "A learning system based on genetic adaptive algorithms," PhD thesis, University of Pittsburgh, 1980.

[16] UCI repository of machine learning databases, Dept. of Information and Computer Science, Univ. of California, Irvine, CA. [Online]. Available: http://www.ics.uci.edu/learn/MLRepository.html

Dipankar Dutta received his Bachelor of Engineering degree in the year of 1996 from Jalpaiguri Government Engineering College, The University of North Bengal, India. In 2000 he completed PGDIM from IGNOU, India. In 2004 he received his M.Tech in Computer Technology from Jadavpur University,India.

During 1996 to 2003 he worked at various industrial houses like Brite Metalloy Pvt. Ltd. and D. C. Industrial Plant Services Pvt. Ltd. From 2004 he worked in Haldia Institute of Technology, Asansol Engineering College, University Institute of Technology. Presently he is serving as Assistant Professor in Computer Science and Engineering Department of University Institute of Technology, The University of Burdwan, West Bengal, India. His fields of interest are Data Mining, Soft Computing etc.

Dr. Paramartha Dutta did his Bachelors and Masters in Statistics from the Indian Statistical Institute, Calcutta in the years 1988 and 1990 respectively. He afterwards completed his Master of Technology in Computer science from the same Institute in the year 1993 and Doctor of Philosophy in Engineering from the Bengal Engineering and Science University, Shibpur in 2005 respectively.

He has served in the capacity of research personnel in various projects funded by Govt. of India, which include Defence Research and Development Organisation, Council of Scientific and Industrial Research, Indian Statistical Institute, Calcutta etc.

Dr. Dutta is now a Professor in the Department of Computer and system Sciences of the Visva Bharati University, West Bengal, India. Prior to this, he has served Kalyani Government Engineering College and College of Engineering and Management, Kolaghat, both in West Bengal as full time faculty members. Apart from this, he has remained associated to National Institute of Management, Calcutta, Department of Statistics of University of Calcutta, Department of Computer Science and Technology of Bengal Engineering and Science University, Shibpur either as a guest faculty member from time to time.

Dr. Dutta has been involved as the principal investigator funded by the All India Council for Technical Education, Govt. of India. He has co-authored four books and has also one edited book to his credit. He has published about sixty five papers in various journals and conference proceedings, both international and national. Presently, he is supervising three students for their Ph. D programme registered with either Visva Bharati University or West Bengal University of Technology.

Dr. Dutta is a Life Fellow of the Optical Society of India (OSI), Life Member of Institution of Electronics and Telecommunication Engineers (OSI), Computer Society of India (CSI), Indian Science Congress Association (ISCA), Indian Society for Technical Education (ISTE), Indian Unit of Pattern Recognition and Artificial Intelligence (IUPRAI) - the Indian associate of the International Association for Pattern Recognition (IAPR), Member of Associated Computing Machinery (ACM).