On the Use of Genetic Programming for Mining Comprehensible Rules in Subgroup Discovery

José María Luna, *Student Member, IEEE*, José Raúl Romero, *Member, IEEE*, Cristóbal Romero, *Member, IEEE*, and Sebastián Ventura, *Senior Member, IEEE*

***Abstract*—This paper proposes a novel grammar-guided genetic programming algorithm for subgroup discovery. This algorithm, called CGBA-SD (Comprehensible Grammar-Based Algorithm for Subgroup Discovery), combines the requirements of discovering comprehensible rules with the ability to mine expressive and flexible solutions thanks to the use of a context-free grammar. Each rule is represented as a derivation tree that shows a solution described using the language denoted by the grammar. Unlike other existing evolutionary algorithms for subgroup discovery, the proposed algorithm only requires a small number of parameters, thereby providing users with an easy way to discover subgroups. The algorithm includes mechanisms to adapt the diversity of the population by self-adapting the probabilities of recombination and mutation. We compare the approach with existing evolutionary and classic subgroup discovery algorithms. CGBA-SD appears to be a very promising algorithm that discovers comprehensible subgroups and behaves better than other algorithms as measures by complexity, interest, and precision indicate. The results obtained were validated by means of a series of non-parametric tests.**

***Index Terms*—genetic programming (GP), grammar-guided genetic programming (G3P), subgroup discovery, data mining (DM)**

I. Introduction

THE aim of data mining (DM) is to extract non-trivial information and knowledge hidden in data. DM is broken down into two main categories, unsupervised [1] and supervised [2] tasks. Unsupervised tasks include approaches that explore the data to find some intrinsic structures in them, so these tasks have a descriptive nature [3]. Regarding the supervised tasks and given a sequence of expected outputs, the goal is to predict [4] the correct output for a new input. This output could be a class label (in classification) or a real number (in regression). Nowadays, there exist new tasks at the intersection of prediction and description, including subgroup discovery (SD) [5], contrast set mining [6], and emerging patterns mining [7]. These techniques are known as supervised descriptive rule induction (SDRI) [3].

SD was first described by *Klösgen* [8] and *Wrobel* [9] as follows: "Given a population of individuals (customers, objects, etc.) and a property of those individuals that we are interested in, the task of SD is to find population subgroups that are statistically most interesting for the user, e.g., subgroups that are as large as possible and have the most unusual statistical characteristics with respect to a target attribute of interest". In such a way, SD combines the features of supervised and unsupervised learning tasks and uncovers explicit subgroups via individual rules, which must be easily understandable by users. Any SD algorithm should be able to discover interesting subgroups by means of simple and comprehensible rules, i.e., rules having a clear structure and few variables or attributes [5]. Those rules are also required to be of highly interest and cover as many instances of the class attribute as they can.

Early SD approaches were based on deterministic models, consisting in adapting either association rule mining or classification algorithms for the SD task [10]. Then, some authors [11], [12] developed evolutionary algorithms that permitted the use of numerical features without the need for a previous discretisation, which was a major problem in deterministic models. Nevertheless, a drawback of current evolutionary algorithms is that they are mainly based on fuzzy systems, requiring a number of predefined linguistic labels. Besides, the number of rules and variables discovered by existing algorithms is rather high, hampering the interpretability of the resulting model.

Taking these points into account, and considering that the descriptive induction perspective has already obtained promising results [13] by using grammar-guided genetic programming [14], [15] (G3P), we propose a G3P approach for mining subgroups of statistical interest to the user [16]. First, the use of a grammar enables solutions that have an expressive and flexible structure. This interesting feature could be shared with SD, where a major issue for the user is the ability to interpret the extracted knowledge. Second, the proposed algorithm, called CGBA-SD (Comprehensible Grammar-Based Algorithm for Subgroup Discovery), represents numerical features without any knowledge about the number of linguistic labels as in fuzzy systems. Finally, the aim of this algorithm is the discovery of reliable and highly representative subgroups, so the number of rules and variables is not as high as in existing evolutionary proposals.

A detailed experimental study was carried out in order to demonstrate the effectiveness of CGBA-SD. In the analysis a series of evolutionary SD algorithms were compared, including NMEEF-SD [17], SDIGA [11] and MESDIF [12]. Additionally, classic SD algorithms, such as CN2-SD [18] and Apriori-SD [10], were also included in the study. In this analysis, a number of nonparametric tests were performed, demonstrating the effectiveness of the proposed approach and its ability to discover subgroups that include few variables.

The authors belong to the Department of Computer Science and Numerical Analysis, University of Cordoba, 14071 Cordoba, Spain (e-mail: {jmluna,jrromero,cromero,sventura}@uco.es.

Digital Object Identifier ?

This paper is structured as follows: a description of SD is included and the most relevant related work is presented in Section II; Section III describes the model proposed as well as its main characteristics; Section IV describes the experiments, including the datasets used, the algorithm set-up, and discusses the results obtained; finally, in Section V, some concluding remarks are outlined.

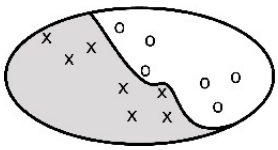
II. Preliminaries

In the following subsections, a brief description of subgroup discovery, an analysis of the quality measures used in the subgroup discovery task, and some related work are given.

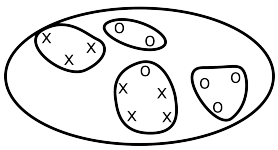
A. The Subgroup Discovery Task

SD is a technique included in SDRI (Supervised Descriptive Rule Induction) as a way of seeking relations among different variable values with respect to a class attribute. SD lies halfway between supervised and unsupervised tasks. In unsupervised tasks, the aim is to describe hidden structures by exploring unlabeled data. On the contrary, the goal of supervised learning is to obtain a classifier that predicts the correct output given an input, and this classifier should be as accurate as possible. The main difference with respect to classification tasks is that SD seeks to extract interesting subgroups (not an accurate classifier) for a given class attribute and the subgroups discovered do not cover all the examples for the class. SD is similar to a clustering model where the data is labeled. Unlike in clustering, where the goal is to describe unlabeled data structure, SD is conceived to form subgroups that behave similarly, so new input data could be classified according to these subgroups. Figure 1 shows the difference between the classification, clustering and SD tasks.

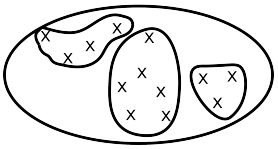
Similarly to the classification task, where classifiers may be opaque (neural networks, support vector machines) or transparent (decision tress and rule-based classifiers), it should be noted that the goal of the SD task is to obtain simple and interpretable models (few variables, and clear and easy to understand structure). Finally, it should be recalled that in classification the emphasis concentrates on quality measures concerned with the classifier, whereas in SD tasks the quality measures are related to each individual subgroup.



(a) Example of a classification model



(b) Example of a SD model



(c) Example of a clustering model

Fig. 1. Difference between classification, clustering and SD tasks

In SD, any subgroup is represented through independent rules of the type *IF Antecedent THEN Class*. The left side of a rule or *Antecedent* in SD comprises a conjunction of conditions (attribute-value pairs), which enables the distribution of the subgroup to be described. On the other hand, the right side or *Class* specifies a value of the attribute of interest.

B. Quality Measures in Subgroup Discovery

The choice of quality measures in SD has been discussed by many authors [19], leading to a wide variety of measures. Despite all these studies, no clear consensus exists currently about which specific quality measures are best for the SD task. Many authors classify quality measures as measures of complexity, generality, precision and interest.

Herrera et al. [5] described that there are some quality measures that are related to the interpretability of the subgroups. These measures, which are considered as complexity measures, describe the simplicity of the knowledge extracted from the discovered subgroups. Two measures in this regard are the number of rules and the number of variables in the antecedent.

As for the generality measures, i.e., those that quantify the quality of the subgroups according to the patterns covered, the most commonly used measure is support, also known as support on the basis of examples of the class. This measure is formally described in Equation 1 as the fraction of retrieved instances T that are relevant, i.e., the percentage of instances from the dataset D that satisfy the antecedent $Antc$ and the class of the rule on the basis of examples of the class.

$$support(R) = \frac{|\{Antc \cup Class \subseteq T, T \in D\}|}{|\{Class \subseteq T, T \in D\}|} \tag{1}$$

One of the most commonly used precision measures in SD is the confidence of the rule. This measure determines the reliability of a subgroup, measuring the relative frequency of examples that satisfy the complete rule among those satisfying only the antecedent (see Equation 2).

$$confidence(R) = \frac{|\{Antc \cup Class \subseteq T, T \in D\}|}{|\{Antc \subseteq T, T \in D\}|} \tag{2}$$

As far as interest measures are concerned, a commonly used one in SD is significance, which indicates the significance of a finding measured by the likelihood ratio of a rule, where n_c is computed as the number of classes in Equation 3.

$$significance(R) = 2 \times \sum_{i=1}^{n_c} (|\{Antc \cup Class_i \subseteq T, T \in D\}|) \times \log \frac{|\{Antc \cup Class_i \subseteq T, T \in D\}|}{|\{Antc \subseteq T, T \in D\}| \times \frac{|\{Class_i \subseteq T, T \in D\}|}{|D|}} \tag{3}$$

Finally, it is also possible to find quality measures that attempt to obtain a trade-off between generality, interest and precision. One example of that kind of measures is the unusualness of a rule, defined as the weighted relative accuracy and computed as depicted in Equation 4.

$$unusualness(R) = \frac{|\{Antc \subseteq T, T \in D\}|}{|D|} \times \left(\frac{|\{Antc \cup Class \subseteq T, T \in D\}|}{|\{Antc \subseteq T, T \in D\}|} - \frac{|\{Class \subseteq T, T \in D\}|}{|D|} \right) \quad (4)$$

C. Related Works

Since the concept of SD was introduced by *Klösgen* [8] and *Wrobel* [9], SD has been widely studied by many researchers and a number of algorithms have been proposed [5]. First approaches are extensions of classification algorithms, like Apriori-SD [10], which was developed by adapting the classification algorithm Apriori-C [20], a modification of the well-known Apriori algorithm for mining association rules [21]. This adaptation to the SD task includes a new post-processing mechanism, a new quality measure for the rules mined, and a probabilistic classification of the examples.

CN2-SD [18] is another example of an algorithm based on classification algorithms and considered to be an adaptation of the CN2 classification algorithm [22]. CN2 works in an iterative fashion, searching for a conjunction of attributes that behaves well according to a heuristic measure, which is based on the information-theoretic entropy measure. This function prefers rules covering a large number of examples of a single class and few of other classes. Then, the algorithm removes those examples the extracted rule covers and adds the rule to a list of rules. The process iterates until no more satisfactory rules can be found. CN2-SD is different from the original version in the following points: (a) a new heuristic is included which combines generality and accuracy; (b) it incorporates example weights into the covering step; and (c) it uses a probabilistic classification based on the class distribution of the examples covered.

Since SD could be approached and solved as an optimization and search problem some authors have treated the SD problem from an evolutionary perspective. Existing evolutionary algorithms in SD are based on a "chromosome = rule" approach and fuzzy logic [23], where each individual or solution to the problem codifies a single rule and the whole resulting set is provided by a combination of several individuals. Furthermore, solutions are encoded as bit chains and comprising fuzzy linguistic labels that represent the values of the continuous variables. One of these algorithms is SDIGA [11], an evolutionary model for mining fuzzy rules for the SD task. SDIGA has to be run once for each value in the target feature or class. Therefore, each run of this algorithm mines a set of subgroups for a specific class value.

MESDIF [12] is a genetic algorithm based on a multi-objective methodology for mining fuzzy rules representing subgroups. The search process is inspired by the multi-objective SPEA2 [24] approach, and it uses several quality measures (confidence, support, significance and unusualness) at the same time to evaluate the quality of the subgroups. In multi-objective optimization, there is no single best solution to the problem, i.e., a solution better than the rest with respect to each objective. Typical multi-objective optimization determines a set of solutions that are superior to the set when all the objectives are considered at time. Solutions of a specific iteration are organized in fronts [25] based on the objectives to be optimized. Thus, solutions from the first front are better than solutions from the second front, and so on. As for the solution from a specific front, none is better than the other solutions in the same front for all the objectives, so all of them are equally acceptable. Thus, in the SPEA2 algorithm, solutions from the same front are ranked according to a fitness value obtained from two measures: (a) the number of solutions dominated by a solution; (b) the Cartesian distance from their k-th nearest neighbours in the population.

Finally, a novel evolutionary fuzzy system, whose objective is to extract either fuzzy or crisp rules for the SD task, was proposed by *Carmona et al.* [17]. Their algorithm, named NMEEF-SD, is based on a multi-objective approach to optimize a set of quality measures at time (confidence, support, sensitivity, significance and unusualness). The search strategy of NMEEF-SD is similar to that of NSGA-II [26], which returns a predefined number of optimal solutions previously organized in fronts. In order to carry out this organization in fronts, NSGA-II determines the density of solutions surrounding a particular solution. Thereafter, for each objective function, the boundary solutions (solutions with the smallest and largest function values) are assigned an infinite distance value. All other intermediate solutions are assigned a distance value equal to the absolute normalized difference in the function values of two adjacent solutions. The overall distance value is calculated as the sum of the distance values for each objective.

Classical or not evolutionary algorithms developed so far are very time consuming and require a high memory with the increment of data size [5]. Besides, a major problem of this type of algorithms is their inability to be applied on numerical domains, requiring a previous discretisation step to transform numerical features into a set of discrete values. In order to address these issues, evolutionary approaches were proposed, which are mainly based on fuzzy systems. These approaches (SDIGA [11], NMEEF-SD [17], MESDIF [12]) formerly require a fixed number of linguistic labels, so previous knowledge about the domain under study is mandatory

D. Genetic Programming

Genetic programming (GP) [27] is an evolutionary and very flexible heuristic technique that enables complex pattern representations to be used. In GP, solutions are represented by means of trees, which enable any kind of function and domain of knowledge to be considered. Each solution represented as a tree structure includes two type of symbols: (1) leave nodes correspond to variables and constants; (2) internal nodes correspond to operators and functions. Figure 2 shows a sample GP tree structure (the genotype) that represents the function 2 * (x + 1) (the phenotype).

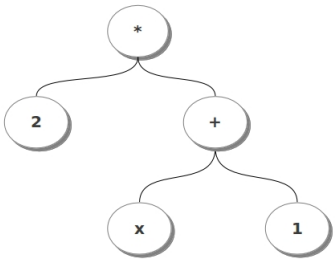


Fig. 2. Example of GP tree structure that represents the function 2*(x+1)

GP [28] was proposed as a way to find an optimized solution by using a complex representation language. Solutions were firstly represented as computer programs and currently, GP is used to evolve other types of knowledge, like rule-based systems [29]. Recently, a new extension of GP, which enables the problem constraints to be formally defined, is being of interest [30], [31]. This new methodology, named grammar-guided genetic programming (GGGP or G3P) [14], makes use of a context-free grammar that generates any feasible solution to the problem under study regardless the type of solutions to be obtained. In this way, the grammar constrains, the search space and the solutions are constructed by applying a set of productions rules.

III. A Grammar-Guided Genetic Programming Algorithm

In this section, the CGBA-SD algorithm is described in depth. This algorithm combines the strength of optimizing by means of evolutionary algorithms with the ability of representing rules in an expressive and flexible way thanks to G3P. Representing the solutions by using a context-free grammar allows for the main requirements for the SD task to be achieved, i.e., the interpretability of the extracted knowledge, the clear and flexible structure of the rules mined, and the ability to represent solutions in any domain, either nominal or numerical.

A. Encoding criterion

As usual in G3P, the approach proposed here represents the individuals by genotype and phenotype. Whist the genotype in genetic algorithms is presented with a fixed structure (in the form of a chain of bits or characters) conveying expectations about the form of a solution, in G3P it is defined by means of a tree structure with different shapes and sizes conformant to a context-free grammar. On the other hand, the phenotype enables the meaning of either the tree structure or the chain of bits to be obtained.

A context-free grammar could be formally defined as a four-tuple (Σ_N, Σ_T, P, S), in which Σ_N is the non-terminal symbol alphabet, Σ_T denotes the terminal symbol alphabet, P stands for the set of production rules, S for the start symbol, and Σ_N and Σ_T are disjoint sets, i.e., $\Sigma_N \cap \Sigma_T = \emptyset$. Any production rule follows the format $\alpha \rightarrow \beta$ where $\alpha \in \Sigma_N$, and $\beta \in \{\Sigma_T \cup \Sigma_N\}^*$. Beginning from the start symbol S, each individual is represented in a derivation syntax-tree as a sentence conformant to the grammar. The benefits of using grammars are the ability to define syntax constraints, providing expressiveness, flexibility, and the ability to restrict the search space. To obtain individuals, a number of production rules are applied from the set P. The maximum number of production rules is properly predefined to control bloat, so there is a maximum tree size enforced.

Once a grammar G is defined to describe valid expressions and to impose restrictions, a validation of this grammar is required. Thus, considering the grammar defined in our problem and depicted in Figure 3, the following language is obtained: $L(G) = \{(AND\ \ Condition)^n\ Condition\ \rightarrow\ Class : n \geq 0\}$. Therefore, using the aforementioned language, the grammar is well-defined and structured since any rule having at least one condition in the antecedent is obtained. Using this grammar it is possible to mine any subgroup containing either numerical or nominal features, i.e., an important feature of using a grammar is its ability to be adapted to each specific application domain or problem. In a sample rule, a categorical attribute accepts one value in a discrete unordered domain D. Given an attribute x and the values $y_1, y_2, ..., y_n$ in the domain D of x, the following expression is valid: $x = y_n$, indicating that x takes the value y_n in D. Furthermore, numerical attributes are used by applying the operator IN, and randomly selecting two feasible values.

For the sake of clarifying individual representation, it is interesting to show the genotype of sample individual generated through the application of eight production rules from P (see Figure 4). The terminal nodes were obtained from the metadata of a sample dataset listed in Table I, and the values were randomly selected from the set of feasible values for each attribute. As for the numerical attributes, the minimum and maximum bounded values were randomly obtained from

G = (Σ_N, Σ_T, P, S) with:

S = Subgroup
Σ_N = {Subgroup, Antecedent, Class, Nominal_Condition, Numerical_Condition }
Σ_T = {'AND', 'Attribute', 'Class_value', '=', 'IN','Min_value', 'Max_value', 'Target_attribute' }
P = {Subgroup ::= Antecedent, Class ;
Antecedent ::= Nominal_Condition | Numerical_Condition | 'AND', Nominal_Condition, Antecedent |
'AND', Numerical_Condition, Antecedent ;
Nominal_Condition ::= 'Attribute', '=', 'value' ;
Numerical_Condition ::= 'Attribute', 'IN', 'Min_value', 'Max_value' ;
Class ::= 'class', '=', 'Class_value'; }

Fig. 3. Context-free grammar expressed in EBNF notation

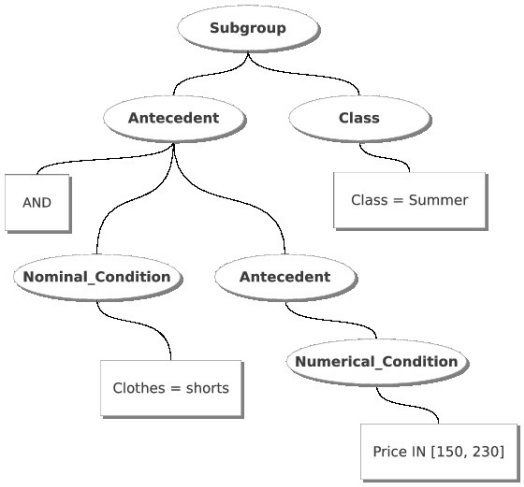


Fig. 4. Sample tree structure (genotype) conformant to the grammar defined in CGBA-SD

TABLE I
SAMPLE DATASET

Attribute	Type	Values
Clothes	Nominal	shirt, shorts, jeans
Color	Nominal	Red, Blue, Green, Black
Price	Numerical	[100, 900]
Class	Nominal	Summer, Winter

the whole range width, i.e., two values from 100 to 900 for the $Price$ attribute. Notice that any value could be obtained, so it is possible to select the whole range, the fitness function determining whether the resulting individual is promising or not.

As mentioned above, the phenotype of an individual represents the meaning of the tree structure. Therefore, focusing on the sample genotype obtained from the sample dataset, the phenotype is the representation of the following rule:

IF $Clothes = shorts \ \wedge \ Price\ IN\ [150, 230]$
THEN $Class = summer$

B. Evaluation of Individuals

In any evolutionary model, the evaluation process is a core task, allowing a fitness value to be assigned to each individual in order to determine how promising a certain individual is, i.e., how close a given solution is to achieve the aim.

In the SD task, the right choice of quality measures is no trivial issue. This problem has been studied by many authors [5], leading to a wide variety of quality measures that could be classified as measures of complexity, generality, precision or interest. Therefore, the evaluation process should be carefully studied to optimize as many quality measures as possible, so the CGBA-SD algorithm does not focus on just one specific quality measure but on many of them. The objective is to mine reliable subgroups and cover a high percentage of examples of each class by using the fitness function (see Equation 5). In such a way, a higher fitness function value enables measures of generality (considering the support based on examples of the class as shown in Equation 1) and precision (see Equation 2) to be optimized. Additionally, the mere fact of searching for highly frequent subgroups implies optimization of measures of complexity, especially in the number of variables in the antecedent of the rule. Note that the higher the length of the rule, the lower its support is, i.e., the more specific the rule is, the higher its complexity.

$$fitness(R) = support(R) \times confidence(R) \tag{5}$$

In order to obtain rules having a high fitness function value, CGBA-SD searches for those having a high value for $|\{Antc \cup Class_i \subseteq T, T \in D\}|$, and also having a low value for $|\{Antc \subseteq T, T \in D\}|$. Therefore, the algorithm is indirectly optimizing two important interest measure, i.e., the significance and the unusualness measures, which were properly defined in Equations 3 and 4.

C. Initialization of the Population

This initial iterative procedure (see Figure 5) allows for individuals to be created by using the context-free grammar defined in Figure 3 and the metadata extracted from the dataset. The main objective of this procedure is to enable the evolutionary process to be started with feasible individuals. To do so, this initial procedure produces a desired number of individual and evaluates them to determine how close these solutions are to the aim. If a specific individual has a fitness function value equal to 0, i.e., an invalid individual having a support value or a confidence value of 0, then a new individual is required to be created in the next iteration, and the invalid one is removed. Whenever the created individual has a value over 0, this individual is kept in the population. In the next iteration, the number of individuals to be created is determined by the number of invalid individuals in the previous iteration. Thus, in each iteration, the number of individuals to be created is lower. The procedure continues until 50 feasible individuals — the optimal parameter obtained from a set of previous studies as properly described in Section IV — are obtained,

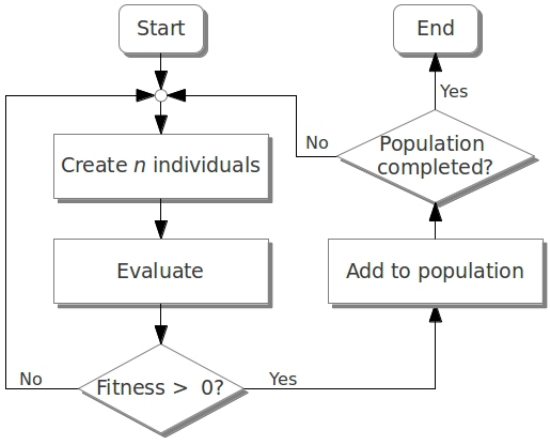


Fig. 5. Procedure for initiating population

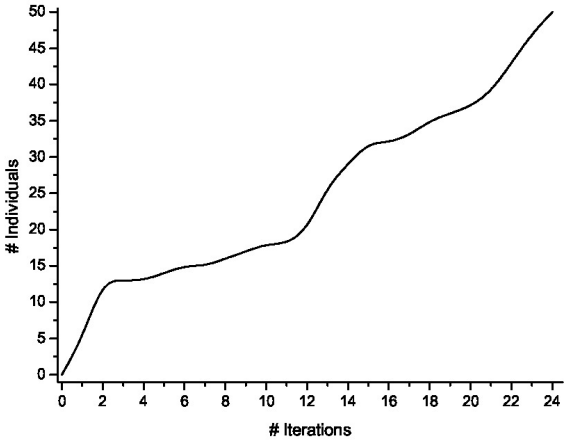


Fig. 6. Number of iterations required to discover 50 feasible individuals

making the evolutionary process better since it starts with a population of diverse but feasible individuals.

For the sake of demonstrating the number of individuals lost during the initialization, we have carried out a series of experiments in this sense, analysing whether the proposed procedure is a high runtime task. In average, the results have revealed that half of the individuals are feasible in the early iterations, and no further iterations are required to achieve the aim. Figure 6 illustrates a sample dataset that requires 50 individuals in its population, achieving this number of feasible individuals in 24 iterations.

D. Application of Genetic Operators

Like any evolutionary algorithm, CGBA-SD has the goal of solving the computational problem by improving the fitness function values throughout the evolutionary process. Therefore, in each generation, new individuals with better fitness values are desired. In this section, two genetic operators[1] specifically designed for this algorithm are presented. The main feature of these operators is that they do not need any previously fixed probability to determine whether the genetic operator is applied or not. These probabilities are adjusted on the basis of population requirements, i.e., it depends on the diversity of the population, as described below.

a) Crossover: In this genetic operator, the goal is to obtain a new individual comprising conditions with a higher frequency of occurrence than the original ones. In such a way, the worst condition (the one that covers the lowest number of instances) of a parent, previously selected by means of a tournament of size 2 selection method, is replaced with the best condition (the one that covers the highest number of instances) of another parent selected in the same way. This genetic operator enables any condition except class to be selected. The proposed procedure chooses two cut-points depending on the attribute selected as the best one or the worst one. Note that the shape of the tree structure or the domain of the condition to be replaced are not of relevance — any condition could be replaced by any other condition in any individual.

b) Mutator: The principal aim of this genetic operator is to modify the genotype of an individual in order to attempt to discover a new one with a better fitness. To do so, the procedure selects from a parent the condition that covers the lowest number of instances and changes that condition in order to obtain a new one that improves the fitness value. The genetic operator modifies the selected condition (Numerical_Condition or Nominal_Condition) and randomly acts in one of the following ways: (a) the value (or values in case of numerical attributes) of the condition is modified and the attribute name remain the same; (b) the condition is fully modified and replaced by another attribute and value of the same type; (c) there is no restriction in the replacement, i.e., a numerical condition could be replaced by a nominal condition.

In evolutionary computation, many parameters are used in a variety of algorithms. Each of these parameters are often adjusted by hand, requiring a parallel tuning since some of the parameters may be related. Sometimes, it is interesting to avoid the use of static parameters, which may contradict the dynamic nature of evolutionary algorithms. For instance, a high probable value for mutation in a specific generation expand the search space, and a low probable value enables a more accurate solution to be obtained.

A major feature of CGBA-SD is its ability to automatically change the probability values used by the genetic operators. A probability value is responsible for determining when a specific operator is applied on an individual in a specific generation. CGBA-SD comprises two different probabilities, one for mutation and the other for recombination. These two probabilities maintain a relation of p and $1-p$, i.e., the sum of both probability values is 1.

In the initial generation, there is no information about how the genetic operators should work, so the parameter values are initialized in a medium rate, i.e., 0.5. Once a new generation is completed, it checks whether the average fitness value of the elite population has increased or decreased with regard to same value obtained in the previous generation. If that value is greater than the one calculated in the previous generation, then the crossover probability is increased in 0.02 (and the mutation probability is decreased in the same range to maintain the $1-p$ relation) since a depth search or exploitation is required. On the contrary, if the average fitness value is less than or equal to the one obtained in the previous generation, then the crossover probability is decreased in 0.02 (and the mutation probability is increased in the same range), since it is necessary to include diversity in the population.

For the purpose of illustrating how this updating process behaves, Figure 7 lists the crossover and mutation probability values in each generation for a sample execution using the well-known *Iris* dataset. Note that in the first generation, both measures have the same probability value. It is interesting to note that in the earliest generations, both probabilities increase and decrease while the optimal value is not found. Thereafter, the mutation probability begins to increase whereas the recombination probability decreases. This means that the

[1]Further information about how these operators work is publicly available at http://www.uco.es/grupos/kdis/kdiswiki/G3P-SD

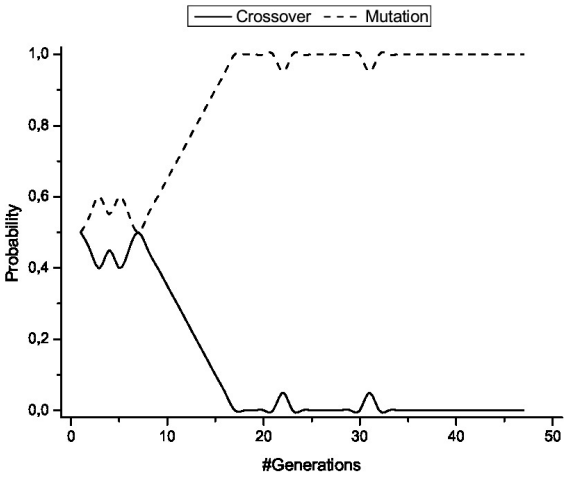


Fig. 7. Crossover and mutation probabilities along the generations

average fitness value is not improving and, therefore, the optimal value is obtained. Remember that the goal of this figure is only to show the behaviour of these probabilities, so a simple dataset was selected where the algorithm converges to the optimal solution in early generations.

E. Iterative Evolutionary Model

In previous subsections, the evaluation process, the initial procedure to obtain individuals conformant to a context-free grammar and the genetic operators used were described. Now, it is necessary to fully describe the complete evolutionary model and discuss how the different procedures described are combined to prompt a promising evolutionary algorithm for mining subgroup of interest according to SD quality measures.

CGBA-SD follows an iterative methodology, mining subgroups of a specific class in each iteration. In such a way, its aim is to discover the best subgroups (induced as rules in the form of a syntax-tree) for each class. For better understanding, Figure 8 illustrates the iterative model graphically. The algorithm runs a complete evolution of the rules for a specific class iteratively, i.e., the individuals evolve over a number of generations in order to obtain the best ones. Once this generational procedure is finished, a new set of individuals satisfying a new class is created. This iterative process is repeated as many times as the number of distinct class values are available in the dataset.

In each generation of CGBA-SD, those rules (subgroups) exceeding a minimum confidence value previously specified are selected to be included in a external population. This selection procedure works as an elitist selection, allowing some of the best individuals discovered from the current generation to be carried over unaltered to the next generation. Selecting the best individuals from a set is not a trivial procedure since it is necessary not only to select the best ones but also to select the most representative ones. In such a way, individuals are compared by using Equation 6, which considers that two individuals are equivalent if they cover similar instances.

$$comparator(V_1, V_2) = \frac{V_1 \cdot V_2}{|V_1| \cdot |V_2|} \quad (6)$$

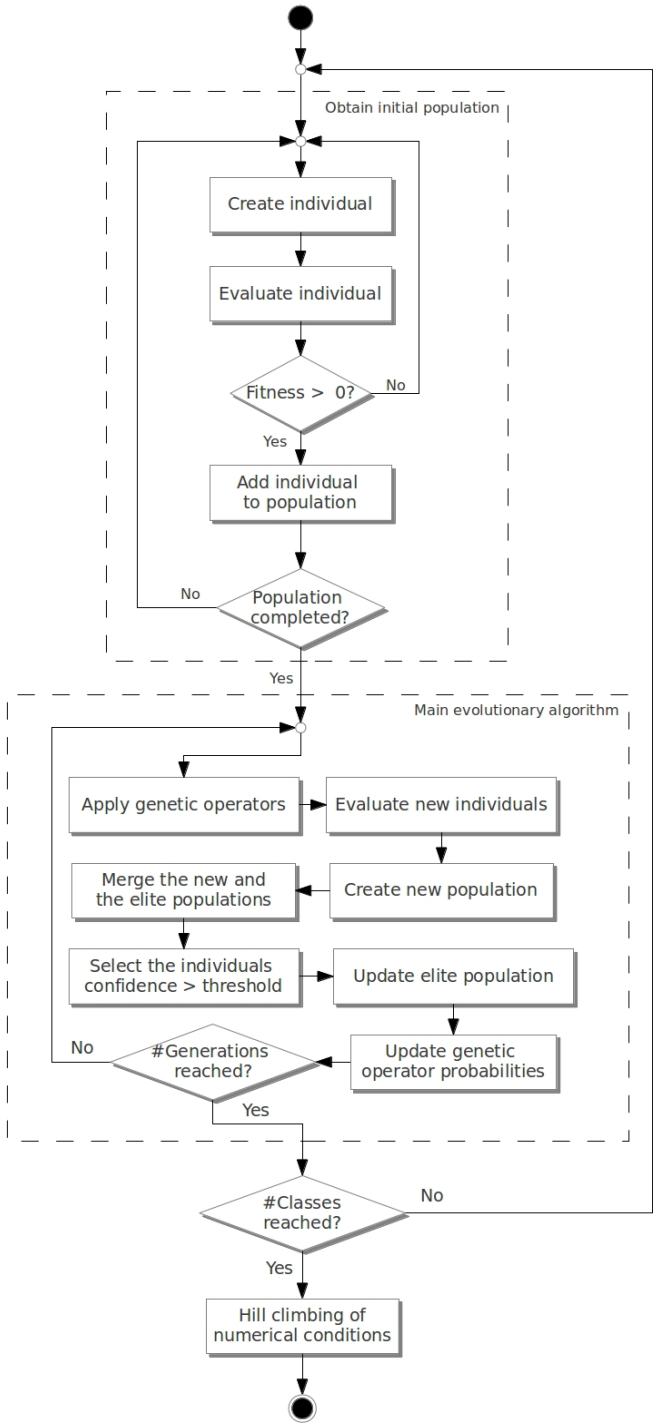


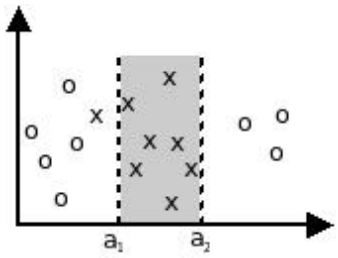
Fig. 8. The flowchart for the proposed algorithm

Two individuals are compared by using their vectors of satisfied instances (V_1 and V_2), where $V_1 \cdot V_2$ is the product of both vectors and $|V_n|$ is the norm of a vector. In this sense, the range of the values is [0, 1], and the closer to one the value is, the more similar these individuals are. In such situations where $V_1 == V_2$, one of them is randomly selected since it is not possible to determine that one of these subgroup is better than the other.

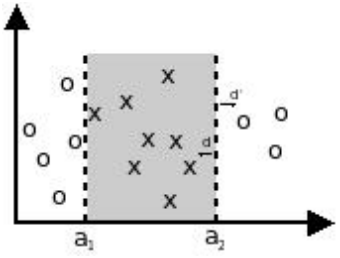
The aim of SD is the discovery of subgroups of interest to the user, so it may not be a good idea to produce a huge set of subgroups covering the same instances, but it should allow. In such a way, CGBA-SD uses a 0.8 threshold to determine whether two subgroups are similar or not. This threshold should be as high as possible while also allowing the extraction of overlapped rules. In SD, overlapped subgroups may be interesting since they can show properties of a group from a different perspective.

Finally, once the algorithm has discovered subgroups for each of the class values in the dataset under study, it runs a final procedure to optimize the intervals of the numerical conditions. The aim of this procedure is to increase or decrease the width of the intervals in order to discover high quality intervals that satisfy more instances or which prompt more reliable rules. This process is based on the well-known hill climbing optimization procedure, which performs a local search from the numerical values obtained by the previous evolutionary process. From these starting numerical values, the procedure iteratively attempts to find a better solution by increasingly or decreasingly the interval width. If the change produces a better solution, then this new solution is selected as starting interval and the procedure is repeated until no further improvements can be found. A solution is better than a previous one if it covers a higher number of instances than the previous one (see value a_1 in Figure 9) or if no new instances are covered but the interval value is located in an intermediate point, i.e., at the same distance between instances of its class and instances of a different class. Thus, this procedure is quite simple since the local search is not usually carried out on a wide search space.

Figure 9(a) shows a sample condition where the intervals (delimited by dashed lines) have not yet been optimized. Thus, after optimizing these intervals (see Figure 9(b)), the left-side of the interval is higher than the original one and includes one more instance than the original interval. Focusing on the right-side of the interval, the boundary has been optimized and established in an intermediate point, i.e., distance d to the instances of its class is equal to the distance d' to the instances of another class. Once this procedure is carried out, the algorithm returns the set of subgroups discovered in each iteration. Note that this procedure is run only on numerical attributes, so nominal datasets do not need of this procedure. Finally, any subgroup discovered by CGBA-SD has a confidence value higher than the minimum confidence threshold determined by the user.



(a) Original intervals



(b) Intervals after the optimization procedure

Fig. 9. Optimizing a numerical condition

IV. Experimental Study

In this section, the experimental study carried out is described in depth. In this study, the behavior of CGBA-SD — the algorithm was written by using JCLEC [32], a Java library for evolutionary computation[2] — is shown with respect to a series of quality measures described in Section II-B. In this experimental stage, a series of evolutionary SD algorithms were compared in detail, including NMEEF-SD [17], SDIGA [11] and MESDIF [12]. Additionally, classic SD algorithms such as CN2-SD [18] and Apriori-SD [10], were also included in the study. Next, a number of nonparametric tests were performed, demonstrating the effectiveness of the proposed approach and its ability to discover subgroups with a low complexity.

All the experiments were carried out over the same ten-fold cross-validation for each dataset. The experimentation was undertaken using 30 datasets from the UCI

TABLE II
Datasets from the UCI repository

Dataset	$\#Var$	$\#Disc$	$\#Cont$	$\#Class$	$\#Inst$
Appendicitis	7	0	7	2	106
Australian	14	8	6	2	690
Balance	4	0	4	3	625
Breast-w	9	9	0	2	699
Bridges	7	4	3	2	102
Bupa	6	0	6	2	345
Car	6	6	0	4	1728
Chess	36	36	0	2	3196
Cleveland	13	0	13	5	303
Dermatology	33	33	0	6	366
Diabetes	8	0	8	2	768
Echo	6	1	5	2	131
German	20	13	7	2	1000
Glass	9	0	9	6	214
Haberman	3	0	3	2	306
Hayesroth	4	4	0	3	132
Heart	13	6	7	2	270
Hepatitis	19	13	6	2	155
Hypothyroid	25	18	7	2	3163
Ionosphere	34	0	34	2	351
Iris	4	0	4	3	150
Led	7	0	7	10	500
Lymp	18	18	0	4	148
Marketing	13	13	0	10	8993
Mushrooms	22	22	0	2	8124
Nursery	8	8	0	5	12960
Tic-tac-toe	9	9	0	2	958
Vehicle	18	0	18	4	846
Vote	16	16	0	2	435
Wine	13	0	13	3	178

[2]JCLEC is freely available for download from http://jclec.sf.net

TABLE III
OPTIMAL PARAMETERS FOR THE EVOLUTIONARY ALGORITHMS

Algorithm	Parameter values
NMEEF-SD	Linguistic labels = 3 and 5 Minimum confidence = 0.6, 0.7, 0.8 and 0.9 Population size = 50 Maximum evaluations = 10000 Crossover probability = 0.60 Mutation probability = 0.10
MESDIF	Linguistic labels = 3 and 5 Elite population = 3, 4, 5 and 10 Population size = 100 Maximum evaluations = 10000 Crossover probability = 0.60 Mutation probability = 0.01
SDIGA	Linguistic labels = 3 and 5 Minimum confidence = 0.6, 0.7, 0.8 and 0.9 Population size = 100 Maximum evaluations = 10000 Crossover probability = 0.60 Mutation probability = 0.01
CGBA-SD	Minimum confidence = 0.6, 0.7, 0.8 and 0.9 Population size = 50 Number of generations = 100

TABLE IV
NUMBER OF PARAMETERS REQUIRED FOR EACH ALGORITHM

Algorithm	# Parameters
NMEEF-SD	1) Number of Linguistic labels 2) Confidence threshold 3) Population size 4) Maximum number of evaluations 5) Crossover probability value 6) Mutation probability value
MESDIF	1) Number of Linguistic labels 2) Elite population size 3) Population size 4) Maximum number of evaluations 5) Crossover probability value 6) Mutation probability value
SDIGA	1) Number of Linguistic label 2) Confidence threshold 3) Population size 4) Maximum number of evaluations 5) Crossover probability value 6) Mutation probability value
CGBA-SD	1) Confidence threshold 2) Population size 3) Maximum number of generations

repository[3] and a ten-fold cross validation, so the results shown for the experiments are the average values obtained for each data set for the different executions, i.e. 50 for evolutionary algorithms (5 per group of cross validation, because these algorithms are non-deterministic). Table II shows the features of each dataset used in this experimental stage, describing the number of variables ($\#Var$), number of discrete attributes ($\#Disc$), number of continuous attributes ($\#Cont$), number of classes ($\#Class$) and number of instances ($\#Inst$). As far as the evolutionary algorithms are concerned, the parameters were established from various experimental studies, which allowed us to determine the values that performed better in the algorithms (see Table III). Notice that the optimal parameters for NMEEF-SD, MESDIF and SDIGA are those given by the authors as optimal parameters, which were analyzed by *Carmona et al.* [17].

As for the classic SD algorithms, a previous discretization of the continuous values is required using Fayyad discretization [33], likewise used in papers describing CN2-SD [18] and Apriori-SD [10]. The results obtained by each algorithm are the average results obtained after running each one ten times (1 per group of cross validation) for the classical algorithms. The results obtained for each quality measure are the average results for the set of subgroups discovered, showing the number of rules, number of variables, significance, unusualness, confidence and support.

Regarding the number of parameters required to be tuned for each compared algorithm, Table IV shows the number of parameters required. CGBA-SD requires half of the parameters needed by existing evolutionary approaches.

For the sake of making a fair comparison with the CGBA-SD algorithm, the results of other algorithms used in this comparison are those given by [17] and publicly available online[4]. The website also contains the partitioned datasets used, so any researcher can use them to extend the comparison.

[3]Machine learning repository. http://archive.ics.uci.edu/ml/

[4]Further information about the comparison of CGBA-SD and other algorithms is publicly available at http://simidat.ujaen.es/NMEEF-SD

A. Computational complexity of CGBA-SD

In this experimental stage, we have carried out a computational complexity analysis to determine the efficiency of the proposed model. In this sense, each of the main procedures are analysed separately: evaluator, parent selector and genetic operators. Finally, the computational complexity of the whole algorithm is determined.

Firstly, the evaluator procedure depends on the number of individuals ($\mathcal{N}_{ind}$), instances ($\mathcal{N}_{ins}$) and attributes ($\mathcal{N}_{att}$). Mathematically, this value is defined as $\mathcal{O}(\mathcal{N}_{ins} \times \mathcal{N}_{att} \times \mathcal{N}_{ind})$. Additionally, the complexity order of the parent selector procedure depends on the population size, i.e., $\mathcal{N}_{ind}$. Finally, the computational complexity of the genetic operators depend on the number of individuals ($\mathcal{N}_{ind}$).

Analysing the computing requirements for each procedure, it is stated that $\mathcal{N}_{ind}$ is previously fixed, so it is considered as constant and its complexity order is $\mathcal{O}(1)$. Additionally, all the procedures are repeated as times as the predefined number of generations, which is also a constant value predefined. Therefore, bearing in mind all these issues, the resultant computational complexity of CGBA-SD is stated as $\mathcal{O}(\mathcal{N}_{ins} \times \mathcal{N}_{att})$. Thus, the complexity of the proposed approach is linear with regard to the number of instances and attributes.

B. Comparison between the Proposed Method and Existing EAs for SD

In this section, the quality measures described in Section II-B are used. Table V shows the results obtained by each evolutionary algorithm using the complete set of measures (the best results are set in bold typeface). The best result of each algorithm was selected, i.e., the best average result obtained for different levels of granularity for the fuzzy evolutionary algorithms and different confidence thresholds

(0.6, 0.7, 0.8 and 0.9). The value "-" means that no subgroup was discovered.

Before analyzing the results statistically, we detail how the type of data affects the results. The whole table of results[5] shows that the results obtained with CGBA-SD are better than the fuzzy algorithms when the data includes continuous attributes. Indeed, when considering only continuous attributes, CGBA-SD obtains the best results for most of the measures. The same occurs with mixed data, i.e., including both numerical and categorical attributes. Note that, in fuzzy algorithms, the interval is granularized into a number of subintervals. Hence, the proposed model behaves better, since it does not divide the interval into subintervals but rather allows for the extraction of subintervals. Finally, looking at the complete set of data (see Table V), the results show that CGBA-SD obtains the best ranking in complexity (number of rules discovered and average number of variables in each rule), significance and unusualness.

In order to analyze the results obtained, a series of statistical tests [34], [35] were carried out. The Friedman test is used to compare the results obtained and to be able to precisely analyze whether there are significant differences among the four algorithms. This test first ranks the jth of k algorithms on the ith of N datasets, and then calculates the average rank according to the F-distribution (F_F) throughout all the datasets, and calculates the Friedman statistics. If the Friedman test rejects the null-hypothesis — indicating that there are significant differences — then a Bonferroni-Dunn test is performed to reveal those differences.

Table VI shows the Friedman statistical values according to F_F with $k-1$ and $(k-1)(N-1)$ degrees of freedom and the critical interval determined by $[0,(F_F)_{p,k-1,(k-1)(N-1)}]$. Thus, if the F_F value does not belong to the critical interval, then the null-hypothesis that all algorithms perform equally well is rejected. In order to analyze whether there are significant differences between CGBA-SD and the other algorithms, the Bonferroni-Dunn test is used to reveal the difference in performance as properly depicted in Table VI for $p = 0.05$.

The results indicate that, at a significance level of $p = 0.05$ (i.e., with a probability of 95%), there are significant differences between CGBA-SD and SDIGA for all the measures. As for the MESDIF algorithm, the Bonferroni-Dunn test reveals that CGBA-SD behaves statistically better for all measures except for support. Finally, the Bonferroni-Dunn test does not determine a statistical difference in behavior with respect to NMEFF for the quality measures under study. Despite the fact that no statistical difference could be considered, CGBA-SD behaves better than NMEEF in all the measures except for support and confidence. In this regard, it is interesting to study in detail how these two measures behave (see Table VII). Regarding the support measure, no statistical difference is obtained among NMEEF, MESDIF and CGBA-SD, the latter algorithm obtaining the second best ranking, the second best average value, and the best standard deviation. Finally, despite the fact that NMEEF obtains a ranking value better than CGBA-SD for the confidence measure, our proposal obtains the best average value and the lowest standard deviation for this measure. Notice that, despite the proposed model uses the support and confidence as fitness function to evolve the rules, it uses a confidence threshold to determine which rules are the most suitable to be kept along the evolutionary process.

Analysing the results as a whole, CGBA-SD obtains the best results in interpretability, discovering the smallest set of subgroups for most of the datasets. This issue is allowed thanks to the elitist selection procedure carried out in each generation, where similar individuals — based on the instances discovered — are not included, so only a set of representative subgroups are selected. Additionally, the set of subgroups discovered by CGBA-SD comprises few conditions thanks to its optimization of the support measure. The smaller the number of conditions included in a subgroup, the more probable is to calculate a high value for the numerator of Equation 1, so the higher is the support value. Nevertheless, even when the support value is considered in the fitness function of CGBA-SD, it is of interest to note that it is not the best algorithm when the support measure is analysed but the second one. This interesting issue occurs due to the selection procedure, which enables a set of representative subgroups (not covering similar instances) to be discovered. Thus, when a subgroup having a high support value is discovered, then only subgroups having smaller support values satisfy the instances that were not

TABLE V
RANKINGS OBTAINED FOR THE EVOLUTIONARY ALGORITHMS

Measure	NMEEF-SD	MESDIF	SDIGA	CGBA-SD
# Rules	2.183	3.866	2.383	**1.566**
# Variables	1.667	3.567	2.717	**1.550**
Significance	1.983	3.267	3.067	**1.683**
Unusualness	1.683	3.433	3.267	**1.617**
Support	**1.683**	2.833	3.100	2.383
Confidence	**1.583**	3.233	3.367	1.816

[5]The values obtained for each specific dataset are publicly available at http://www.uco.es/grupos/kdis/kdiswiki/G3P-SD

TABLE VI
STATISTICAL ANALYSIS OF THE RESULTS WHEN COMPARING CGBA-SD TO THE OTHER ALGORITHMS

	Friedman		Bonferroni-Duun $p = 0.5$		
Measure	F_F	$F_{p=0.01}$	NMEEF	MESDIF	SDIGA
#Rules	51.35	[0, 4.015]	Reject	Accepted	Accepted
#Variables	39.57	[0, 4.015]	Reject	Accepted	Accepted
Significance	33.17	[0, 4.015]	Reject	Accepted	Accepted
Unusualness	52.31	[0, 4.015]	Reject	Accepted	Accepted
Support	20.73	[0, 4.015]	Reject	Reject	Accepted
Confidence	46.73	[0, 4.015]	Reject	Accepted	Accepted

TABLE VII
ANALYSIS OF SUPPORT AND CONFIDENCE MEASURES

	Ranking		
Measure	NMEEF	MESDIF	CGBA-SD
Support	1.683	2.833	2.383
Confidence	1.583	3.233	1.816
	Average $\pm$ standard deviation		
Support	0.710±0.3	0.573±0.2	0.622±0.2
Confidence	0.781±0.2	0.598±0.2	0.793±0.1

previously satisfied.

As for the significance and unusualness measure, CGBA-SD obtains the best values since it optimizes support and confidence at time. Both quality measures (significance and unusualness) consider the antecedent of the subgroup as an important part in their definitions (see Equation 3 and 4). More specifically, the unusualness quality measure includes the confidence in its definition, so the higher the confidence value, the higher is the unusualness value.

To conclude this analysis, CGBA-SD is statistically better than MESDIF and SDIGA when all measures are used. When only the support measure is considered, it is not possible to assert that there are significant differences between CGBA-SD and MESDIF. However, despite this fact, the ranking obtained by CGBA-SD in this measure is better and the average support value is also better than MESDIF. Finally, when comparing CGBA-SD and NMEEF-SD, it is not possible to statistically assert that there are significance differences between these two algorithms. Nevertheless, CGBA-SD obtains a ranking value better than MESDIF for all the measures except for the support and confidence. Additionally, CGBA-SD obtains a higher average confidence value with a lower standard deviation. Further, CGBA-SD requires a lower number of parameters and is able to discover subgroups in any dataset, in contrast to NMEEF, which fails in mining subgroups over some datasets like $Vehicle$.

The analysis was carried out by using the best result of each algorithm for each dataset, i.e., the best average result obtained for different levels of granularity and different confidence thresholds. In this sense, it is interesting to provide an analysis that consider the algorithms with different confidence thresholds (0.6, 0.7, 0.8 and 0.9) and levels of granularity (for the NMEEF-SD algorithm). The complete set of results for each dataset can be publicly found[6]. Thereby, it is demonstrated that the proposed algorithm is very promising for measures of comprehensibility complexity, generality and interest, it does not depend on the minimum confidence threshold, and it does not require as many parameters as existing algorithms do. Additionally, the results for the confidence measure do not differ so much from CGBA-SD and NMEEF-SD.

C. Comparison of CGBA-SD and classic algorithms

In this section, a comparison between CGBA-SD and classic algorithms for SD is carried out (see Table VIII). To this end, a subset of the 30 datasets in Table II was used, since classic algorithms do not work properly with high dimensionality, including Appendicitis, Australian, Balance, Breast-w, Bridges, Bupa, Car, Cleveland, Diabetes, Echo, German, Haberman, Hayes-roth, Heart, Hepatitis, Iris, Led, Tic-tac-toe, Vote and Wine. The results were provided by *Carmona et al.* [17] and allow new researchers to improve the comparison available online[7].

Table IX illustrates the Friedman statistical values according to F_F with $k-1$ and $(k-1)(N-1)$ degrees of freedom and the critical interval determined by $[0,(F_F)_{p,k-1,(k-1)(N-1)}]$. Similarly to the previous analysis, if the F_F value does not belong to the critical interval, then the null-hypothesis that all algorithms perform equally well is rejected. In such a situation, the Bonferroni-Dunn test (using a significance level of $p = 0.05$) is carried out to analyze whether there are significant differences between CGBA-SD and the other algorithms.

TABLE VIII
COMPARISON BETWEEN CGBA-SD AND CLASSICAL SD ALGORITHMS

Measure	CN2-SD	Apriori-SD	CGBA-SD
# Rules	2.625	2.275	**1.100**
# Variables	2.250	**1.350**	2.400
Significance	**1.750**	2.350	1.900
Unusualness	2.250	2.275	**1.475**
Support	1.900	2.450	**1.650**
Confidence	2.650	1.750	**1.600**

TABLE IX
STATISTICAL ANALYSIS OF THE RESULTS WHEN COMPARING CGBA-SD TO CLASSICAL ALGORITHMS

	Friedman		Bonferroni-Duun $p = 0.5$	
Measure	F_F	$F_{p=0.01}$	CN2	Apriori
#Rules	33.50	[0, 5.21]	Accepted	Accepted
#Variables	9.04	[0, 5.21]	Reject	Accepted
Significance	2.05	[0, 5.21]	Friedman test accepted	
Unusualness	7.96	[0, 5.21]	Accepted	Accepted
Support	3.82	[0, 5.21]	Friedman test accepted	
Confidence	9.04	[0, 5.21]	Accepted	Reject

The results indicate that, at a significance level of $p = 0.05$ (i.e., with a probability of 95%), there are significant differences between CGBA-SD and CN2 for the number of rules, unusualness and confidence. As for the Apriori algorithm, the Bonferroni-Dunn test reveals that CGBA-SD behaves statistically better for the number of rules, number of variables and unusualness.

To conclude this analysis, CGBA-SD is seen to be an interesting algorithm for SD, discovering a more comprehensible set of rules, i.e., a set of few rules having fewer variables. Furthermore, the algorithm proposed in this paper uncovers more reliable rules than the classic algorithms, as well as obtaining the best ranking for the unusualness, support and confidence measures. Additionally, it should be noted that CGBA-SD does not need any discretization process, unlike classic algorithms, which were discretized by using the Fayyad discretization technique.

D. Comprehensibility of the Subgroups

A subgroup is considered as comprehensible if it is easy to understand, so the lower the number of variables the better its understandability. The notion of comprehensibility, also known as interpretability or understandability in contexts such as data mining, is highly subjective and, sometimes, depends on the knowledge of experience. In an overview on subgroup discovery [5], the authors stated that the measures of complexity were related to the comprehensibility of the subgroups. They defined as complexity measures the number of rules and the number of variables.

In previous sections, we have carried out an analysis of the behaviour of the proposed model with regard to comprehen-

[6]Any additional information about the experiments is available at http://www.uco.es/grupos/kdis/kdiswiki/G3P-SD

[7]Further information available at http://simidat.ujaen.es/NMEEF-SD

TABLE X
EXAMPLES OF RULES AND THE RESULTS FOR THE QUALITY MEASURES FOR THE $Diabetes$ DATASET

NMEEF-SD				
Rule	Significance	Unusualness	Support	Confidence
1	8.736	0.078	0.800	0.774
2	1.235	0.052	0.980	0.700
3	6.117	0.051	0.740	0.764
4	5.597	0.087	0.880	0.769
5	4.793	0.083	0.940	0.711
6	7.391	0.069	0.820	0.732
7	6.058	0.081	0.860	0.728
8	5.057	0.066	0.840	0.747
9	6.467	0.054	0.630	0.595
1	IF (plass=Medium AND pres=Medium AND age=Low) THEN Negative			
2	IF (plass=Medium) THEN Negative			
3	IF (plass=Medium AND pres=Medium AND pedi=Low AND insu=Low) THEN Negative			
4	IF (plass=Medium AND age=Low) THEN Negative			
5	IF (plass=Medium AND insu=Low AND age=Low)			
6	IF (plass=Medium AND pres=Medium AND insu=Low) THEN Negative			
7	IF (plass=Medium AND pres=Medium) THEN Negative			
8	IF (plass=Medium AND insu=Low AND pedi=Low) THEN Negative			
9	IF (plass=Medium) THEN Positive			

CGBA-SD				
Rule	Significance	Unusualness	Support	Confidence
1	10.655	0.099	0.540	0.900
2	6.338	0.070	0.420	0.875
3	7.531	0.071	0.581	0.721
1	IF plass [21.61, 102.11] THEN Negative			
2	IF plass [4.36, 107.55] AND pres [25.16, 81.50] THEN Negative			
3	IF plas IN [20.31, 81.98] THEN Positive			

sibility measures. The numerical comparisons have revealed that CGBA-SD behaves quite well, obtaining the best ranking in both measures: number of rules and number of variables.

In order to back up the previous numerical analysis, we have selected a dataset ($Diabetes$) and the most representative evolutionary algorithms (the one that better performs). Table X shows the subgroups discovered from a single run.

As shown, NMEEF-SD discovers nine subgroups, whereas CGBA-SD mines only three. Analysing the subgroups discovered by NMEEF-SD, most of the rules are quite similar. For instance, rules 1, 3, 4 and 7 comprise the same set of variables, so a reduction of the subgroup could be considered. In this sense, CGBA-SD only requires two rules (1 and 2) to represent the same knowledge. Thus, CGBA-SD enables to organize the space of solutions by using a low number of subgroups.

Regarding the number of variables, the results have shown that CGBA-SD requires one or two variables when the $Diabetes$ dataset is considered. On the contrary, NMEEF-SD discovers subgroups that comprise three variables in most of the cases, so the information represented by CGBA-SD is more understandable.

V. CONCLUDING REMARKS

In this paper, a novel G3P algorithm for mining SD was presented and described in depth. The aim of this algorithm is to harness the features of G3P to solve the requirements of SD, i.e., rules that have a clear and flexible structure and obtain interesting subgroups according to a series of quality measures, which could be classified as complexity, generality, precision and interest.

This evolutionary algorithm, called CGBA-SD, makes use of a grammar to properly encode individuals, allowing for a mining process in any domain to be carried out. The algorithm also defines two genetic operators that allow to preserve the diversity of the population. To this end, the probabilities of both genetic operators are self-adapted depending on the quality of the rules discovered in each generation. Additionally, the algorithm presented here does not require any discretization of the numerical values. Also, the numerical conditions discovered are optimized by searching for the best boundaries for each attribute included in the rules.

One of the main features of this algorithm is its ability to discover comprehensible subgroups, i.e. a few set of rules having few variables, in an evolutionary way and without the need for tuning as many parameters as required by existing evolutionary algorithms in this field.

Finally, a complete experimental study was performed, making an exhaustive comparison between CGBA-SD and the other existing algorithms (NMEEF-SD, MESDIF, SDIGA, CN2-SD and Apriori-SD). The experimental results, which were backed up with a series of nonparametric tests, reveal the effectiveness of CGBA-SD as a means of discovering comprehensible subgroups with better behavior than the other algorithms in measures of complexity, interest, and precision.

To sum up, CGBA-SD is a robust algorithm for SD which obtains comprehensible and highly representative rules. The subgroups it discovers provide interesting values for any quality measure considered for SD.

ACKNOWLEDGMENTS

This work has been supported by the Ministry of Science and Technology project TIN-2011-22408, FEDER funds and the Spanish Ministry of Education under the FPU grant AP2010-0041.

REFERENCES

[1] C. Romero, J. M. Luna, J. R. Romero, and S. Ventura, "RM-Tool: A framework for discovering and evaluating association rules," *Advances in Engineering Software*, vol. 42, no. 8, pp. 566–576, 2011.

[2] J. L. Olmo, J. R. Romero, and S. Ventura, "Using ant programming guided by grammar for building rule-based classifiers," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,*, vol. 41, no. 6, pp. 1585–1599, 2011.

[3] P. K. Novak, N. Lavra, and G. I. Webb, "Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining," *Journal of Machine Learning Research*, vol. 10, pp. 377–403, 2009.

[4] V. S. Cherkassky and F. Mulier, *Learning from Data: Concepts, Theory, and Methods*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1998.

[5] F. Herrera, C. J. Carmona, P. González, and M. J. del Jesus, "An overview on subgroup discovery: Foundations and applications," *Knowledge and Information Systems*, vol. 29, no. 3, pp. 495–525, 2011.

[6] S. D. Bay and M. J. Pazzani, "Detecting group differences: Mining contrast sets," *Data Mining and Knowledge Discovery*, vol. 5, no. 3, pp. 213–246, 2001.

[7] G. D. and J. L., "Efficient mining of emerging patterns: Discovering trends and differences," in *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1999, pp. 43–52.

[8] W. Klösgen, "Explora: A multipattern and multistrategy discovery assistant," in *Advances in Knowledge Discovery and Data Mining*, 1996, pp. 249–271.

[9] S. Wrobel, "An algorithm for multi-relational discovery of subgroups," in *Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery*, ser. PKDD '97. London, UK, UK: Springer-Verlag, 1997, pp. 78–87.

[10] B. Kavek and N. Lavra, "APRIORI-SD: Adapting association rule learning to subgroup discovery," *Applied Artificial Intelligence*, vol. 20, no. 7, pp. 543–583, 2006.

[11] M. J. del Jesus, P. González, F. Herrera, and M. Mesonero, "Evolutionary fuzzy rule induction process for subgroup discovery: A case study in marketing," *Fuzzy Systems, IEEE Transactions on*, vol. 15, no. 4, pp. 578 –592, aug. 2007.

[12] C. J. Carmona, P. González, M. J. del Jesus, M. Navío-Acosta, and L. Jimnez-Trevino, "Evolutionary fuzzy rule extraction for subgroup discovery in a psychiatric emergency department," *Soft Computing*, vol. 15, no. 12, pp. 2435–2448, 2011.

[13] J. M. Luna, J. R. Romero, and S. Ventura, "Design and behavior study of a grammar-guided genetic programming algorithm for mining association rules," *Knowledge and Information Systems*, vol. 32, no. 1, pp. 53–76, 2012.

[14] R. I. McKay, N. X. Hoai, P. A. Whigham, Y. Shan, and M. O'Neill, "Grammar-based genetic programming: a survey," *Genetic Programming and Evolvable Machines*, vol. 11, no. 3-4, pp. 365–396, 2010.

[15] F. Alonso, L. Martínez, A. Santamaría, A. Pérez, and J. P. Valente, "GGGP-based method for modeling time series: operator selection, parameter optimization and expert evaluation," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, ser. GECCO '10, Portland, Oregon, USA, 2010, pp. 989–990.

[16] J. M. Luna, J. R. Romero, C. Romero, and S. Ventura, "Discovering subgroups by means of genetic programming," in *Proceedings of the 16th European Conference*, ser. EuroGP 2013, Vienna, Austria, 2013, pp. 121–132.

[17] C. J. Carmona, P. González, M. J. del Jesus, and F. Herrera, "NMEEF-SD: Non-dominated multiobjective evolutionary algorithm for extracting fuzzy rules in subgroup discovery," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 5, pp. 958 –970, oct. 2010.

[18] N. Lavrač, B. Kavšek, P. Flach, and L. Todorovski, "Subgroup discovery with cn2-sd," *Journal of Machine Learning Research*, vol. 5, pp. 153–188, Dec. 2004.

[19] W. Duivesteijn and A. J. Knobbe, "Exploiting false discoveries - statistical validation of patterns and quality measures in subgroup discovery," in *Proceedings of the 11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, 2011, pp. 151–160.

[20] V. Jovanoski and N. Lavrac, "Classification rule learning with APRIORI-C," in *Proceedings of the 10th Portuguese Conference on Artificial Intelligence on Progress in Artificial Intelligence, Knowledge Extraction, Multi-agent Systems, Logic Programming and Constraint Solving*, ser. EPIA '01. London, UK, UK: Springer-Verlag, 2001, pp. 44–51.

[21] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proceedings of the 20th International Conference on Very Large Data Bases*, ser. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 487–499.

[22] P. Clark and T. Niblett, "The cn2 induction algorithm," *Machine Learning*, vol. 3, no. 4, pp. 261–283, 1989.

[23] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning I,II,III," *Information Sciences*, vol. 8-9, pp. 199–249, 301–357, 43–80, 1975.

[24] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization," in *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, K. Giannakoglou *et al.*, Eds. International Center for Numerical Methods in Engineering (CIMNE), 2002, pp. 95–100.

[25] C. A. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. Berlin: Springer-Verlag, 2007.

[26] D. Kalyanmoy, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, 2000.

[27] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*, 1st ed. A Bradford Book, 1992.

[28] P. González-Espejo, S. Ventura, and F. Herrera, "A Survey on the Application of Genetic Programming to Classification," *IEEE Transactions on Systems, Man and Cybernetics: Part C*, vol. 40, no. 2, pp. 121–144, 2010.

[29] A. Tsakonas, G. Dounias, J. Jantzen, H. Axer, B. Bjerregaard, and D. G. von Keyserlingk, "Evolving rule-based systems in two medical domains using genetic programming," *Artificial Intelligence in Medicine*, vol. 32, no. 3, pp. 195–216, 2004.

[30] R. Poli, W. B. Langdon, and N. F. McPhee, *A Field Guide to Genetic Programming*. Lulu Enterprises, UK Ltd, 2008.

[31] P. Wang, K. Tang, T. Weise, E. P. K. Tsang, and X. Yao, "Multiobjective genetic programming for maximizing ROC performance," *Neurocomputing*, no. 0, 2013.

[32] S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás, "JCLEC: A java framework for evolutionary computation," *Soft Computing*, vol. 12, no. 4, pp. 381–392, 2008.

[33] U. M. Fayyad and K. B. Irani, "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning," in *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, 1993, pp. 1022–1027.

[34] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the cec'2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

[35] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

José María Luna was born in Córdoba, Spain, in 1985. He received a B.Sc. degree from the University of Córdoba in 2007, and a M.Sc. degree from the University of Córdoba, Spain, in 2009, both in Computer Science. Since 2009, he has been with the Department of Computer Science and Numerical Analysis, the University of Córdoba, Spain, where he is currently working towards obtaining the Ph.D., as well as research tasks. His research interests include the application of evolutionary computation, association rule mining and its applications. José María Luna is a Student Member of the IEEE Computer, Computational Intelligence and Systems, Man and Cybernetics societies.

José Raúl Romero is currently an Associate Professor at the Department of Computer Science of the University of Córdoba, Spain. He received his Ph.D. in Computer Science from the University of Malaga, Spain, in 2007. He has worked as an IT consultant for important business consulting and technology companies for several years. His current research interests include the use of bio-inspired algorithms for data mining, the industrial use of formal methods and model-driven software development and its applications. Dr. Romero is a member of IEEE, the ACM, and the Spanish Technical Normalization Committee AEN/CTN 71/SC7 of AENOR. He can also be reached at http://www.jrromero.net.

Cristóbal Romero is currently an Associate Professor at the University of Córdoba. He received his BSc and Ph.D. degrees in computer science from the University of Granada, Spain, in 1996 and 2003, respectively. He has published more than 40 international papers, 15 of which have been published in international journals. He is the co-editor of two books specically regarding EDM. His current research interest is focussed on the application of data mining in e-learning systems. Dr. Romero is a member of the IEEE Computer Society, the International EDM Working Group and the steering committee of several conferences about education, personalisation, and data mining.

Sebastián Ventura is currently an Associate Professor in the Department of Computer Science and Numerical Analysis at the University of Córdoba, where he heads the Knowledge Discovery and Intelligent Systems Research Laboratory. He received his BSc and Ph.D. degrees in sciences from the University of Córdoba, Spain, in 1989 and 1996, respectively. He has published more than 150 papers in journals and scientific conferences, and he has edited three books and several special issues in international journals. He has also been engaged in 11 research projects (being the coordinator of three of them) supported by the Spanish and Andalusian governments and the European Union. His main research interests are in the fields of soft-computing, machine learning, data mining, and their applications. Dr. Ventura is a senior member of the IEEE Computer, the IEEE Computational Intelligence and the IEEE Systems, Man and Cybernetics Societies, as well as the Association of Computing Machinery (ACM).