Available online at www.sciencedirect.com
ScienceDirect
Applied Soft Computing 8 (2008) 477–487
Applied Soft Computing
www.elsevier.com/locate/asoc

Application of elitist multi-objective genetic algorithm for classification rule generation

S. Dehuri [a,*], S. Patnaik [a], A. Ghosh [b], R. Mall [c]

[a] P.G. Department of Information & Communication Technology, Fakir Mohan University, Balasore 756019, Orissa, India
[b] Machine Intelligence Unit and Center for Soft Computing Research, Indian Statistical Institute, 203 B.T. Road, Kolkata, West Bengal, India
[c] Department of Computer Science & Engineering, Indian Institute of Technology, Kharagpur 721302, West Bengal, India

Received 20 December 2005; received in revised form 9 January 2007; accepted 8 February 2007
Available online 23 March 2007

Abstract

We present an elitist multi-objective genetic algorithm (EMOGA) for mining classification rules from large databases. We emphasize on predictive accuracy, comprehensibility and interestingness of the rules. However, predictive accuracy, comprehensibility and interestingness of the rules often conflict with each other. This makes it a multi-objective optimization problem that is very difficult to solve efficiently. We have proposed a multi-objective genetic algorithm with a hybrid crossover operator for optimizing these objectives simultaneously. We have compared our rule discovery procedure with simple genetic algorithm with a weighted sum of all these objectives. The experimental result confirms that our rule discovery algorithm has a clear edge over simple genetic algorithm.

Keywords: Genetic algorithm; Multi-objective genetic algorithm; Rule mining; Data mining

1. Introduction

Classification rule mining is one of the most studied tasks in data mining community and is an active research area because the data being generated and stored in databases of organizations are already enormous and continues to grow very fast. This large amount of stored data normally contains valuable hidden knowledge, which if harnessed could be used to improve the decision-making process of an organization. For instance, data about previous sales might contain interesting relationships between products, customer segmentation and buying habits of customers. The discovery of such relationships can be very useful to efficiently manage the sales of a company. However, the volume of the archival data often exceeds several gigabytes and sometimes even terabytes. Such an enormous volume of data is beyond the manual analysis capability of human beings. Thus, there is a clear need for developing automatic methods for extracting knowledge from data that not only has a high predictive accuracy but also comprehensible and interestingness by users [1–5]. The user should be able to understand the mining system's results and combine them with his/her knowledge to make a well-informed decision, rather than blindly trusting the incomprehensible output of a ''black box'' system.

Furthermore, individual datasets may be gathered and studied collectively for purposes other than those for which they were originally created. Knowledge with multiple-objectives may be obtained in the process while eliminating one of the largest costs, viz., data collection. Medical data, for example, often exist in vast quantities in an unstructured format. The application of classification/clustering of data mining can facilitate systematic analysis in such cases. Medical data, however, require a large amount of preprocessing in order to be useful. Here numeric and textual information may be interspersed, different symbols can be used with the same meaning, redundancy often exists in data, erroneous/misspelled medical terms are common, and the data are frequently rather sparse. A robust preprocessing system is required in order to extract any kind of knowledge from even medium-sized medical datasets. The data must not only be cleaned of errors and redundancy, but also organized in a fashion that makes sense to the problem.

Soft computing, a consortium of methodologies, can be a viable tool for handling real-life ambiguous situations [6]. The

* Corresponding author.
E-mail addresses: satchi.lapa@gmail.com (S. Dehuri), srikantapatnaik@hotmail.com (S. Patnaik), ash@isical.ac.in (A. Ghosh), rajib@cse.iitkgp.ernet.in (R. Mall).

1568-4946/$ – see front matter
doi:10.1016/j.asoc.2007.02.009

aim of this consortium is to exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial truth in order to achieve tractability, robustness, and low-cost solutions. Its principal components, at this juncture, are fuzzy logic (FL), neural computing (NC), evolutionary algorithm (EAs), and rough sets (RS). It has been shown that the different soft computing tools are utilized, both in individual and integrated manner, in various ways to develop rule mining systems based on single objective. However, a very rare attempt has been taken by the soft computing community to solve the rule mining system based on multiple objectives.

Evolutionary algorithms (EAs) have inspired many research efforts for optimization as well as rule generation [7,8]. Traditional rule generation methods, are usually accurate, but have brittle operations. Evolutionary algorithms on the other hand provide a robust and efficient approach to explore large search space. One of the EA called simple genetic algorithm (SGA) introduced by J.H. Holland (1975) [8–10] is good for rule generation satisfying a single objective. However, practical rule generation is naturally posed as multi-objective problems with three criteria: (i) predictive accuracy, (ii) comprehensibility and (iii) interestingness [4]. The SGA normally handles problems with such criteria by single objective problems. In other words, one can transfer the original multi-objective problem into a single-objective problem by using a weighted sum formula. However, this approach is unsatisfactory due to the nature of the optimality conditions for multiple objectives. In the presence of multiple and conflicting objectives, the resulting optimization problem gives rise to a set of optimal solutions, instead of just one optimal solution. Multiple optimal solutions exist because no single solution can be a substitute for multiple conflicting objectives. In order to overcome this difficulty we have proposed an elitist multi-objective genetic algorithm with a hybrid crossover operator that can extract the high-level classification/prediction rules and are represented in the following form.

IF some conditions hold on the values of a set of predicting attributes
THEN predict a value for the goal attribute.

In other words, the value of a special attribute, called the goal attribute, is predicted by the values given for other attributes called the predicting attributes.

In this approach, rule generation is to associate each individual of the population with the same predicted class, which is never modified during the running of the algorithm. We would need to run at least for the specified number of classes. So that in the ith run, the algorithm discovers only rules predicting the ith class [11,12]. As it is difficult to find out a single global solution for a multi-objective problem, so it is natural to find out a set of solutions (i.e. called Pareto-optimal set) depending on the non-dominance criterion in each run. Out of the discovered rules the experimental results demonstrated the rule having the objective function values more balanced towards the associated measures. We shall use the results reported in [12] for comparison with our results. By comparison it has been observed that the predictive accuracy, comprehensibility and interestingness is encouraging in genetic algorithm for multi-objective optimization over SGA.

This paper is organized as follows. Section 2 discusses the SGA for classification rule generation. In Section 3, we describe evolutionary algorithms for multi-objective problems. In Section 4, we have discussed and elitist multi-objective genetic algorithm with a hybrid crossover for rule generation. The implementation of our simulation experiments is discussed in Section 4. Finally, Section 5 concludes this article.

2. Using SGA for classification rule generation

In this section we review the function of SGA for rule generation. Genetic algorithms are probabilistic search algorithms characterized by the fact that a number N of potential solutions (called individuals $I_k \in \Omega$, where Ω represents the space of all possible individuals) of the optimization problem simultaneously samples the search space. This population $P = \{I_1, I_2, \ldots, I_N\}$ is modified according to the natural evolutionary process: after initialization, selection S: $I^N \rightarrow I^N$, recombination $\Re$: $I^N \rightarrow I^N$ and mutation are executed in a loop until some termination criterion is reached. Each run of the loop is called a generation and $P(t)$ denotes the population at generation t.

The selection operator is intended to improve the average quality of the population by giving individuals of higher quality a higher probability to be copied into the next generation. Selection thereby focuses on the search of promising regions in the search space. The quality of an individual is measured by a fitness function f: $P \rightarrow R$. Genetic operators changes the genetic material in the population either by crossover or by mutation in order to obtain new points in the search space. Fig. 1 depicts the steps that are performed in SGA.

The following subsection discusses the individual representations, fitness function, and genetic operators used for classification rule discovery.

2.1. Individual representations

Each individual in the population represents a candidate rule '$\Re$' of the form if A then C. The antecedent of this rule can be formed by a conjunction of at most $n-1$ attributes, where n is the number of attributes being mined. Each condition is of the form $A_i = V_{ij}$, where A_i is the ith attribute and V_{ij} is the jth value of the ith attribute's domain. The consequent consists of a single condition of the form $G = g_l$, where G is the goal attribute and g_l is the lth value of the goal attribute domain.

A string of fixed size encodes an individual with n genes representing the values that each attribute can assume in the rule and each gene is divided into two parts, one is value and the other is flag for indication of the inclusion or exclusion of the attributes in the rule. This encoding is shown in Fig. 2.

If an attribute is not present in the rule antecedent, the corresponding flag value in gene is 0. This value is a flag to indicate that the attribute does not occur in the rule antecedent.

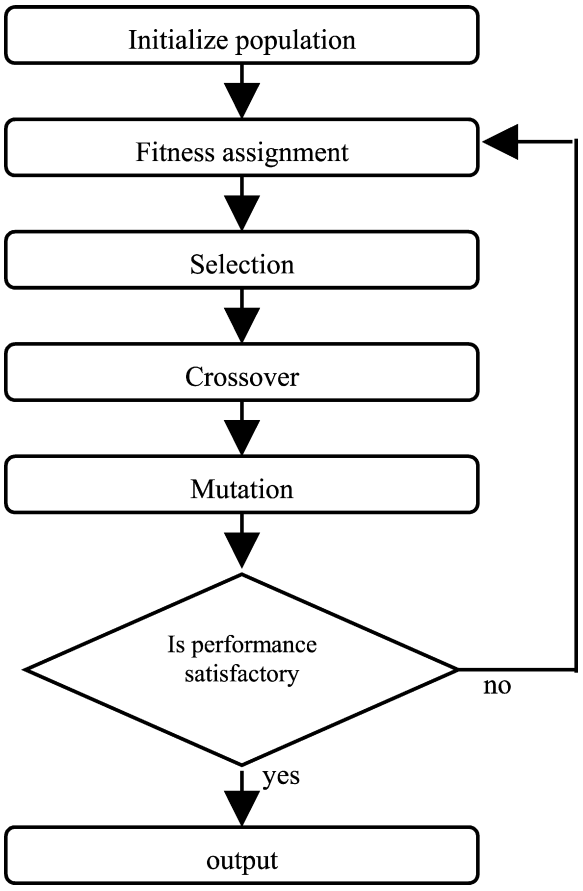

Fig. 1. Flow diagram of SGA.

Hence, this encoding effectively represents a variable-length individual (rule).

2.2. Fitness function

As discussed in Section 1, the discovered rules should have (a) predictive accuracy (b) comprehensibility and (c) interesting. In this subsection we discuss each of these objectives and how these can be incorporated into a single objective fitness function.

2.2.1. Comprehensibility metric

In practice the comprehensibility metric is a kind of subject concept as it varies from user to user. However, the data mining literature uses an objective measure: in general the smaller the rule, more comprehensible it is. There are various ways to measure rule comprehensibility [13–16]. The standard way of measuring comprehensibility is to count the number of rules and the number of conditions in these rules. If these numbers increase then the comprehensibility decreases. If a rule has at most M_c conditions, the comprehensibility ς of the rule $\Re$ can be defined as:

$$\varsigma(\Re) = 1 - \left(\frac{N_c(\Re)}{M_c}\right),$$

where $N_c(\Re)$ is the number of conditions in the rule $\Re$.

The comprehensibility is also measured by the following expression

$$\varsigma(\Re) = M_c - (N_c(\Re)).$$

Confusion matrix	Actual Class		
		C	~C
Predicted Class	C	L_{CC}	$L_{C\sim C}$
	~C	$L_{\sim CC}$	$L_{\sim C\sim C}$

Fig. 3. Confusion matrix to measure predictive accuracy.

2.2.2. Predictive accuracy

As already mentioned, our rules are of the form IF $A_1 = v_{1j} \wedge A_2 = v_{2j}$ THEN C. That means it has two predicting attributes and one goal attribute. In general the antecedent part of the rule is a conjunction of conditions. A very simple way to measure the predictive accuracy of a rule $P(\Re)$ is

$$P(\Re) = \frac{|A\&C|}{|A|}$$

where $|A|$ is the number of instances satisfying all the conditions in the antecedent A and $|A \& C|$ is the number of examples that satisfy both the antecedent A and the consequent C. Intuitively this metric measures predictive accuracy in terms of how many cases both antecedent and consequent hold out of all cases where the antecedent holds. However, it is quite prone to overfitting, because a rule covering small number of instances could have a high value, even though such a rule would not be able to generalize the unseen data during training.

An alternative measure of predictive accuracy of the rule is the number of correctly classified test instances divided by the total number (correctly classified + wrongly classified) of test instances. Although this method is widely used it has a disadvantages of unbalanced class distribution [17].

Hence to avoid these limitations the following measure of predictive accuracy is taken into consideration and are discussed in more detail [17,18]. A confusion matrix can summarize the performance of a classification rule with respect to predictive accuracy.

Let us consider a simplest case, where there are only two classes to be predicted, referred to as the class C and the class ~C. In this case the confusion matrix will be a 2×2 matrix and is illustrated in Fig. 3.

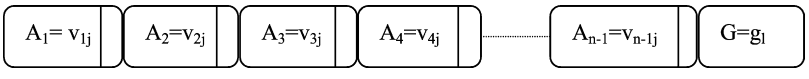

Fig. 2. Chromosome representation.

In Fig. 3, C denotes the class generated by a rule, and all other classes are simply considered as ∼C. The labels in each quadrant of the matrix have the following meaning:

L_{CC} Number of instances satisfying A and having class C
$L_{C\sim C}$ Number of instances satisfying A and having class ∼C
$L_{\sim CC}$ Number of instances not satisfying A but having class C
$L_{\sim C\sim C}$ Number of instances not satisfying A and having class ∼C

Intuitively, the higher the values of the diagonal elements and lower the values of other elements, better the corresponding classification rule. This matrix can also work for a 'm' class problem: when there are more than two classes one can still work with the assumption that algorithm evaluates one rule at a time and the class C predicted by the rule is considered as the Cth class, and all other classes are simply considered as ∼C classes.

Given the values of L_{CC}, $L_{C\sim C}$, $L_{\sim CC}$ and $L_{\sim C\sim C}$ as discussed above, the *predictive accuracy* is defined as

$$p(\Re) = \frac{L_{CC} \times L_{\sim C\sim C}}{(L_{CC} + L_{C\sim C}) \times (L_{CC} + L_{\sim CC})},$$

where $0 \leq p(\Re) \leq 1$.

2.2.3. Interestingness

The third objective of the rule called interestingness has both subjective (or user driven) and objective (or data driven) measures. Subjective measures are dependent on previous knowledge or previous expectations of the user. When using this kind of measure, discovered rule is often considered interesting in the sense of being novel and/or surprising for the user—when it contradicts the previous knowledge or expectation of the user. On the other hand, objective measures try to estimate the interestingness of a rule based on the data being mined. Hence, objective measures tend to be domain independent, whereas subjective measures tend to be domain-dependent. Therefore, it is wise to combine objective and subjective measures rather than being used in a mutually exclusive fashion.

Using general impressions Liu et al. [19] have proposed a subjective approach for selecting interesting rules based on the notion of general impression. In essence, the user specifies his/her general impressions about data relationships in the application domain in an IF–THEN prediction rule like format. For example, a given user might specify the following general impression: IF (%_of_mark = high) THEN (Result = -pass). Note that this is a general impression in the sense that it is quite vague (fuzzy), different from a reasonably precise rule such as: IF (%_of_mark > 60%) THEN (Result = pass). The basic idea is that although the user is not supposed to know reasonably precise rule, he/she can have general impression about the application domain that are valuable clues for the system to determine what is interesting (novel, surprising) for the user.

Once the general impressions are specified, the discovered rules are compared with the general impression. There are essentially two kinds of interesting rules selected by the system. First one includes the rules with unexpected consequent (THEN part). In this case, the conditions in IF part matches the general impression, but the rule's consequent and the impression's consequent are different. The second category includes the rules with unexpected conditions. In this case a rule's consequent matches a general impression's consequent, but the conditions in their antecedent are different. Note that a rule is interesting is no guarantee that it will be accurate and comprehensible. For a more comprehensive discussion about the general impressions approach and related subjective measures of rule interestingness the reader is referred to refs. [19,20].

The general impression of measuring rule interestingness has two main disadvantages: (a) it requires that the user spend some time specifying general impressions, (b) It is application domain dependent, i.e. the general impressions are valid only for the current application domain and possibly only for the current user, since different users of a given application domain might have somewhat different general impressions about that domain.

To cope with these disadvantages one can estimate that a rule is novel and/or surprising based only on objective, data-driven factors, without requiring that the user specify her/his general impressions. Hence, the system would have greater autonomy and generality, being to a large extent independent of the application domain.

Noda et al. [20] has proposed two relatively simple objective measures of rule Surprisingness (or interestingness). The basic idea of one of these measures is that a rule is considered surprising to the extent that it predicts a class different from the classes predicted by its minimum generalizations. Let a rule antecedent be a conjunction of m conditions of the form cond_1 and cond_2 and cond_3 and ... and cond_m. In essence, a rule has m minimum generalizations, one for each of its m conditions. The ith minimum generalization of the rule $i = 1, \ldots, m$ can be obtained buy removing the ith condition from the rule antecedent.

When a minimum generalization of a rule is generated, the system recomputes the class predicted by the generalized rule, which is the majority class of the data instances covered by the generalized rule. Let c be the class predicted by both the original rule and the ith minimum generalization of the original rule. Then the system compares c with each c_i, $i = 1, \ldots m$, and counts the number of times that c differs from c_i. The higher the value of this count the higher the degree of Surprisingness (interestingness) assigned to the original rule.

In other words, the system effectively considers that a rule has a large degree of Surprisingness when attribute interactions make that rule cover a set of data instances whose majority class is different from the majority class of the sets of data instances covered by most of the minimum generalizations of that rule. One can also regard a rule with a large degree of Surprisingness as an exception rule, since it covers fewer data instances and

makes a prediction different from most of its minimum generalizations.

The second objective measure of rule Surprisingness proposed by Frietas [21] is an information theoretic measure.

Thus the computation of the degree of interestingness of a rule, in turn, consists of two terms. One of them refers to the antecedent of the rule and the other to the consequent. The degree of interestingness of the rule antecedent is calculated by an information–theoretical measure, which is a normalized version of the measure proposed by [15]. Initially, as a preprocessing step, the algorithm calculates the information gain of each attribute (InfoGain) [16]. Then the degree of interestingness of the rule antecedent (RInt) is given by:

$$\mathrm{RInt} = 1 - \frac{\sum_{i=1}^{n-1} \mathrm{InfoGain}(\mathrm{A}_i)/(n-1)}{\log_2(|\mathrm{dom}(\mathrm{G})|)}$$

where n is the number of attributes in the antecedent and $(|\mathrm{dom}(\mathrm{G})|)$ is the domain cardinality (i.e. the number of possible values) of the goal attribute G occurring in the consequent. The log term is included in the RInt formula to normalize the value of RInt, so that this measure takes on a value between 0 and 1. The InfoGain is given by:

$$\mathrm{InfoGain}(\mathrm{A}_i) = \mathrm{Info}(\mathrm{G}) - \mathrm{Info}(\mathrm{G}|\mathrm{A}_i)$$

where

$$\mathrm{Info}(\mathrm{G}) = -\sum_{i-1}^{m_k}(P(g_l)\log_2(p(g_l)))$$

$$\mathrm{Info}(\mathrm{G}|\mathrm{A}_i) = \sum_{i=1}^{n_i}\left(p(v_{ij})\left(-\sum_{j=1}^{m_k} p(g_1|v_{ij})\log_2(p(g_l|v_{ij}))\right)\right)$$

where m_k is the number of possible values of the goal attribute G, n_i is the number of possible values of the attribute A_i. The fitness function is computed as the arithmetic weighted mean of comprehensibility, predictive accuracy and interestingness.

Finally, the fitness function is given by:

$$f(x) = \frac{w_1\varsigma(\Re) + w_2 p(\Re) + w_3\mathrm{RInt}}{w_1 + w_2 + w_3},$$

where w_1, w_2 and w_3 are user-defined weights.

2.3. Genetic operators

2.3.1. Crossover

The crossover operator we consider is based on single-point uniform crossover. This operator can also be treated as hybrid crossover because it combines the best attribute of single point and uniform crossover [22,23]. Let us discuss each of these operators briefly and how these operators influence the hybrid one.

In single-point crossover between two individuals a single point is randomly chosen in the range of the length of the chromosomes. Then the genes to the right of the crossover point are swapped between the two individuals, yielding the new offspring. In the uniform crossover, for each gene position the genes from the two parents are swapped with a fixed, position independent probability p. Unlike one-point crossover, the swapped genes do not need to be adjacent to each other. The influence of the value of p in the exploratory power of uniform crossover is as follows. If the value of p is closer to 0.5, then large numbers of genes are swapped between the two parents. Hence the exploratory power of uniform crossover is high, i.e. the search performed by this operator tends to be global. Conversely, if the value of p is closer to 0 or 1, then the number of genes swapped between the two parents is smaller, and so more local searching is performed by this operator.

It has been found that both the operators suffer from search bias. The single-point crossover operator has a strong positional bias, i.e. the creation of new individuals by swapping genes of the parents depends on the position of the genes, in the genome. Similarly, the uniform crossover suffers from distributional bias but it has no positional bias. So the idea here is to combine the best attribute of both the operators and generate a hybrid one-point uniform crossover operator.

2.3.2. Mutation

Mutation is an operator that acts on a single individual at a time. Unlike crossover, which recombines genetic material between two (or more) parents, mutation replaces the value of an attribute into another value belonging to the same domain of the attributes.

Besides crossover and mutation, the insert and remove operators directly try to control the size of the rules being evolved and thereby influence the comprehensibility of the rules. These two operators randomly insert and remove, respectively, a condition in the rule antecedent. These operators are not part of regular GA. The removal operator is also called dropping condition operator.

3. Genetic algorithm for multi-objective problems

3.1. Overview of MOEA

There are many multi-objective problems requiring simultaneous optimization of several competing objectives. Formally, it can be stated as follows.

We want to find $\vec{x} = (x_1, x_2, \ldots, x_n)$ which maximizes the values of 'p' objective functions $F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \ldots f_p(\vec{x}))$ within a feasible domain Ω. Generally the answer is not a single solution but a family of solutions called a *Pareto-optimal set*.

Definitions.

- A vector $\vec{u} = (u_1, u_2, \ldots, u_p)$ is said to dominate $\vec{v} = (v_1, v_2, \ldots, v_n)$ iff $\vec{u}$ is partially greater than $\vec{v}$, i.e. $\forall i \in \{1, 2, 3, \ldots p\}, u_i \geq v_i \wedge \exists i \in \{1, 2, \ldots p\} : u_i \succ v_i$.
- A solution $x \in \Omega$ is said to be Pareto-optimal with respect to Ω iff there is no $x' \in \Omega$ for which $\vec{v} = F(x') =$

$(f_1(x'),\ f_2(x'),\ \ldots,\ f_p(x'))$ dominates $\vec{u} = F(x) = (f_1(x),\ f_2(x), \ldots, f_p(x))$.

- For a given multi-objective problem $F(x)$, the Pareto-optimal set P_s is defined as: $P_s = \langle x \in \Omega | \neg\ \exists x' \in \Omega : F(x') \geq F(x) \rangle$
- For a given multi-objective problem $F(x)$ and Pareto-optimal set P_s, the Pareto front P_f is defined as: $P_f = \langle \vec{u} = F(x) = (f_1(x),\ f_2(x), \ldots f_p(x) | x \in P_s) \rangle$

Optimization methods generally try to find a given number of Pareto-optimal solutions which are uniformly distributed in the Pareto-optimal set, such solutions provide the decision maker sufficient insight into the problem to make the final decision. In Fig. 4, Pareto-optimal front of a set of solutions generated from two conflicting objectives like $f_1(x,y) = 1/(x^2 + y^2 + 1)$ and $f_2(x,y) = x^2 + 3y^2 + 1$, $-3 \leq x,y \leq 3$ is illustrated. Methods such as weighted sum, ε-constraint, and goal programming have been proposed to search for Pareto optima [24,25]. However, an a priori articulation of the preferences to the objectives is required, which is often hard to decide beforehand. Besides, these methods can only find one solution at a time. Other solutions cannot be obtained without re-computation with the free parameters reset.

By contrast, genetic algorithms (GAs) [9] maintain a population and thus can search for many non-dominated solutions in parallel. GA's ability to find a diverse set of solutions in a single run and its exemption from demand for objective preference information renders it immediate advantage over aforementioned techniques. A lot of multi-objective GAs (MOGAs) [26–29] have been proposed. Basically, an MOGA is characterized by its fitness assignment and diversity maintenance strategy.

In fitness assignment, most MOGAs fall into two categories, non-Pareto and Pareto-based. Non-Pareto methods use the objective values as the fitness value to decide an individual's survival. Schaffer's VEGA is such a method. The Predator–prey approach [30] is another one, where some randomly walking predators will kill a prey or let it survive according to the prey's value in one objective. In contrast, Pareto based methods measure individual's fitness according to their dominance property. The non-dominated individuals in the population are regarded as fittest regardless of their single objective values. Since Pareto-based approaches respect better the dominating nature of multi-objective problems, their performance is reported to be better.

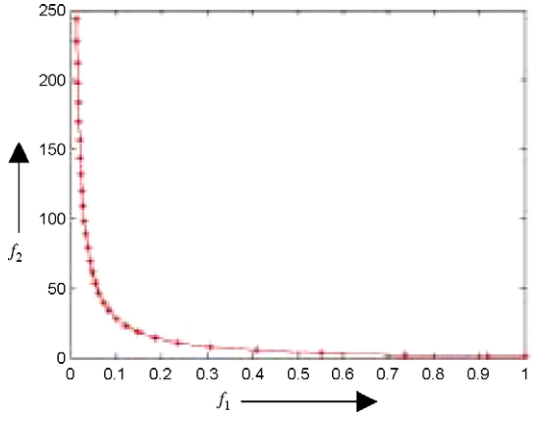

Fig. 4. Pareto-optimal front with Pareto-optimal set.

Diversity maintenance strategy is another characteristic of MOGAs. It works by keeping the solutions uniformly distributed in the Pareto-optimal set, instead of gathering solutions in a small region only. Fitness sharing [31], which reduces the fitness of an individual if there are some other candidates nearby, is one of the most renowned techniques. Restricted mating, where mating is permitted only when the distance between two parents are large enough, is another technique. More recently, some parameter free techniques were suggested. The techniques used in SPEA [26] and NSGA-II [32] are two examples of such techniques. PAES [30], SPEA [26], and NSGA-II [32] are representatives of current MOGAs. They all adopt Pareto-based fitness assignment strategy and implement elitism, an experimentally verified technique known to enhance performance. A good comprehensive study of MOGA can also found in Ghosh and Dehuri [33].

3.2. Mathematical formulation of rule generation and proposed method

Formally, the rule generation problem can be written as follows:

Maximize

$$f(\vec{x}) = (f_1(\vec{x}),\ f_2(\vec{x}),\ f_3(\vec{x})),$$

where

$$f_1(\vec{x}) = \varsigma(\Re) = 1 - \left(\frac{N_c(\Re)}{M_c}\right),$$

$$f_2(\vec{x})\, p(\Re) = \frac{(L_{CC} \times L_{\sim C \sim C})}{(L_{CC} + L_{C \sim C}) \times (L_{CC} + L_{\sim CC})},$$

and

$$f_3(\vec{x}) = \mathrm{RInt} = 1 - \frac{\sum_{i=1}^{n-1} \mathrm{InfoGain}(A_i)/n - 1}{\log_2(|\mathrm{dom}(G)|)}.$$

Hence to optimize these three objectives simultaneously using evolutionary approach, this paper provides a method called multi-objective genetic algorithm with a hybrid one-point uniform crossover. The pseudocode given here not only serves the task identified by us but also serves as a general framework for any kind of multi-criterion rule generation problem like association rule generation, fuzzy classification rule generation, dependency rule generation with a proper parameter setting.

Pseudocode

1. *g=1; External (g)=ϕ;*
2. *Initialize Population P(g);*
3. *Evaluate the P(g) by Objective Functions;*
4. *Assign Fitness to P(g) Using Rank Based on Pareto Dominance*
5. *External (g) ← Chromosomes Ranked as 1;*
6. ***While** (g <= Specified_no_of_Generation) **do***
7. *P'(g)← Selection by Roulette Wheel Selection Schemes P(g);*
8. *P"(g)← Single-Point Uniform Crossover and Mutation P'(g);*
9. *P"'(g)← Insert/Remove Operation P"(g);*
10. *P(g+1)← Replace (P(g), P"'(g));*
11. *Evaluate P(g+1) by Objective Functions;*
12. *Assign Fitness to P(g+1) Using Rank Based Pareto Dominance;*
13. *External (g+1) ← [External (g) + Chromosome Ranked as One of P(g+1)];*
14. *g=g+1;*
15. ***End while***
16. *Decode the Chromosomes Stored in External as an IF-THEN Rule.*

4. Experimental studies

4.1. Description of the dataset

The simulation was performed using the zoo, nursery and adult datasets obtained from the UCI machine repository (http://www.ics.uci.edu/). These datasets are normally used as a benchmark for evaluating algorithms performing classification task.

4.1.1. Zoo dataset

The zoo dataset contains 101 instances and 18 attributes. Each instance corresponds to an animal. In the preprocessing phase the attribute containing the name of the animal was removed, since this attribute has no generalization power. The attributes in the zoo dataset are all categorical. The attribute names in the dataset are as follows: hair (h), feathers (f), eggs (e), milk (m), predator (p), toothed (t), domestic (d), backbone (b), fins (fs), legs (l), tail (tl), catsize (c), airborne (a), aquatic (aq), breathes (br), venomous (v) and type (ty). Except type and legs, all other attributes are Boolean. The goal attributes are type 1–7. The type 1 has 41 records, type 2 has 20 records, type 3 has 5 records, types 4–7 have 13, 4, 8, 10 records, respectively.

4.1.2. Nursery dataset

This dataset has 12,960 records and nine attributes, all of them categorical. The ninth attributes is treated as class attribute and there are five classes: not_recom (NR), recommended (R), Very_recom (VR), Priority (P), and spec_prior SP). The attributes and corresponding values are listed in Table 1.

4.1.3. Adult dataset

The adult dataset contains 48,842 data instances, a mix of continuous and discrete. The number of attributes is 15. After a removal of incomplete records there are only 45,222 instances. It is divided into two sets, one set is for training and another one is for test having records 30,162 and 15,060, respectively. The classes are distributed are as follows. Probability for the label '>50 K': 23.93%/24.78% (without unknowns). Probability for the label '<=50 K': 76.07%/75.22% (without unknowns).

The classification rule-mining algorithm needs to discover rules by accessing the training set only. In order to do this, the algorithm has access to the values of both predicting attributes and the goal attribute of each example (record) in the training set.

Table 1
Description of nursery dataset in summary

Attribute	Values
Parents	Usual, pretentious, great_pret
Has_nurs	Proper, less_proper, improper, critical, very_crit
Form	Complete, completed, incomplete, foster
Children	1,2,3, more
Housing	Convenient, less_conv, critical
Finance	Convenient, inconv
Social	Nonprob, slightly_prob, problematic
Health	Recommended, priority, not_recom

Table 2
Parameters for both SGA and EMOGA

Dataset	P	P_c	P_m	External	R_m	R_I
Zoo	100	0.8	0.03	15	[0,0.7]	[0,0.8]
Nursery	500	0.75	0.002	50	[0.2, 0.8]	[0,0.6]
Adult	500	0.7	0.02	50	[0.2, 0.9]	[0,0.6]

P, population size; P_c, probability of crossover; P_m, probability of mutation; External, external population size; R_m, removal operator; R_I, insert operator.

Once the training process is finished and the algorithm finds a set of classification rules, the predictive performance of these rules is evaluated on the test set, which was not seen during training. Note that, once we take into account the large number of attributes, this can be considered a difficult classification problem.

4.2. Results and analysis

The experiments have been performed using MATLAB 6.5 on a Linux server. The data specific parameters and the parameters, which are encountered during the rule discovery, are listed in Table 2.

For each of the dataset, the simple genetic algorithm and EMOGA was run 500 generations. The parameters values such as P_c, P_m, R_m, and R_I given in Table 2 were sufficient to find some good individuals. The following computational protocols are used in the simple genetic algorithm as well as the proposed elitist multi-objective genetic algorithm for rule generation.

The dataset is divided into two parts: *training set* and *test set.* Here we have used a two fold cross validation of 2/3 for *training set* and 1/3 for *test set.* We represent the predicted class to all individuals of the population, which is never modified during the running of the algorithm. Hence, for each class we run the algorithms separately and collect the corresponding rules with an average performance metric in case of SGA and an average performance metric with respect to all the associated objectives in the case of an elitist MOGA. The generated rules of SGA and GA for multi-criterion rule generation have been compared and all rules are listed in the following table.

Tables 3 and 4 shows the results generated by SGA and EMOGA for multi-criterion rule discovery respectively from zoo dataset. The table has three columns namely class#, mined rules, fitness function of each rule. Table 3 presents the final seven rules discovered by the SGA and one rule for each class. For each class, the SGA was run five times, varying the random seed used to generate the initial population. The best rule from each run is collected; according to its fitness value measured in the training set predicting that class. Hence for each class, the corresponding rule with average performance metric is given in column number 2 and 3, respectively. Once the seven rules were selected, they were evaluated on a separate test set.

For each rule in Table 1 the third column shows the fitness value of the rule computed by Eq. (1). One can see that fitness value of the rules 1–4 discovered from the training set has a standard deviation of 0.02189 in contrast to the fitness value of the rules 5–7 discovered from the training set of standard deviation of 1.560. In other words, the fitness values of the rules 5–7 are closer enough than fitness values of the rules 1–4.

In Table 4, the *predictive accuracy, comprehensibility* and *interestingness* is given in columns 3, 4 and 5, respectively for each rule. The basic idea is that instead of transforming a multi-objective problem into a single objective problem and then solving it by using SGA, we use EMOGA to solve the original multi-objective problem. Intuitively this approach makes sense. Similar to SGA, for each class the EMOGA was run five times varying the random seed used to generate the initial population. For each run the set of non-dominated solutions are collected and the best compromising rules with the associated objective values are presented.

Table 3
Rules generated by SGA from zoo dataset

Class#	Mined rules	Fitness
1	*IF* (*hair* = *1*) ∧ (*eggs* = *0*) *A* (*venomous* = *0*) ∧ (*domestic* = *0*) *THEN* (*type* = *1*)	0.773625
2	*IF* (*hair* = *0*) ∧ (*feathers* = *1*) ∧ (*venomous* = *0*) ∧ (*legs* = *2*) ∧ (*domestic* = *0*) *THEN* (*type* = *2*)	0.765
3	*IF* (*eggs* = *1*) ∧ (*aquatic* = *0*) ∧ (*predator* = *1*) (*toothed* = 1) ∧ (*fins* = *0*) ∧ (*domestic* = *0*) ∧ (*catsize* = *0*) *THEN* (*type* = *3*)	0.721
4	*IF* (*aquatic* = *1*) ∧ (*breathes* = *0*) ∧ (*venomous* = *0*) ∧ (*tail* = *1*) *THEN* (*type* = *4*)	0.774
5	*IF* (*hair* = *0*) ∧ (*airbone* = *0*) ∧ (*aquatic* = *1*) ∧ (*toothed* = *1*) ∧ (*breathes* = *1*) ∧ (*legs* = *4*) *A* (*catsize* = *0*) *THEN* (*type* = *5*)	0.810
6	*IF* (*airbone* = 1) ∧ (*fins* = *0*) ∧ (*tail* = *0*) *THEN* (*type* = *6*)	0.8371
7	*IF* (*hair* = *0*) ∧ (*predator* = 1) ∧ (*breathes* = *0*) ∧ (*tail* = *0*) *A* (*domestic* = *0*) *THEN* (*type* = *7*)	0.814

Table 4
Rules generated by EMOGA from zoo dataset

Class #	Mined rules	$p(\Re)$	$\mathfrak{S}(\Re)$	RInt
1	*IF* (eggs = 0) ∧ (venomous = 0) ∧ (domestic = 0) *THEN* (type = l)	0.9090	0.8667	0.67
2	*IF* (*feathers* = *1*) ∧ (*breathes* = *1*) ∧ (*domestic* = *0*) *THEN* (*type* = *2*)	0.9333	0.8667	0.563
3	*IF* (*eggs* = *1*) ∧ (*predator* = *1*) ∧ (*toothed* = *1*) ∧ (*catsize* = *0*) *THEN* (*type* = *3*)	1.0	0.8	0.563
4	*IF* (*aquatic* = *1*) ∧ (*breathes* = *0*) ∧ (*tail* = 1) *THEN* (*type* = 4)	0.8	0.8667	0.823
5	7F (*airbone* = *0*) ∧ (*aquatic* = *1*) ∧ (*toothed* = *1*) ∧ (*breathes* = *1*) ∧ (*catsize* = *0*) *THEN* (*type* = *5*)	1.0	0.7333	0.81
6	*IF* (*airbone* = *1*) ∧ (*fins* = 0) ∧ (*tail* = 0) *THEN* (*type* = 6)	0.8333	0.8	0.856
7	IF (*predator* = *1*) ∧ (*breathes* = *0*) ∧ (*tail* = 0) ∧ (*domestic* = *0*) *THEN* (*type* = *7*)	0.875	0.8	0.911

Table 5
Rules generated by SGA from nursery dataset

Class	Mined rules	Fitness
P	*IF* (*parents* = *usual*) ∧ (*housing* = *less_conv*) ∧ (*social* = *problematic*) ∧ (*health* = *recommended*) *THEN* (*class* = *P*)	0.639
	IF (*parents* = *great_pret*) ∧ (*children* = *3*) ∧ (*social* = *slightly_prob*) ∧ (*health* = *recommended*) *THEN* (*class* = *P*)	0.5946
NR	*IF* (*parents* = *usual*) ∧ (*housing* = *less_conv*) ∧ (*social* = *slightly_prob*) ∧ (*health* = *not_recom*) *THEN* (*class* = *NR*)	0.567
	IF (*parents* = *pretentious*) ∧ (*children* = *3*) ∧ (*housing* = *convenient*) ∧ (*health* = *not_recom*) *THEN* (*class* = *NR*)	0.632
	IF (*parents* = *great_pret*) ∧ (*children* = *2*) ∧ (*housing* = *critical*) ∧ (*health* = *not_recom*) *THEN* (*class* = *NR*)	0.641
VR	*IF* (*parents* = *usual*) ∧ (*housing* = *less_conv*) ∧ (*finance* = *inconv*) ∧ (*social* = *slightly_prob*) ∧ (*health* = *recommended*) *THEN* (*class* = *VR*)	0.627
R	*IF* (*has_nurs* = *proper*) ∧ (*finance* = *convenient*) ∧ (*health* = *recomended*) *THEN* (*class* = *R*)	0.7175

Table 6
Rules generated by EMOGA from nursery dataset

Class#	Mined rules	$p(\Re)$	$\mathfrak{S}(\Re)$	RInt
P	*IF* (*parents* = *usual*) ∧ (*housing* = *less_conv*) ∧ (*social* = *problematic*) *THEN* (*class* = *P*)	0.7780	0.625	0.815
	IF (*parents* = *great_pret*) ∧ (*social* = *slightly_prob*) ∧ (*health* = *recommended*) *THEN* (*class* = *P*)	0.8114		
NR	*IF* (*parents* = *usual*) ∧ (*housing* = *less_conv*) ∧ (*social* = *slightly_prob*) *A*(*health* = *not_recom*) *THEN* (*class* = *NR*)	0.634	0.5	0.883
	IF (*parents* = *pretentious*) ∧ (*children* = *3*) ∧ (*housing* = *convenient*) ∧ (*health* = *not_recom*) *THEN* (*class* = *NR*)	0.7641		
	IF (*parents* = *great_pret*) ∧ (*children* = *2*) ∧ (*housing* = *critical*) ∧ (*health* = *not_recom*) *THEN* (*class* = *NR*)	0.783		
VR	*IF* (*housing* = *less_conv*) ∧ (*finance* = *inconv*) ∧ (*social* = *slightly_prob*) ∧ (*health* = *recommended*) *THEN* (*class* = *VR*)	0.897	0.5	0.761
R	*IF* (*has_nurs* = *proper*) ∧ (*finance* = *convenient*) ∧ (*health* = *recomended*) *THEN* (*class* = *R*)	0.81	0.625	0.781

Table 7
Rules generated by SGA from adult dataset

Class#	Mined rules	Fitness
>50 K	*IF* (*work_class* = *private*) ∧ (*Sex* = *Male*) ∧ (*Relation-ship* = *Husband*) *THEN* (*class* = < = *50 K*)	0.5146
<=50 K	*IF* (*work_class* = *private*) ∧ (*Marital_status* = *Married-civ-Spouse*) ∧ (*Relation-ship* = *Not-in-family*) *THEN* (*class* = < = *50 K*)	0.832

Tables 5 and 6 show the result generated by SGA and EMOGA for multi-criterion rule discovery, respectively, from nursery dataset. The table has five columns namely class#, mined rules, predictive accuracy, comprehensibility and interestingness. In Table 5, for class P and NR we have discovered two and three rules, respectively, because a single rule is unable to have a high expressive power. However, for class VR and R, we have got one rule each but with a good expressive power. With the same argument as in Table 4, EMOGA has discovered two rules for class P and three rules for class NR.

Similarly Tables 7 and 8 show the results generated by SGA and an elitist MOGA for rule discovery from adult dataset. With the same argument as in the case of zoo and nursery datasets, Tables 7 and 8 show results generate by SGA and an elitist MOGA (EMOGA) for rule discovery from adult dataset. In this case single rule is generated for each class both by using SGA and EMOGA with a good expressive power.

Further in Fig. 5 we have shown the performance metric of EMOGA obtained from three different dataset in the form of histogram. The performance metric corresponding to rules 6 and 7 shows that rule interestingness is no guarantee that the rule will be accurate and comprehensible.

Further, it is observed that in the case of zoo dataset after 100 generations it ceases to generate new rules. Similarly for the case of nursery dataset after 500 generations we are not getting any new rules with satisfactory performance. In the case of adult dataset after 500 generations though it ceases but the rules are different from each other. Moreover very few number of attributes got involved in the rules, which means that all the attributes are not equally important but the trends of rule comprehensibility is increasing. Except few, most of the rules are not as much interesting as we are expecting.

Table 8
Rules generated by EMOGA from adult dataset

Class#	Mined rules	$p(\Re)$	$\mathfrak{S}(\Re)$	RInt
>50 k	*IF* (*Sex* = *Male*) ∧ (*Relation-ship* = *Husband*) ∧ (*Native_country* = *United-State*) *THEN* (*class* = < = *50 k*)	0.5480	0.7857	0.615
<=50 k	*IF* (*work_class* = *private*) ∧ (*Marital_status* = *Married-civ-Spouse*) ∧ (*Relation-ship* = *Not-in-family*) ∧ (*Race* = *white*) *THEN* (*class* = < = *50 k*)	0.8621	0.7143	0.65

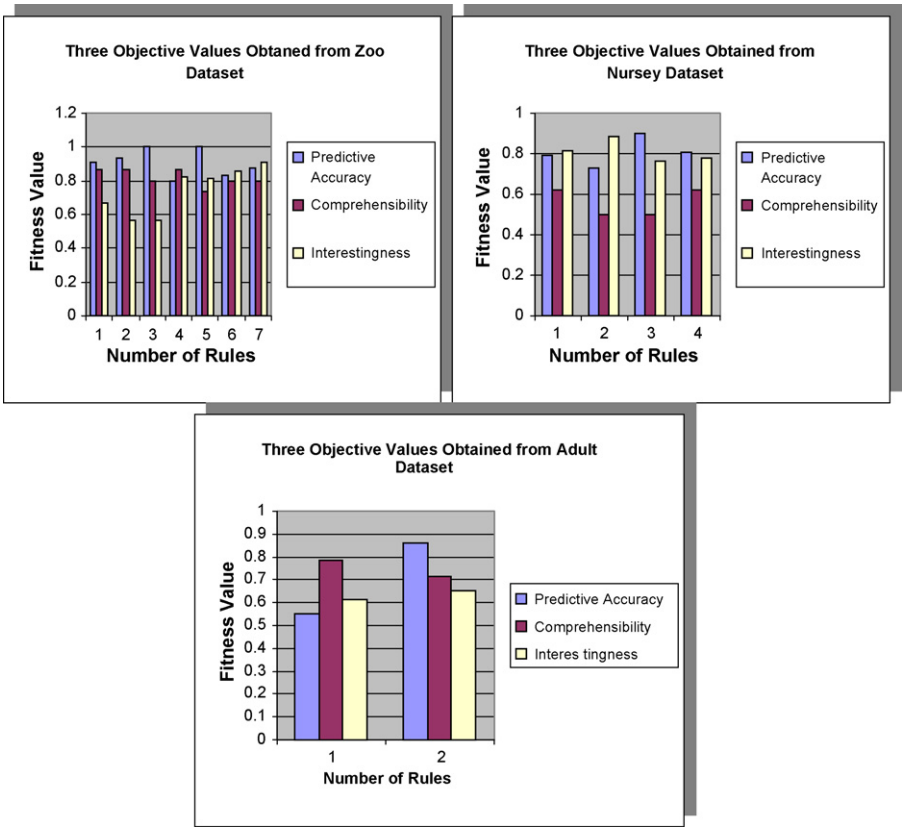

Fig. 5. Histogram representation of performance metric.

5. Conclusions and future research directions

In this article we have explored the use of an elitist multi-objective genetic algorithms for discovering *predictive, comprehensible* and *interestingness* rules. We have discussed the basic concepts and principles of application of SGA and the proposed method called EMOGA for classification rule generation and experimental results. Despite the small number of patterns available in our application domain the results can be considered to show the approximate trend. The comprehensibility of the discovered rules could, in principle, be improved with a proper modification of the fitness assignment method. We are now concentrating on careful selection of attributes [36] in a preprocessing step in order to reduce the number of attributes (and the corresponding search space) given to the EMOGA. Though there are few applications of EMOGA in data mining tasks [34,35], for validating its robustness and scalability more practical application to various domains of data mining and more studies are needed.

In addition, the complexity of the proposed algorithm seems to be low, because of the different runs at least once for each class (value of the goal attribute) and the continuous growth of the datasets. To make the algorithm more scalable, either it requires considering only a subset of the available data to evaluate the fitness of an individual or parallelizing EMOGA and then running on a clusters of PCs. Other important issues to be addressed include MOGA for fuzzy rule discovery and dealing with noisy, imprecise, and uncertain information. These issues provide the soft computing community a new dimension for further research.

References

[1] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, in: U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), From Data Mining to Knowledge Discovery: An Overview. Advances in Knowledge Discovery & Data Mining, AAAI/MIT, 1996, pp. 1–34.
[2] A.A. Freitas, A genetic programming framework for two data mining tasks: classification and generalized rule induction, in: Genetic Programming 1997, Proceedings of Second Annual Conference, Morgan Kaufmann, (1997), pp. 96–101.
[3] A.A. Freitas, On rule interestingness measures, Knowledge Based Syst. 12 (1999) 309–315.
[4] A.A. Freitas, A survey of evolutionary algorithms for data mining and knowledge discovery, in: A. Ghosh, S. Tsutsui (Eds.), Advances in Evolutionary Computing, Springer-Verlag, New York, 2003, pp. 819–845.
[5] S. Dehuri, R. Mall, Predictive and comprehensible rule discovery using a multi-objective genetic algorithm, Knowledge Based Syst. 19 (2006) 413–421.
[6] Soft computing approach to pattern recognition and image processing, A. Ghosh, S.K. Pal (Eds.), Series in Machine Perception and Artificial Intelligence, vol. 53, World Scientific Publishing, Singapore, 2002.
[7] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer-Verlag, New York, 2002.
[8] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.
[9] Z. Michalewicz, Genetic Algorithms + Data Structure = Evolution Programs, Springer-Verlag, Berlin, 1994.
[10] J.D. Schaffer. Some experiments in machine learning using vector evaluated genetic algorithms, Doctoral dissertation, Vanderbilt University, Nashville, TN, 1984.

[11] C.Z. Jainkow, A knowledge intensive genetic algorithm for supervised learning, Mach. Learn. 13 (1993) 189–228.
[12] M.V. Fedelis, et al., Discovering comprehensible classification rules with a genetic algorithm, in: Proceeding of Congress on Evolutionary Computation, 2000.
[13] W. Kwedlo, M. Kretowski, Discovery of decision rules from databases: an evolutionary approach, in: Proceedings of Second European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD-98). Lecturer Notes in Artificial Intelligence 1510, Springer-Verlag, 1998, pp. 371–378.
[14] M.J. Pazzani, Knowledge discovery from data? IEEE Intell. Syst. (2000) 10–12.
[15] A.A. Freitas, A genetic algorithm for generalized rule induction, in: Roy, et al. (Eds.), Advances in Soft Computing—Engineering Design and Manufacturing, Springer-Verlag, 1999, pp. 340–353.
[16] J.M. Cover, J.A. Thomas, Elements of Information Theory, John Wiley & Sons, 1991.
[17] G. Piatetsky-Shapiro, Discovery, analysis and presentation of strong rules, in: G. Piatetsky-Shapiro, W.J. Frawley (Eds.), Knowledge Discovery in Databases, AAAI Press, 1991, pp. 229–248.
[18] D.J. Hand, Construction and Assessment of Classification Rules, John Wiley & Sons, 1997.
[19] B. Liu, W. Hsu, S. Chen, Using general impressions to analyze discovered classification rules, in: Proceedings of Third International Conference on Knowledge Discovery & Data Mining (KDD-97), AAAI Press, 1997, pp. 31–36.
[20] E. Noda, A.A. Freitas, H.S. Lopes, Discovering interesting prediction rules with a genetic algorithm, in: Proceedings of Conference on Evolutionary Computation 1999 (CEC-99), Washington, DC, USA, (1999), pp. 1322–1329.
[21] A.A. Freitas, On objective measures of rule surprisingness, in: Proceedings of Second European Symposium on Principle of Data Mining and Knowledge Discovery (PKDD-98), Lecturer Notes in Artificial Intelligence, 1510, 1998, pp. 1–9.
[22] E. Falkenauer, The worth of the uniform, in: Proceedings of the Congress on Evolutionary Computation (CEC'99), IEEE, 1999, pp. 776–782.
[23] G. Syswerda, Uniform crossover in genetic algorithms, in: Proceedings of Third International Conference on Genetic Algorithms (ICGA 89), 1989, pp. 2–9.
[24] C.L. Hwang, K. Yoon, Multiple Attribute Decision Making, Methods and Application, A State of Art Survey, Springer-Verlag, New York, 1981.
[25] M. Zeleny, Multiple Criteria Decision Making, McGraw-Hill, New York, 1982.
[26] E. Ziztler, L. Thiele, Multi-objective evolutionary algorithms: a comparative case study and strength Pareto approach, IEEE Trans. Evol. Comput. 3 (1999) 257–271.
[27] J. Horn, N. Nafpliotis, E. Goldberg, A niched Pareto genetic algorithm for multi-objective optimization, in: Proceeding of the first IEEE Conf. on Evolutionary Computation, IEEE World Congress on Computational Intelligence, vol. 1, 1994, 82–87.
[28] C.A. Coello Coello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary algorithms for solving multi-objective problems, Kluwer Academic Publishers, New York, 2002.
[29] K. Deb, Multi-objective optimization using evolutionary algorithms, Wiley, New York, 2001.
[30] M. Laumanns, G. Rudolph, H.P. Schwefel, A spatial predator–prey approach to multi-objective optimization, Parallel Prob. Solv. Nat. 5 (1998) 241–249.
[31] D.E. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: Proceedings of Second International Conference on Genetic Algorithm, 1987, pp. 41–49.
[32] K. Deb, A. Pratap, S. Agrawal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2002) 182–197.
[33] A. Ghosh, S. Dehuri, Evolutionary algorithms for multi-criterion optimisation: a survey, Int. J. Comput. Inform. Sci. 2 (2004) 38–57.
[34] S. Bhattacharya, Evolutionary algorithms in data mining: multi-objective performance modeling for direct marketing, in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'2000), ACM, 2000, pp. 465–473.
[35] S. Bhattacharya, Multi-objective data mining using genetic algorithms, in: A.S. Wu (Ed.), Proceeding of the 2000 Genetic and Evolutionary Computation Conference Workshop Program—Workshop on Data Mining with Evolutionary Algorithms, 2000, 76–79.
[36] J.R. Cano, F. Herrera, M. Lozano, On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining, Appl. Soft Comput. 6 (2006) 323–332.