Reducing gaps in quantitative association rules: a genetic programming free-parameter algorithm

José María Luna[a], José Raúl Romero[a], Cristóbal Romero[a] and Sebastián Ventura[a,b,*]
[a]*Department of Computer Science and Numerical Analysis, University of Cordoba, Albert Einstein Building, Rabanales Campus, Córdoba, 14071, Spain*
[b]*Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia Kingdom*

Abstract

The extraction of useful information for decision making is a challenge in many different domains. Association rule mining is one of the most important techniques in this field, discovering relationships of interest among patterns. Despite the mining of association rules being an area of great interest for many researchers, the search for well-grouped continuous values is still a challenge, discovering rules that do not comprise patterns which represent unnecessary ranges of values. Existing algorithms for mining association rules in continuous domains are mainly based on a non-deterministic search, requiring a high number of parameters to be optimised. These parameters hinder the mining process, and the algorithms themselves must be known to those data mining experts that want to use them. We therefore present a grammar guided genetic programming algorithm that does not require as many parameters as other existing approaches and enables the discovery of quantitative association rules comprising small-size gaps. The algorithm is verified over a varied set of data, comparing the results to other association rule mining algorithms from several paradigms. Additionally, some resulting rules from different paradigms are analysed, demonstrating the effectiveness of our model for reducing gaps in numerical features.

Keywords: Quantitative association rules, grammar guided genetic programming, evolutionary computation, data mining

1. Introduction

Generally speaking, different business areas require the extraction of useful and hidden knowledge from their data, as the raw information is meaningless without in-depth analysis and study. Currently, many research studies have focused their studies on the extraction of useful knowledge [15] of great interest for customers [29], companies [26], or even medical environments [22].

One of the most important techniques for the extraction of hidden knowledge is association rule mining (ARM) [6], an unsupervised learning task that includes approaches of a descriptive nature, whose aim is the extraction of strong and interesting relationships among patterns hidden in the data. This technique has received more and more attention since it was defined by *Agrawal et al.* [2]. ARM was originally designed for market basket analysis to obtain relationships between products such as *diapers* $\rightarrow$ *beer*, which describes the high probability of someone who is buying diapers also buying beer. It would allow shop-keepers to exploit this particular relationship by moving the products far away from one another on the shelves, thus there is a greater chance that one see something interesting to buy.

Even though nominal patterns are the focus of many researchers [9][21], the use of continuous values is increasingly important, and the first algorithms used for this sort of patterns worked by discretising the search space. It is only recently that researchers are focusing on the extraction of such

* Corresponding autor. E-mail: sventura@uco.es

relations in a direct way [19][28], which is not a trivial issue. In general, existing algorithms in this field are based on evolutionary methodologies [3][37][38], which overcome the high computational time and the memory requirements which arise when using such a huge search space, caused by the existence of continuous ranges of values.

The use of evolutionary algorithms [4][5] has solved many problems in the ARM field [17]. One of the most important is the use of both numerical and nominal attributes without requiring a previous discretization step. Nevertheless, this issue was overcome by looking for the correct amplitude, and no analysis of the instance distribution was considered. The instance distribution is mandatory, especially in descriptive learning tasks like ARM. Without both a further analysis of the instance distribution and the continuous values, a huge number of extracted rules could comprise unnecessary ranges of values. For instance, a quantitative pattern (comprising a range of continuous values) is not well-defined if it comprises huge gaps, that is, its instances are not exceptionally well-grouped.

Existing evolutionary algorithms do not carry out an accurate group of such values, meaning that some of the discovered quantitative association rules could be meaningless since they cover patterns that are almost always satisfied. Most of these algorithms are focused on the maximization of the values of a set of specific quality measures, but the fact of mining rules with a highly representative range of values hampers this issue. To the best of our knowledge, it is more important to mine rules with highly representative patterns (despite the fact that the support values slightly decrease), than discover meaningless rules with high support values.

In this paper, we propose a solution to this problem by using an evolutionary methodology [12][30]. The aim is to reduce the size of the gaps within quantitative patterns. Even when the problem could be tackled in many different ways, we suggest the use of a grammar-guided genetic programming (G3P) model [10], and extension of genetic programming [23] that has achieved excellent results in unsupervised learning tasks [18]. Not only is the proposed model able to extract and properly define numerical patterns, but it also deals with discrete domains. Besides, evolutionary ARM proposals usually comprise a huge set of parameters to be tuned, and this task could be a difficult process for non-expert users in evolutionary computation [11][39]. In this regard, the proposed model self-adapts its parameters, which is not an innovation since it is a well-studied area by many researchers [20]. This self-adaptation is highly useful for users with no experience in evolutionary computation.

Finally, in order to demonstrate the performance and usefulness of the proposed model, a series of experiments were carried out, which are fully described in the experimental section. In the analysis of the proposed model, a comparison between samples quantitative association rules obtained from different algorithms is carried out. The analysis reveals that the proposed model extracts rules comprising small gaps, so the set of instances described by the rules is highly grouped.

This paper is arranged as follows. First, Section 2 describes the ARM task, stating the most important quality measures in this field, as well as the most important algorithms for mining association rules. The proposed model is described in depth in Section 3. Section 4 presents the data sets used in the experiments and a detailed analysis of the results obtained. Finally, some concluding remarks are presented in Section 5.

2. Preliminaries

For a better comprehension of ARM, this section describes the most important aspects and the general background of ARM.

2.1. Quality measures

ARM aims to discover frequent relations among patterns. An association rule is denoted as an implication of the form $A \rightarrow C$, where A stands for the antecedent and C for the consequent. Both A and C are sets of patterns which do not have attributes in common, that is, $A \cap C = \emptyset$.

To begin with, ARM searches for highly representative rules [27], meaning that the rules discovered are those which are satisfied at least a minimum number of times [2]. To demonstrate this in a formula, where the antecedent is A, the consequent C, and the number of transactions in a data set $|D|$, the support (see Eq. (1)) indicates the proportion of transactions T including both A and C in the data set D.

$$sup(A \rightarrow C) = \frac{\left|\{A \cup C \subseteq T, T \in D\}\right|}{|D|} \qquad (1)$$

In addition to support, confidence is a quality measure that appears in any problem where ARM is applied. This quality measure enables the reliability of the rule to be determined, meaning that the higher the value of this measure, the more accurate the rule is (see Eq. (2)).

$$conf(A \rightarrow C) = \frac{\left|\{A \cup C \subseteq T, T \in D\}\right|}{\left|\{A \subseteq T, T \in D\}\right|} \quad (2)$$

Support and confidence are broadly considered as the best quality measures in ARM, and a great variety of proposals make use of them [17][21][28]. However, many authors have considered additional quality measures, lift (see Eq. (3)) being one them [24].

$$lift(A \rightarrow C) = \frac{\sup(A \rightarrow C)}{\sup(A) \cdot \sup(C)} \quad (3)$$

Conviction [36] is another quality measure proposed to tackle some of the weaknesses of confidence and support. This measure, which is properly defined in Eq.(4), represents the degree of implication of a rule, and values which fall further from the unity indicate interesting rules.

$$conviction(A \rightarrow C) = \frac{1 - \sup(C)}{1 - conf(A \rightarrow C)} \quad (4)$$

Leverage (see Eq. (5)) is also a well-known quality measure in ARM. Similarly to lift, leverage [16] calculates the proportion of additional cases covered by both *A* and *C* above those expected where both *A* and *C* were independent of each other. Values close to zero therefore imply uninteresting rules.

$$leverage(A \rightarrow C) = \sup(A \rightarrow C) - (\sup(A) \cdot \sup(C)) \quad (5)$$

2.2. ARM algorithms

The first algorithms for mining association rules, including Apriori [2], FP-Growth [9] and Predictive-Apriori [32], were based on exhaustive search methodologies. These algorithms initially mine frequent patterns, i.e., any pattern that is satisfied with at least a minimum support value, and then discover association rules which satisfy a minimum confidence threshold value from the set of frequent patterns. In order to produce a good performance, this sort of algorithm reduces the number of candidate patterns by using the anti-monotone property, which establishes that if a length-*k* pattern is not frequent, none of its length-(*k*+1) super-sets can be frequent.

These algorithms are not appropriate for data sets with a large number of frequent patterns resulting from relatively low minimum support threshold values. The use of exhaustive search methodology also indicates the impossibility of being applied to numerical data sets, where a previous discretisation step is required to deal with this kind of data set.

The problems which arose from exhaustive search algorithms motivated researchers to deal with an evolutionary methodology. Therefore, many evolutionary ARM algorithms [3][17][38] were proposed in order to overcome the existing problems, such as large memory requirements and huge computational time, having to deal with numerical attributes, and the discovery of association rules in two steps, as it is tedious to repeatedly scan the database to check a large set of candidate patterns.

The use of evolutionary algorithms [34][35], especially genetic algorithms [1][8][25][33], is considered to be one of the most successful search techniques used in computing in order to find both approximate and exact solutions [14][31]. These algorithms are suggested in cases where the search space is too large to use deterministic search methods. Quant-Miner [28] is important in this field, using a genetic algorithm which dynamically discovers interesting intervals in association rules by optimising both the support and the confidence. Another important contribution in this field was carried out by *Luna et al.* [17], who proposed a grammar guided genetic programming (G3P) algorithm to solve all the aforementioned problems. This algorithm, called G3PARM (Grammar Guided Genetic Programming Association Rule Mining), makes use of a grammar which can be applied to association rules in any domain and does not require previous extraction of interesting items. G3PARM mines association rules with high support and confidence values, guiding the search process through a set of genetic operators which search for frequent rules.

More recently, other bio-inspired grammatical proposals have been presented in this field [21], such as GBAP-ARM (Grammar-Based Ant Programming for Association Rule Mining) and MOG-BAP-ARM (Multi-Objective Grammar-Based Ant Programming for Association Rule Mining). These two ant programming approaches were developed based on two different methodologies; one following a single-objective rule evaluation point of view, and the other one, a Pareto-based methodology. The

use of grammars to create rules is therefore of great interest for many researchers, due to its ability to restrict search space and represent solutions in hierarchical structures of variable-length, where the size, shape and structural complexity are not constrained a priori.

The model proposed here highly differ from existing G3P algorithms in the ARM field. Firstly, it uses a different context-free grammar to mine numerical attributes as range of values. In existing G3P proposals, numerical attributes are considered by using logical operators such as “greater than”, “less than”, “greater or equal to”, and “less or equal to”. Thus, specific ranges of values are hardly obtained. Secondly, existing G3P algorithms in this field do not consider the reduction of gaps with no patterns, and no analysis about the instance distribution is carried out. This issue is mandatory in the proposed model, which demonstrates the importance of reducing the gaps in numerical attributes. Thirdly, the proposed algorithm is designed to self-adapt its parameters along the evolutionary process, a well-studied area in evolutionary computation, avoiding a previous parameter tuning that could be hard for non-expert users. Finally, the new model introduces a fitness function whose aim is the optimisation, as defined in Section 3.5, of three functions that describe a specific behaviour.

3. Mining quantitative association rules with grammar-guided genetic programming

This section describes in detail the features of the proposed algorithm. First, the general schema is properly presented. Secondly, the encoding criterion is described in depth. Finally, the genetic operator and the evaluation process are presented.

3.1. Main idea behind the proposal

The major feature of the proposed approach is its ability to reduce gaps from the range of values in numerical attributes. Thus, the aim is to mine quantitative association rules comprising small gaps instead of highly frequent rules without considering the distribution of the covered instances. Additionally, the proposed model provides a number of advantages in different ways. It is a self-adaptive algorithm, so no so many parameters are required to be tuning beforehand. It provides a context-free grammar to represent solutions, so the structure of the desired solutions is previously established and the search space is reduced. Also, both continuous and discrete domains could be mined without a preprocessing step.

In the proposed approach, its main process accordingly evaluates the rules discovered by considering how the continuous patterns are grouped. The support and confidence of each rule is not therefore the only point of interest, but also the distribution of instances satisfied by the rule. The aim of mining this type of quantitative association rules is to select the right amount of values to contain as few gaps as possible. We consider gaps to be spaces that do not comprise any instance. Let us consider the sample range of values [A, B] (see Figure 1) that comprises 15 instances. Two sample intervals could be obtained, for example, *X* and *Y*, comprising 10 and 13 instances, respectively. Analysing the support of each interval, *Y* seems to be more interesting than *X*. However, when analysing the distribution of instances within the interval, *X* is of high interest as its instances are uniformly distributed. On the contrary, *Y* comprises a gap *Z*, so its instances are not as well distributed, as in *X*.

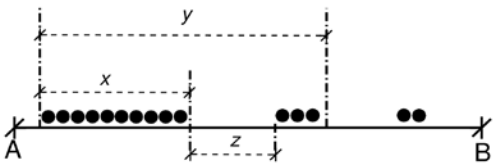


Figure 1. Sample range of values that represents two sample intervals with different distribution of instances

The proposed algorithm (see Algorithm 1) is based on a generic evolutionary process with elitism. It starts by generating an initial set of individuals (see line 3) which conform to a context-free grammar. This set of individuals, or general population, depends on the number of rules to be mined, in a relation of 5 individuals per solution to be mined.

The aim of the proposed algorithm is to obtain the best *n* (parameter required to be fixed by the user) solutions or rules according to a specific fitness function. These *n* quality rules are saved in a pool of individuals that acts as an elitist population. In each generation, the elite population is updated with the best *n* individuals from the joint of both the elite and the general population. This means that individuals included in it are ranked according to their fitness function values, and the best individuals are kept for

new generations. A promising solution is never lost unless a better one is found.

Algorithm 1 Proposed algorithm

Require: *n*
Ensure: *elitePopulation*
1: *elitePopulation* ← ∅
2: *parents* ← ∅
3: *generalPopulation* ← createIndividuals(*n*)
4: evaluate(*generalPopulation*)
5: *stoppingCriterion* ← false
6: **while(***stoppingCriterion = false)*
7: *parents* ← getParents(*elitePopulation* ∪ *generalPopulation*)
8: *offspring* ← geneticOperator(*parents*)
9: evaluate(*offspring*)
10: *generalPopulation* ← *offspring*
11: *elitePopulation* ← getBest(*n*, *elitePopulation* ∪ *generalPopulation*)
12: updateGeneticProbability(*elitePopulation*)
13: **if** (elitePopulation does not improve && maximum genetic probability reached)
14: *stoppingCriterion* ← true
15: **end if**
16: **end while**
17: **return** *elitePopulation*

For the sake of generating new individuals in each generation of the evolutionary process, a genetic operator, described in subsequent sections, is applied (lines 6 and 7). This genetic operator is used based on a probability value. Current evolutionary algorithms in ARM require fixed values, meaning that the optimal probability values are determined by the data miner, which in turn is based on the data set used. A major feature of the algorithm presented in this paper is its ability to self-update the genetic operator probability value (line 10). Thus, no previous study of the parameters to obtain optimal results is required. A more detailed description of this updating process is presented in Section 3.4.

The proposed algorithm is an iterative process that does not require a maximum number of generations but a stopping criterion (lines 11 to 13). It is based on the improvement of the average fitness function value of the individuals in the elite population. Once this average fitness value does not improve with the passing of the generations and the maximum genetic probability value is reached, then the algorithm finishes and the rules saved in the elite population are returned to the user.

3.2. Encoding criterion

The proposed model is based on the use of a context-free grammar (CFG) which defines all the possible solutions that could be obtained. Each individual is represented in a derivation syntax-tree as a sentence which conforms to the grammar. To obtain individuals, a series of production rules is applied beginning from the start symbol <Rule>, which always has a child node representing the antecedent and the consequent of the rule. Considering the grammar *G* defined in this problem the following language is obtained: L(G) = {(AND Condition)^n Condition → (AND Condition)^m Condition: $n \geq 0$, $m \geq 0$}. A condition is a 3-tupla comprising attribute, operator and value. The grammar is therefore well-defined and structured, as any rule with at least one condition in the antecedent and consequent is obtained. Notice that the antecedent and consequent are disjoints sets, meaning that they have no items in common. Using this grammar, it is possible to mine any association rule containing either numerical or nominal features. Numerical attributes are used applying the operator IN and randomly selecting two feasible values within the feasible range of values. As for categorical attributes, they could be considered as expressions in both of the following ways: $X = u$ or $X != u$, where X is a categorical attribute and u is a value in the domain D of X. The expression $X != u$ indicates that X takes any value in D\{u}. For a domain D of a categorical attribute, the support of any value u in this domain might be very low. Using the operator "!=", it is possible to obtain a higher support for this attribute. For example, for a categorical attribute X in a domain D = {a, b, c, d}, the support of X = a is 0.14, whereas the support of X != a will be 0.86.

3.3. Genetic operator

In order to obtain new individuals in each generation of the evolutionary process, the proposal described in this paper uses a typical GP mutation operator to introduce diversity into the population, avoiding entrapment in non-optimal solutions. No recombination is required in the proposed algorithm since it highly converges to the optimal solutions thanks to the elite population.

The genetic operator randomly chooses a sub-tree from the tree structure of one individual, generating a new sub-tree. A major restriction of the application of a random genetic operator to tree structures is

preservation of the grammar. Therefore, the selection of the sub-tree is carefully supervised by this operator, avoiding the construction of invalid individuals that do not satisfy the language derived from the grammar. To this end, the proposed genetic operator restricts sub-tree selection to those sub-trees that form a whole condition. Hence, the new tree maintains the desired structure.

3.4. Updating the genetic probability value

A major feature of this G3P algorithm is its ability to autonomously update the genetic operator probability. This is a well-studied problem that has achieved excellent results so we consider including this interesting concept into the proposed model.

This updating process is based on the fact that a higher exploration is required in situations where the average fitness value is not improving along the generations. In this process, the proposed algorithm calculates the average fitness value from the elite population in a specific generation of the evolutionary process. The resulting average fitness value is compared to the prior value, that is, the average fitness value obtained in the last generation. In situations where the evolutionary process is behaving well and the average fitness value obtained is improving, modification of the parameter values would not be required. On the other hand, a higher genetic probability value is needed if no better solutions are being found, that is, the average fitness value remains the same. Notice that this average value cannot decrease, since new individuals are included in the elitist population if and only if their fitness values are higher than the previous ones.

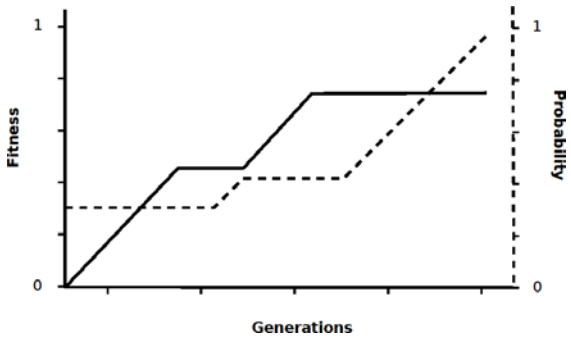


Figure 2. Updating process sample based on the fitness value. The dashed line represents the genetic probability, whereas the solid line indicates the fitness value along the generations

For a better understanding, Figure 2 illustrates a synthetic updating process, demonstrating how the genetic probability value (dashed line) changes in relation to the fitness function value (solid line). In the initial generation of this example, the algorithm generates new individuals based on a specific starting probability, improving the average fitness value so that the genetic operator probability remains the same. In early generations, the average fitness value comes to a standstill, but after some generations, the genetic probability begins to increase, and while this occurs, the fitness value does not improve. At the instant in which the fitness value begins to improve, the algorithm puts a halt to the probability increment.

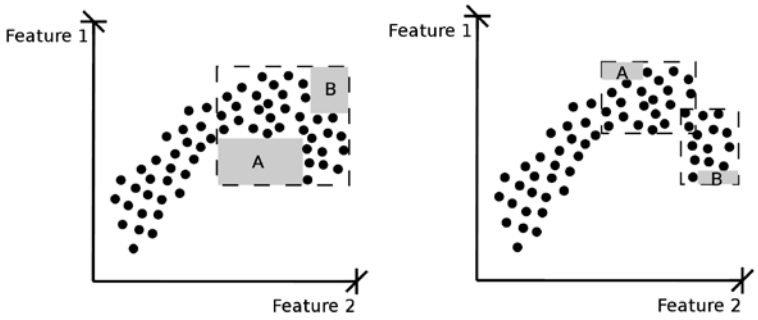
The updating process continues until the genetic probability value reaches the maximum value allowed and the average fitness value does not improve after a specific number of generations. The algorithm then finishes, and the solutions from the elite population are returned to the expert.

3.5. Evaluation process

One of the main processes in any evolutionary model is the evaluation procedure, which determines how promising a certain individual is, that is, how close a given solution is to achieve the aim. The proposed evaluation procedure considers that rules with shared antecedents and consequents are discarded.

As mentioned in previous sections, a major feature of the proposed algorithm is its ability to reduce the gaps in quantitative association rules. To this end, the search process is guided to look for the right width of values, that is, values containing as few spaces which do not comprise any instances as possible. The fitness function F considers the biggest gap within each rule condition, so the size of this gap plays an important role in determining the quality of the condition.

In order to better understand this process, let us consider a synthetic association rule which comprises two quantitative features (features 1 and 2), whose instance distribution is depicted in Figure 3. In situations where the algorithm is designed to discover as many frequent rules as possible, a sample rule could comprise the instances within the dashed line rectangle (Figure 3(a)). In analysing the rectangle formed by this association rule, we discover that there are two main gaps, represented by A and B. Therefore, despite the fact that this association rule covers a high percentage of instances, huge gaps are also comprised by this rule, meaning it would not be



(a) Instances covered by maximising the support (b) Instances covered by minimising the gaps

Figure 3. Sample gaps found within the distribution of the instances covered by sample association rules

as promising as it seemed to be initially. This means that it is not only the discovery of frequent instances that is of interest, but the distribution of these instances also plays an important role.

On the other hand, in analysing Figure 3(b), two different synthetic association rules could be determined to satisfy the same set of instances as the rule depicted in Figure 3(a). However, the gaps included in these rules are smaller, so these rules are of high interest to the user. Discovering these gaps is not a trivial issue, especially for rules whose instances are not accordingly distributed or even for rules with a high number of features. Notice that the search space is smaller for minimum rules, that is, rules that comprise only two features (one in the antecedent and one in the consequent), as shown in Figure 3(b).

The problem of searching for the biggest gap could be hardly addressed from a deterministic point of view since the search space might be intractable. In this regard, we propose the use of a second evolutionary process to find the biggest gap within a region. Thus, we propose the use of a real-coded genetic algorithm in the evaluation process. This way, the goal of this genetic algorithm is to discover the best combination of values for each feature in such a way that they represent the biggest gap. In this sense, we are reducing the problem to optimise the biggest blank space.

Let us consider that the evaluation process analyses the rule *Feature1 [2.5, 3.7]* $\rightarrow$ *Feature2 [5.7, 8.1]*. In this way, the real-coded genetic algorithm included in the evaluation process searches for subranges with a maximum blank space, a sample individual of which is depicted in Figure 4. The individual comprises 4 genes, as two features are to be optimised in this example. The first two genes represent *Feature1*, whereas the last two genes represent *Feature2*. None of the individuals could discover a range of values not comprised within the range determined by the rule.

2.70	3.10	5.73	6.02

Figure 4. Sample genotype for the genetic algorithm included in the evaluation procedure

The genetic algorithm included into the evaluation procedure follows an elitist methodology and uses two well-known genetic operators, the BLX-Alpha crossover and a random mutation. In each generation, the best individual is kept, and this individual is returned if the best result does not improve after 50 generations (this number has been experimentally obtained). The specific number of generations, was obtained in a study, determines that a higher value does not provide better results. Once the best gap is found, the fitness function for the specific association rule is calculated. This fitness function comprises three functions (see Eq. 6), which are described in detail below.

$$fitness(A \rightarrow C) = F_3 + F_2 * F_1 \qquad (6)$$

F_1 is responsible for searching for a set of instances with small gaps (see Eq.7). Therefore, the biggest blank space discovered using the real-coded genetic algorithm is used to determine the interest of the rule, considering its interval width to this end.

$$F_1 = x^2 = \left(1 - \prod_{i=1}^{i=n} \frac{BlankWidth_i}{IntervalWidth_i}\right)^2 \qquad (7)$$

It should also be noted that this function is applied in a quadratic way within the fitness function,

meaning that the smaller the gap within a rule, the better the rule is.

As the goal of any algorithm for mining association rules is the discovery of frequent and reliable association rules, the support and confidence measures must also be considered. Regarding the support measure, it is of interest to discover the most frequent rules possible. However, the higher the support of a rule, the lower is the degree of interest for the user (see Section 2.1). For instance, maximum support values imply misleading rules as stated by the lift, leverage and conviction measures.

Additionally, since the aim of the proposed model is to discover frequent association rules, rules having low support values [27] are not appealing, so a value of zero is assigned to rules that satisfy a low set of instances. In many the proposals for mining frequent association rules, support values lower than 50% are not of interest [13][17][40], and the higher the value, the better it is. We have therefore considered a function (see Figure 5) that reaches the maximum function value with a support close to 50% and decreases the function value when the support is close to the maximum (remember that rules that satisfy most of the instances are not interesting).

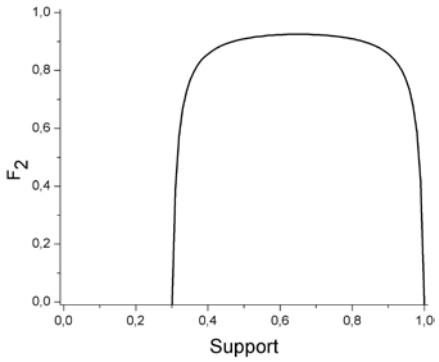


Figure 5. Representation of the F_2 function

The mathematical expression of the above representation of the F_2 fitness function is proposed in Eq. 8, the "x" variable stating for the support value in this function. The aim was to obtain a convex parabola throughout its domain, so using the general equation (ax^2+bx+c) of the parabola, the constant a should be less than 0. Additionally, it was interesting that the parabola cuts the horizontal axis in the values 0.3 and 1, which represent the desired range of support values. Thus, the equation $-10x^2+13x-3$ was obtained, which satisfies all the aforementioned restrictions. Finally, in order to reduce the slope in intermediate values and increase the slope in values close to the boundaries, the aforementioned parabola is divided by a similar one, obtaining the F_2 function.

$$F_2 = \frac{x(13-10x)-3}{x(13-10x)-2.9} \quad (8)$$

Finally, the third function included in the fitness function is related to the confidence measure, an important quality measure in determining the reliability of the rules. Therefore, the higher the confidence value of the rule, the more accurate the rule is. Generally speaking, rules with low confidence values are not of interest to the user. This issue is reinforced by the fact that ARM algorithms usually seek frequent rules, and the confidence value is always greater than the support value.

$$F_3 = x^{10} \quad (9)$$

In this sense, we have defined the F_3 function (see Eq. 9), which states that the higher the confidence of a specific rule, the higher the F_3 value ("x" representing confidence values). Low confidence values imply low function rates, and these values get higher and higher with the increment of confidence values. The power number (10 in this function) was selected after a number of analysis, obtaining that the optimal number was 10 since provide a low function value (close to zero) for confidence values less than 0.8.

As mentioned above, the goal is to maximise the resulting fitness function (Eq. 6), which determines values within the range [0, 2]. In situations where rules comprising only discrete features are discovered, this fitness function discards the F_1 function by using its unity value. Consequently, the evaluation process should be designed with this issue in mind, as no searching for gaps is required. Therefore, the real-coded genetic algorithm is simply carried out in such situations where at least one numerical feature is considered within the rule.

4. Experimental study

This section presents the data sets, the algorithms employed in the experimental study, and their parameter configuration. Finally, a number of experiments were carried out, demonstrating the proposed algorithm's effectiveness in reducing gaps in quantitative association rules.

4.1. Data sets

To analyse the effectiveness of the proposed algorithm, a varied set of data from the well-known UCI (University of California, Irvine) machine learning repository were considered. The main features of these data sets are depicted in Table 1, where data are arranged from the lowest to highest number of instances.

Table 1. Data sets characteristics

DATASET	INSTANCES	ATTRIBUTES	
		Cont.	*Nom.*
Zoo	102	0	17
Lymphography	148	3	16
Wisconsin Prognostic	194	33	1
Sonar	208	60	1
Primary-tumour	339	0	18
Automobile	392	8	0
Wisconsin Diagnostic	569	30	1
Soybean	683	0	36
Australian	690	6	10
Vowel	990	10	4
Credit-g	1000	6	15
Izmir Weather	1461	10	0
Contraceptive	1473	2	8
Ankara Weather	1608	8	0
Segment	2310	19	1
Splice	3190	0	61
Chess	3196	0	37
Nursery	12960	0	9
House 16H	22784	17	0
Connect-4	67557	0	43

4.2. Algorithms and experimental set-up

For a fair comparison to be drawn, the experiments were split into two separate parts, depending on whether the algorithms work only on discrete features or any type of feature. The algorithms that worked on any type of feature were our model, G3PARM [17] and Quant-Miner [28]. As for nominal features, exhaustive search algorithms such as Apriori [2] and FP-Growth [9], together with the evolutionary algorithm GBAP-ARM [21] were considered in the experimental stage. The Apriori algorithm used in this experimental stage is the version available in the WEKA machine learning software. The FP-Growth algorithm is available for download from the webpage of the Department of Computer Science of the University of Liverpool[1]. The remaining algorithms are the original algorithms provided by the authors. Finally, it is worth noting that we have considered the optimal parameter given by the original articles where the algorithms are described.

For the G3PARM algorithm, the optimal values were used - a population of 50 individuals, a maximum number of 100 generations, probabilities of 70% and 14% for the crossover and mutation operators respectively, a maximum derivation number of 24, an external population size of 20, a 90% confidence threshold, and a 70% support threshold.

The optimal parameters of the Quant-Miner algorithm are a population size of 250 individuals, 100 generations, 40% mutation probability, and 50% crossover probability. The support and confidence thresholds considered in this algorithm are 90% and 70%, for confidence and support, respectively.

For the GBAP-ARM algorithm, the parameter configuration corresponds to a population size of 20 ants, 100 generations, a maximum number of 10 derivations, an initial and maximum amount of pheromone of 1.0, a minimum amount of pheromone equal to 0.1, an evaporation rate of 0.05, a value of 0.4 for the α exponent, a value of 1.0 for the β exponent, a 70% support threshold, and a maximum size for the set of rules returned by 20. It is worth noting that that this algorithm does not require a confidence threshold.

Apriori and FP-Growth used the same support and confidence thresholds as the other algorithms so that a fair comparison could be drawn. Moreover, since both algorithms are exhaustive search methods, they discover any rule that satisfies the aforementioned thresholds. We have therefore determined the maximum number of rules they can extract, limiting this number to 2,000,000, so it is large enough to analyse how these algorithms behave.

The proposed G3P algorithm reduces the number of parameters significantly, especially in relation to evolutionary algorithms in the ARM field. The algorithm here proposed only requires the number of rules to be mined. In order to make a fair comparison, this number of rules is set to 20 as the other algorithms. The remaining parameters self-adapt along the evolutionary process.

[1] http://cgi.csc.liv.ac.uk/~frans/KDD/Software/FPgrowth/fpGrowth.html.

4.3. Running example

In this section we provide two samples of running the proposed algorithm, which enables a better understanding about how the algorithm behaves. We have used the *Automobile Performance* dataset, which includes 8 attributes defined in a continuous domain. The algorithm is run by fixing the number of rules to be discovered to a value of 5 rules. Table 2 shows the resulting set of rules.

As it is depicted, the proposed algorithm is able to discover highly reliable quantitative association rules. All of the rules discovered have a confidence value above 98%, so the rules are highly probable to be satisfied if their antecedents are previously satisfied. Analysing the interestingness measures (lift, leverage and conviction), the results reveal that the rules discovered are of high interest to the user. All of the rules provide a lift value greater than the unity, and a high conviction value. As for the leverage quality measure, no rule provides a negative value, indicating that the rules are interesting. Finally, if we analyse the results obtained for the support measure, it is stated that the rules discovered are very frequent. They satisfy around 60% and 70% of the instances.

4.4. Analysis of the behaviour of the updating process

The aim of this section is to demonstrate how the algorithm behaves when the process is applied to real data, paying special attention to the probability updating process. In this sense, a number of experiments are carried out with different probability values and different numbers of rules to be discovered. The goal of this section is not to compare whether the updating process performs better than a fixed probability value, since it is a well-studied area. On the contrary, we want to demonstrate that the starting probability value does not change the fitness function value obtained but the time required to reach the value.

The aforementioned probability value denotes the starting value for the genetic operator, which self-adapts this value based on the algorithm's behaviour. Figure 6 relates the average fitness value (solid line) and the probability value of the genetic operator (dashed line). The increment in the number of rules to be discovered softens the trend of the fitness value along the generations, meaning that a lower number of rules give rise to a sharp increase regardless of the starting probability value.

This behaviour makes sense, as improvements in the fitness value of one rule in a set of 5 gives rise to a higher average fitness value than that obtained in a set of 20 rules. Also, an increment in the number of rules to be discovered requires a higher number of generations to reach the optimum value. This explains why the ranges of the horizontal axes differ. It should be noted that there is not a fixed number of generations but a stopping criterion that is based on the improvement of the average fitness value of the elitist population.

Table 2. Sample running and resulting set of rules discovered by looking for 5 rules

Rules and quality measure values
IF mpg IN [18.55, 40.88] **THEN** acceleration IN [10.76, 24.75]
Support: 0.648
Confidence: 0.996
Lift: 1.531
Leverage: 0.255
Conviction: 89.770
IF displacement IN [68.79, 177.31] **THEN** acceleration IN [10.76, 24.75]
Support: 0.556
Confidence: 0.995
Lift: 1.782
Leverage: 0.244
Conviction: 97.209
IF cylinders IN [3.03, 6.35] **THEN** acceleration IN [10.76, 24.75]
Support: 0.724
Confidence: 0.996
Lift: 1.371
Leverage: 0.196
Conviction: 78.520
IF horsepower IN [76.42, 168.13] **THEN** mpg IN [12.61, 40.76]
Support: 0.648
Confidence: 0.988
Lift: 1.525
Leverage: 0.223
Conviction: 30.158
IF horsepower IN [74.08, 155.70] **THEN** acceleration IN [10.76, 24.75]
Support: 0.676
Confidence: 0.993
Lift: 1.468
Leverage: 0.216
Conviction: 43.251

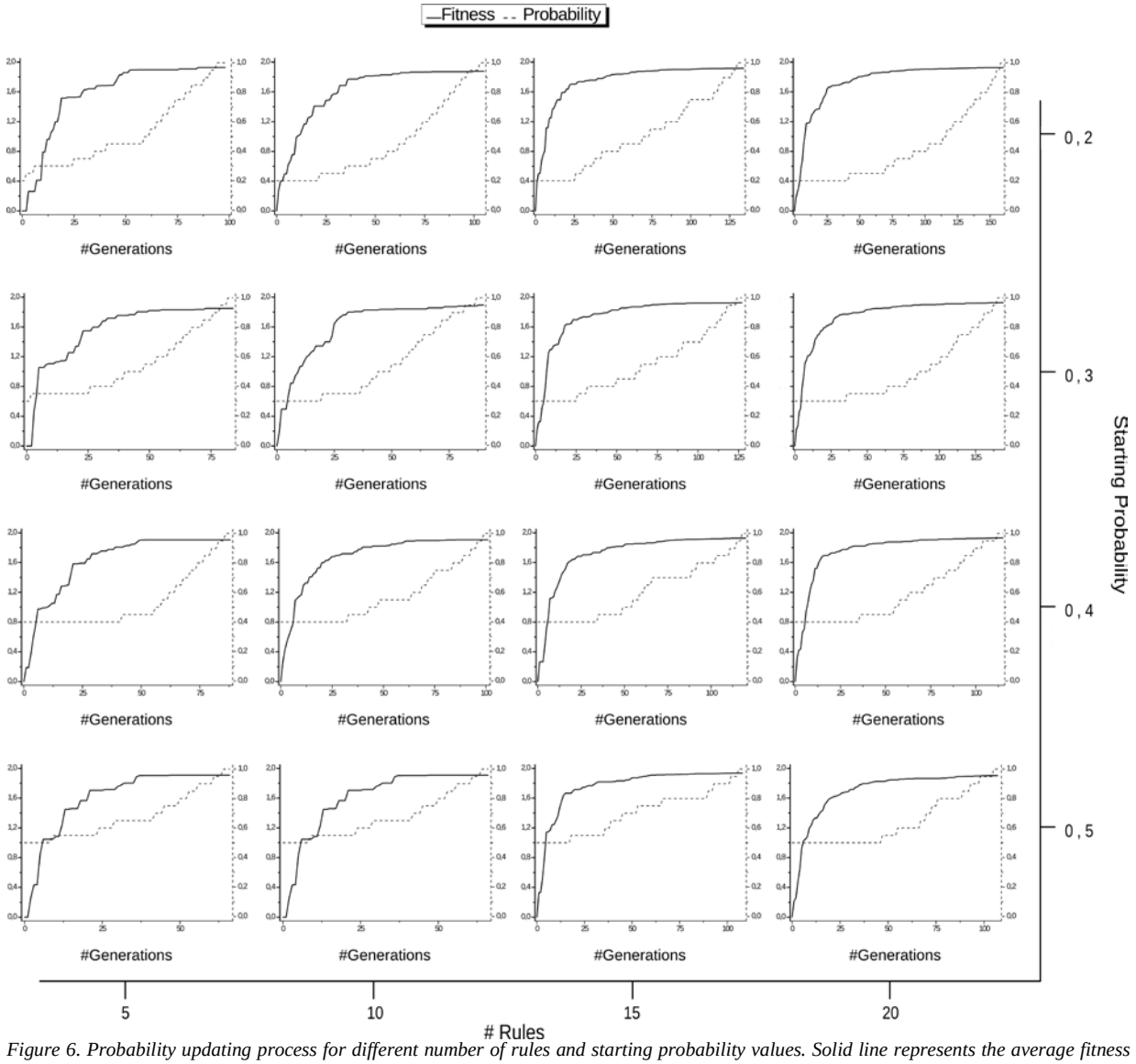


Figure 6. Probability updating process for different number of rules and starting probability values. Solid line represents the average fitness value, whereas the probability value of the genetic operator is represented by the dashed line.

The most interesting part of this study is the analysis of the algorithm's behaviour for different starting probability values. As shown in Figure 6, there is no particular improvement. The algorithm automatically adapts its probability value along the generations, obtaining similar fitness values regardless of the starting probability value. However, it is worth noting that despite the resulting average fitness value remaining the same, the number of generations required to achieve this value increases while the starting probability decreases.

This study explains how the algorithm behaves similarly, regardless of the used parameter value, meaning that its self-adaptation is excellent, providing the same results for different starting probability values. This analysis could therefore state that any starting probability value could be suitable for obtaining the optimal solutions, but a probability of 0.5 enables the number of generations and, consequently, the execution time required by the algorithm to be reduced. However, this value is not mandatory, as the algorithm behaves similarly when other values are used.

4.5. Analysis of the size of the gaps

This proposed algorithm, together with Quant-Miner and G3PARM, is executed on real data and the resulting association rules are analysed. The aim is to demonstrate its ability to mine quantitative association rules having small gaps. In order to draw a fair comparison between the three algorithms, only those association rules that comprise the same two features (horsepower and mpg) from the *Automobile Performance* data set are considered.

The G3PARM algorithm discovers the rule *IF (horsepower <= 190) THEN (mpg < 39.08).* The distribution of the instances satisfied by this rule is properly depicted in Figure 7. As shown, G3PARM mines rules having huge gaps, and the covered instances are not well-grouped. Quantitative association rules discovered with this algorithm comprise four logic operators: <, <=, > and >=. It aims to maximise the support quality measure, obtaining high support and confidence values (0.939 and 0.973). Nevertheless, an analysis of the distribution of the instances is not carried out in the mining process, so the rules discovered by G3PARM are not of interest as stated by the quality measures (a lift value of 0.999, and conviction and leverage values of 0). Hence, it is possible to state that G3PARM does not consider the distribution in its mining process, so its rules are not interesting since they do not describe well the data information.

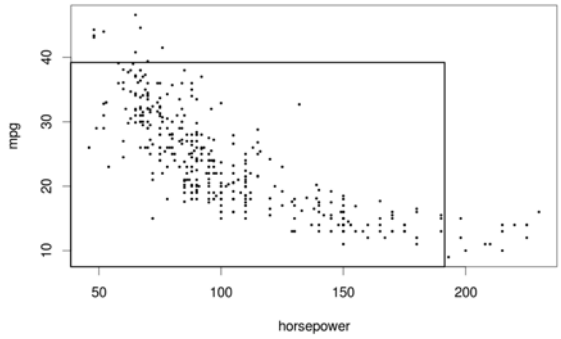


Figure 7. Representation of the instances covered by the rule IF (horsepower <= 190) THEN (mpg < 39.08)

As for the Quant-Miner algorithm, two quantitative rules that comprise the aforementioned attributes are considered: (1) *IF horsepower IN [49.0; 125.0] THEN mpg IN [16.0; 41.5]*; and (2) *IF horsepower IN [65.0; 145.0] THEN mpg IN [15.0; 37.3]*. As shown in Figure 8, the distribution of the instances is better grouped than in G3PARM. Thus, the size of the gaps is smaller and the description of the instances is much more informative than in G3PARM. This issue is transferred to the quality measure values. Analysing the first rule, it obtains a support value of 0.717, a confidence value of 0.972, a lift value of 1.221, a leverage value of 0.130, and a value of 10.216 for the conviction measure. As for the second rule, it obtains a support value of 0.714, a confidence value of 0.948, a lift value of 1.155, and the values of 0.095 and 5.586 for leverage and conviction, respectively. Thus, the rules obtained by Quant-Miner are less representative, but of higher interest, and this is the result of a better searching process of the instance distribution.

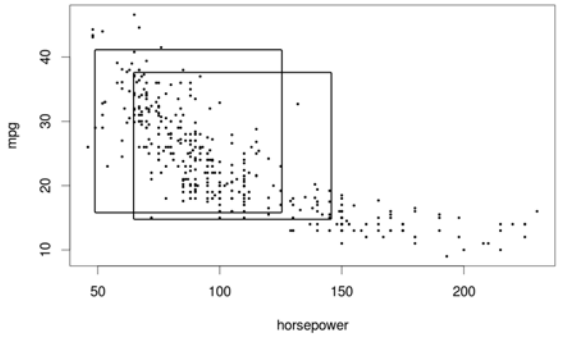


Figure 8. Representation of the instances covered by the rules IF (horsepower IN [49, 125]) THEN (mpg IN [16.0, 41.5]) *and* IF (horsepower IN [65.0, 145.0]) THEN (mpg IN [15.0, 37.3])

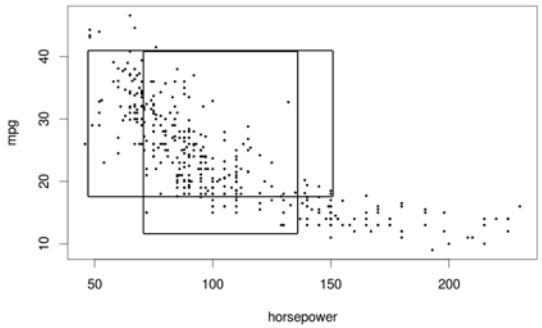


Figure 9. Representation of the instances covered by the rules IF (horsepower IN [71.8, 136.5]) THEN (mpg IN [12.6, 40.8]) *and* IF (mpg IN [18.5, 40.9]) THEN (horsepower IN [46.2, 150.9])

Finally, our model discovers two association rules whose instances are quite well-grouped (see Figure 9): (1) *IF horsepower IN [71.8; 136.5] THEN mpg IN [12.6; 40.8]*; and (2) as *IF mpg IN [18.5; 40.9]*

THEN horsepower IN [46.2; 150.9]. As depicted in Figure 13, no huge gaps could be found within the instances covered by these rules. This makes the rules highly interesting, and the confidence, lift, leverage and conviction values confirm this statement. These values, for the first rule are 0.996, 1.719, 0.242 and 105.25, respectively; whereas for the second rule are 0.992, 1.531, 0.224 and 44.38.

In comparing the distribution for the instances, both Quant-Miner and the proposed model are the algorithms that best group these spaces. These two algorithms obtain numerical intervals that properly represent the set of instances and their rules are of higher interest, as illustrated by analysing the quality measures. On the contrary, G3PARM provides intervals with no descriptive values, since the instances covered are not well-grouped. Regarding the proposed algorithm, the mined rules have better values for the confidence, lift, leverage and conviction measures, obtaining values close to the maximum for these quality measures.

Table 3. Results obtained (presented in a per unit basis). Bold type values indicate the algorithm that attains the best result.

	Support			Confidence			Lift		
Data set	G3PARM	Quant-Miner	Proposal	G3PARM	Quant-Miner	Proposal	G3PARM	Quant-Miner	Proposal
Zoo	0.780	---	**0.898**	**0.973**	---	0.950	1.055	---	**1.058**
Lymphography	**0.961**	0.818	0.605	**1.000**	0.954	0.973	1.020	1.121	**1.751**
Wisconsin Prg.	**0.952**	0.808	0.785	**1.000**	0.965	**1.000**	1.007	1.134	**1.293**
Sonar	**1.000**	0.712	0.618	**1.000**	0.931	0.949	1.000	1.753	**2.079**
Prim. tumour	0.872	---	**0.980**	**0.993**	---	0.989	1.008	---	**1.009**
Automobile	**0.858**	0.703	0.594	**0.993**	0.982	**0.993**	1.013	1.568	**1.766**
Wisconsin Dig.	**0.982**	0.697	0.774	**1.000**	0.992	0.995	1.002	**1.421**	1.357
Soybean	0.934	---	**0.974**	0.987	---	**0.993**	1.005	---	**1.019**
Australian	**0.984**	0.809	0.653	**0.998**	0.960	0.997	1.001	1.252	**1.677**
Vowel	**0.944**	0.700	0.713	0.981	0.949	**0.994**	1.002	**1.587**	1.517
Credit-g	0.923	0.740	**0.999**	0.995	0.982	**0.999**	1.001	**1.487**	1.001
Izmir Weather	**0.958**	0.693	0.672	**0.999**	0.956	0.972	1.001	1.351	**1.598**
Contraceptive	0.889	**0.921**	0.657	**0.991**	0.979	0.928	1.002	1.001	**1.497**
Ankara	**0.940**	0.792	0.772	0.999	0.961	**1.000**	1.004	**1.422**	1.232
Segment	**0.998**	0.700	0.734	**1.000**	0.975	**1.000**	1.000	**1.473**	1.399
Splice	**0.999**	---	0.899	**1.000**	---	**1.000**	1.000	---	**1.211**
Chess	0.919	---	**0.993**	**0.998**	---	**0.998**	1.001	---	**1.005**
Nursery	**0.759**	---	0.532	0.990	---	**1.000**	1.003	---	**1.549**
House 16H	**0.958**	0.752	0.794	**0.999**	0.932	**0.999**	1.000	1.321	**1.349**
Connect-4	0.926	---	**0.999**	**1.000**	---	0.999	**1.001**	---	**1.001**
Ranking	**1.350**	2.550	2.100	**1.400**	2.950	1.650	2.550	2.150	**1.300**

	Leverage			Conviction		
Data set	G3PARM	Quant-Miner	Proposal	G3PARM	Quant-Miner	Proposal
Zoo	**0.052**	---	0.048	2.214	---	**2.379**
Lymphography	0.012	0.104	**0.189**	Infinity	5.278	**20.994**
Wisconsin Prg.	0.009	0.110	**0.158**	Infinity	**27.576**	Infinity
Sonar	0.000	0.103	**0.234**	Infinity	6.874	**10.852**
Prim. tumour	**0.012**	---	0.009	**20.455**	---	2.140
Automobile	0.015	0.183	**0.219**	33.811	37.771	**45.542**
Wisconsin Dig.	0.003	**0.172**	0.144	Infinity	**57.310**	41.187
Soybean	0.007	---	**0.018**	4.991	---	**5.209**
Australian	0.002	0.121	**0.181**	1.102	65.110	**79.639**
Vowel	0.008	**0.189**	0.162	2.313	**66.442**	65.460
Credit-g	0.001	**0.162**	0.001	3.982	**18.942**	4.107
Izmir Weather	0.001	0.154	**0.201**	1.102	27.329	**33.871**
Contraceptive	0.005	0.003	**0.166**	1.374	1.199	**5.083**
Ankara	0.002	**0.166**	0.102	1.561	**33.651**	23.387
Segment	0.000	**0.201**	0.181	Infinity	**32.653**	Infinity
Splice	0.000	---	**0.092**	Infinity	---	Infinity
Chess	**0.026**	---	0.005	**29.661**	---	8.699
Nursery	0.001	---	**0.142**	**1.655**	---	Infinity
House 16H	0.000	0.129	**0.131**	1.105	102.233	**132.701**
Connect-4	**0.013**	---	0.001	Infinity	---	**4.107**
Ranking	2.400	2.125	**1.475**	2.425	2.000	**1.575**

Finally, it should be noted that none of the ARM algorithms analysed in this study describe the correlation of the instances distribution. The goal of the analysed ARM algorithms is to provide a highly representative set of instances that provide useful knowledge to the users.

4.6. Analysis of the quality measures

A larger study was carried out, considering 20 data sets and comparing the resulting average values (see Tables 3 and 4) for the five aforementioned quality measures and two additional measures (number of attributes per rule and interval width) using 10 different executions. Note that, first, the average results per data set are computed, and the last row of each table is the ranking obtained by each algorithm. Bold type values indicate the algorithm that attains the best result for a specific data set. Results marked with "---" point out that no rules were obtained. For instance, Quant-Miner is not able to discover any rule in situations where no numerical attribute is considered. However, this algorithm does enable nominal features to be discovered in situations where at least one numerical attribute is considered.

In order to analyse these results statistically [7], the Iman-Davenport test was performed. The computed value for the Iman-Davenport statistic for the average support distributed according to a *F* distribution is equal to 11.039, for the confidence measure, its value is equal to 42.788, 13.067 for the lift measure, 5.536 for leverage, 4.188 for the conviction measure, 9.464 for the number of rules, and 5.755 for the interval width. None of the values fall within the critical interval at the $\alpha = 0.05$ significance level, which is 3.245. Therefore, the null-hypothesis that all algorithms perform equally well is rejected for all of the measures considered. Following this, a post-hoc test is used in order to find out whether the proposed algorithm presents statistical differences with regard to the remaining algorithms (see Table 5). In this study, we therefore proceed with the Wilcoxon signed-rank test which assumes that the performance of the proposed algorithm considering the level of significance $\alpha = 0.05$.

The Wilcoxon test reveals that the proposed model performs statistically better than G3PARM when the interest measures are considered (lift, leverage and conviction). This behaviour was expected since G3PARM does not consider any quality measure but the mining of high frequent and reliable rules. Besides, it does not carry out any consideration about the distribution of the instances satisfied by the rules. Additionally, the proposed algorithm obtains a lower number of rules and a smaller interval width.

Table 4. Average number of attributes and interval width (percentage with regard to the whole range of values). Bold type values indicate the algorithm that attains the best result.

	Number of attributes			Interval width (percentage)		
Data set	G3PARM	Quant-Miner	Proposal	G3PARM	Quant-Miner	Proposal
Zoo	3.07	---	**2.00**	---	---	---
Lymphography	2.57	**2.32**	2.55	80.23	72.22	**45.31**
Wisconsin Prg.	2.87	2.93	**2.00**	75.26	57.31	**52.23**
Sonar	2.53	2.24	**2.05**	85.82	70.54	**55.68**
Prim. tumour	**2.26**	---	2.65	---	---	---
Automobile	2.78	**2.14**	2.59	75.26	**60.10**	64.02
Wisconsin Dig.	2.95	**2.17**	2.35	59.27	**30.27**	31.81
Soybean	2.65	---	**2.13**	---	---	---
Australian	**2.12**	2.20	2.20	86.20	72.42	**65.58**
Vowel	2.30	**2.20**	**2.20**	84.54	**63.24**	67.28
Credit-g	2.59	2.30	**2.00**	78.15	**61.86**	80.22
Izmir Weather	2.81	**2.13**	2.41	73.19	54.66	**53.56**
Contraceptive	2.58	2.26	**2.10**	75.72	77.42	**68.42**
Ankara	2.82	2.34	**2.00**	76.06	**68.11**	70.26
Segment	2.58	2.42	**2.12**	79.72	62.02	**61.66**
Splice	2.42	---	**2.04**	---	---	---
Chess	2.99	---	**2.30**	---	---	---
Nursery	2.12	---	**2.00**	66.45	---	**35.76**
House 16H	2.80	2.34	**2.00**	70.73	**60.55**	62.54
Connect-4	3.05	---	**2.00**	---	---	---
Ranking	2.450	2.200	**1.350**	2.550	1.800	**1.650**

Table 5. Statistical analysis of the results when comparing the proposed model to Quant-Miner and G3PARM

	Wilcoxon signed-rank test $\alpha = 0.05$	
Measure	Quant-Miner	G3PARM
Support	Accept	Accept
Confidence	Accept	Reject
Lift	Reject	Accept
Leverage	Reject	Accept
Conviction	Reject	Accept
Number rules	Accept	Accept
Interval width	Reject	Accept

As for the Quant-Miner algorithm, just the support and confidence measures are statistically better for the proposed algorithm. However, despite the fact that it is not possible to statistically state that the proposed model performs better than Quant-Miner for the remain quality measures (lift, leverage and conviction), the ranking values depicted in Table 3 show that our algorithm is better than Quant-Miner in all of the five measures. Besides, Quant-Miner is not able to discover association rules in all the datasets, just in those having at least one numerical attribute. Finally, considering the number of rules and the interval width, our proposal behaves statistically better than Quant-Miner for the number of rules, as revealed by the Wilcoxon test.

4.7. Analysis of the nominal datasets

In order to compare the behaviour of the proposed algorithm with respect to algorithms that only discover rules in discrete domains, a study over the four nominal data sets is carried out (see Tables 6 and 7). Since FP-Growth and Apriori are exhaustive search approaches, a statistical test here is meaningless (the number of rules are very dissimilar), and only the average results have sufficient levels of significance.

Table 6. Average values obtained for the support quality measure

Data set	FP-Growth	Apriori	GBAP-ARM	Proposal
Primary-tumour	0.757	0.757	0.973	0.980
Soybean	0.730	0.730	0.968	0.974
Chess	0.759	0.759	0.988	0.993
Connect-4	0.917	0.917	0.998	0.999

Table 7. Average values obtained for the confidence measure

Data set	FP-Growth	Apriori	GBAP-ARM	Proposal
Primary-tumour	0.942	0.942	0.989	0.989
Soybean	0.886	0.886	0.989	0.993
Chess	0.939	0.939	0.995	0.998
Connect-4	0.976	0.976	0.999	0.999

The results of FP-Growth are the average values from 209, 112,650, 2,000,000 and 2,000,000 rules that correspond to the datasets Primary-tumour, Soybean, Chess and Connect-4 respectively. As for the Apriori algorithm, the results are the average values from 209, 47,304, 2,000,000 and 2,000,000 rules for the same data sets. It is also worth noting that only the support and confidence measures are considered in this study. These algorithms do not take additional quality measures into account.

4.8. Computational cost and complexity of our proposal

In this experimental stage, a computational complexity analysis has been carried out to determine the efficiency of the proposed model. In this sense, each of the main procedures are analysed separately: evaluator and genetic operator. Then, the computational complexity of the whole algorithm is determined. Finally, we provide some execution tie

Firstly, the evaluator procedure depends on the number of individuals (N_{ind}), instances (N_{ins}) and attributes (N_{att}). Mathematically, this complexity order is defined as $O(N_{ind} \times N_{ins} \times N_{att})$. Additionally, if at least one attribute is numerical, then an additional evolutionary process is carried out to obtain the biggest gap within each solution. This second evolutionary process acts as a sub-process that depends on the number of individuals ($N_{ind\_subproces}$), instances ($N_{ins\_subprocess}$) and attributes ($N_{att\_subprocess}$). Thus, when numerical attributes are considered, the complexity order is defined as $O(N_{ind} \times N_{ins} \times N_{att} \times N_{ind\_subproces} \times N_{ins\_subprocess} \times N_{att\_subprocess})$.

Analysing the computing requirements for each procedure, it is stated that both N_{ind} and $N_{ind\_subprocess}$ are previously fixed, so they are considered as constant and its complexity order is O(1). Additionally, all the procedures are repeated as times as the predefined number of generations, which is also a constant value. Therefore, bearing in mind all these issues, the resultant computational complexity of the proposed model is stated as $O(N_{ins} \times N_{att})$ in case

that no numerical attribute is considered, and as $O(N_{ins} \times N_{att} \times N_{ins\_subprocess} \times N_{att\_subprocess})$ in such a situation where at least one numerical attribute is considered. Thus, the complexity of the proposed approach is linear with regard to the number of instances and attributes.

To determine the computational cost of the proposed model, we run the algorithms by using the most complex dataset, i.e. House 16H, the one with the highest number of instances and comprising continuous attributes. Using the dataset as it is, i.e. without a discretization of the attributes, the proposed model requires 1,260 seconds, whereas G3PARM and Quant-Miner require 147 and 128 seconds, respectively. If we discretize each continuous attribute into 4 equal-width intervals, then exhaustive search algorithms (FP-Growth and Apriori) could be applied, requiring 900 and 3,268 seconds, respectively. On the contrary, Quant-Miner, G3PARM and our proposal require, for the same discretized dataset, 111, 57 and 746 seconds, respectively. As it was expected, the new proposal require a higher computation time than existing evolutionary algorithm, since it includes a second evolutionary computation system as part of the evaluation function. Nevertheless, the results obtained are so promising that it is meaningless to require a higher time in running it.

5. Concluding remarks

In this article we have presented an interesting algorithm for reducing gaps in quantitative association rules. The main feature of this algorithm is the employment of an interesting fitness function to group the patterns and avoid misleading intervals in the mined rules. Additionally, we have made use of previous research studies to reduce the number of parameter to be tuned in the evolutionary algorithm, being a great advantage for non-expert users.

We have paid special attention to the fitness function, which has been defined to reduce gaps in numerical intervals. Therefore, it is not only the discovery of frequent and reliable association rules that is of interest, but also the extraction of rules that properly describe the behaviour of instance distribution. Further, the reduction of the number of parameters required by the proposed algorithm is a really good advantage. Many evolutionary algorithms in the ARM field have a number of parameters that should be established beforehand, which may require previous knowledge. This makes the process quite difficult for non-expert users. The algorithm proposed in this paper is a self-adaptive proposal, which does not require as many configuration parameters as other proposals.

Finally, experimental results demonstrate that this algorithm not only extracts rules of high interest, according to five quality measures, but it also discovers frequent and reliable association rules, having smaller gaps in the space of instances covered. In future, we would like to deal with ordinal attributes that include more information than the nominal. The proposed algorithm behaves better than exhaustive search proposals like Apriori and FP-Growth, especially for support and confidence quality measures. As for evolutionary algorithms for mining quantitative association rules, the proposed model obtain significant statistical differences with regard to G3PARM and Quant-Miner for most of the seven metrics used in the experimental study. Finally, it should be noted that the new model requires a higher computational time since it includes a second evolutionary process in the evaluation stage. Nevertheless, the computational cost spent is not very high in comparison to the promising results obtained for the quality measures.

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] H. Adeli, and S. L. Hung. Machine Learning - Neural Networks, Genetic Algorithms, and Fuzzy Sets, John Wiley and Sons, New York, 1995.

[2] 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 (VLDB'94), San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 1994, pp. 487-499.

[3] B. Alatas and E. Akin. An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules. Soft Computing, 10, pp. 230–237, 2006.

[4] B.R. Campomanes-Álvareza, O. Cordón and S. Damasa. "Evolutionary Multi-Objective Optimization for Mesh Simplification of 3D Open Models," Integrated Computer-Aided Engineering, 20:4, 2013, 375-390.

[5] T. Chabuk, J. A. Reggia, J. Lohn, and D. Linden. "Causally-Guided Evolutionary Optimization and its Application to Antenna Array Design," Integrated Computer-Aided Engineering, 19:2, 2012, 111-124.

[6] E. Datar, M. Fujiwara, S. Gionis, A. Indyk, P. Motwani, R. Ullman, J. D. Yang, and C. Cohen. Finding interesting associations without support pruning. IEEE Transactions on Knowledge and Data Engineering, 13(1), pp. 64–78, 2001.

[7] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research, 7, 2006, pp. 1-30.

[8] C. Fuggini, E. Chatzi, D. Zangani, and T.B. Messervey. "Combining Genetic Algorithm with a Meso-scale Approach for System Identification of a Smart Polymeric Textile," Computer-Aided Civil and Infrastructure Engineering, 28:3, 2013, pp. 227-245.

[9] J. Han, J. Pei, Y. Yin and R. Mao. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Mining and Knowledge Discovery, 8, 2004, pp. 53-87.

[10] R. I. Hoai, N. X. Whigham, P. A. Shan, Y. O'neill, M. McKay, Grammar-based genetic programming: A survey, Genetic Programming and Evolvable Machines 11 (3-4), 2010, pp. 365–396.

[11] F. Y. Hsiao, S. S. Wang, W.C. Wang, C.P. Wen and W. D. Yu. "Neuro-Fuzzy Cost Estimation Model Enhanced by Fast Messy Genetic Algorithms for Semiconductor Hookup Construction," Computer-Aided Civil and Infrastructure Engineering, 2012, 27:10, pp. 764-781.

[12] X. Jiang and H. Adeli. Neuro-Genetic Algorithm for Nonlinear Active Control of Highrise Buildings. International Journal for Numerical Methods in Engineering, 75 (8), 2008. pp. 770-786.

[13] C.S.Kanimozhi Selvi and A.Tamilarasi. An automated association rule mining technique with cumulative support thresholds. Int. J. Open Problems in Compt. Math, 2(3), 2009, pp. 427-438.

[14] H. Kim, and H. Adeli. "Discrete Cost Optimization of Composite Floors using a Floating Point Genetic Algorithm", Engineering Optimization, Vol. 33, No. 4, 2001, pp. 485-501.

[15] D.T. Larose, Discovering Knowledge in Data: An Introduction to Data Mining, Wiley, 2005.

[16] N. Lavrac, P.A. Flach and B. Zupan. Rule evaluation measures: a unifyingview. Proceedings of the 9th International Workshop on Inductive Logic Programming. London, UK, 1999, pp. 174-185.

[17] J.M. Luna, J.R. Romero and S. Ventura, Design and behaviour study of a grammar guided genetic programming algorithm for mining association rules, Knowledge and Information Systems 32(1), 2012, pp. 53-76.

[18] 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 on Genetic Programming, EuroGP2013, pages 121-132, Vienna, Austria, 2013.

[19] M. Martínez-Ballesteros, F. Martínez-Álvarez, A. Troncoso and J. C. Riquelme. Mining quantitative association rules based on evolutionary computation and its application to atmospheric pollution. Integrated Computer-Aided Engineering, 17, 2010, pp. 227-242.

[20] J. Niehaus and W. Banzhaf. Adaption of operator probabilities in genetic programming. Proceedings of EuroGP'2001, vol. 2038, pp. 325-336, Lake Como, Italy, 2002.

[21] J.L. Olmo, J.M. Luna, J.R. Romero and S. Ventura. Mining association rules with single and multi-objective grammar guided ant programming. Integrated Computer Aided Engineering, 20 (3), pp. 217-234, 2013. DOI:10.3233/ICA-130430.

[22] C. Ordoñez, N. Ezquerra, and C. Santana. Constraining and summarizing association rules in medical data. Knowledge and Information Systems, 9(3), 2006, pp. 259–283.

[23] E.C. Pedrino, V.O. Roda, E. R. R. Kato, J. H. Saito, M.L. Tronco, R. H. Tsunaki, O. Morandin, and M. C. Nicoletti. "A Genetic Programming Based System for the Automatic Construction of Image Filters," Integrated Computer-Aided Engineering, 20:3, 2013, pp. 275-287

[24] G. Piatetsky-Shapiro. Discovery, analysis and presentation of strong rules. Knowledge Discovery in Databases, G. Piatetsky-Shapiro and W. Frawley, Eds. AAAI Press, 1991, pp. 229-248.

[25] R. Putha, L. Quadrifoglio, and E. Zechman. "Comparing Ant Colony Optimization and Genetic Algorithm Approaches for Solving Traffic Signal Coordination under Oversaturation Conditions," Computer-Aided Civil and Infrastructure Engineering, 27:1, 2012, pp. 14-28.

[26] A. Rahman, C.I. Ezeife, and A.K. Aggarwal. Wifi miner: An online apriori-infrequent based wireless intrusion system. In Proceedings of the 2nd International Workshop in Knowledge Discovery from Sensor Data, Sensor-KDD '08, pp. 76–93, Las Vegas, USA, 2008.

[27] C. Romero, J.M. Luna, J.R. Romero and S. Ventura. Mining rare association rules from e-learning data. In Proceedings of the 3rd International Conference on Educational Data Mining, EDM 2010, pp. 171-180, Pittsburgh, PA, USA.

[28] A. Salleb-Aouissi, C. Vrain, C. Nortet. Quantminer: A genetic algorithm for mining quantitative association rules, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyberadad, India, 2007, pp. 1035–1040.

[29] D. Sánchez, J. M. Serrano, L. Cerda, and M. A. Vila. Association rules applied to credit card fraud detection. Expert systems with applications, (36), pp. 3630–3640, 200.

[30] K.C. Sarma and H. Adeli. Life-Cycle Cost Optimization of Steel Structures. International Journal for Numerical Methods in Engineering, 55 (12), 2002. pp. 1451-1462.

[31] K. C. Sarma, and H. Adeli. "Bi-Level Parallel Genetic Algorithms for Optimization of Large Steel Structures", Computer-Aided Civil and Infrastructure Engineering, 2001, 16 (5), pp. 295-304.

[32] T. Scheffer. Finding association rules that trade support optimally against confidence. In Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery 2001, pp. 424-435, Freiburg, Germany.

[33] L. Sgambi, K.. Gkoumas F. and Bontempi. "Genetic Algorithms for the Dependability Assurance in the Design of a Long Span Suspension Bridge," Computer-Aided Civil and Infrastructure Engineering, 2012, 27:9, pp. 655-675.

[34] Y. Shafahi and M. Bagherian. "A Customized Particle Swarm Method to Solve Highway Alignment Optimization Problem," Computer-Aided Civil and Infrastructure Engineering, 2013, 28:1, pp. 52-67.

[35] N. Siddique and H. Adeli. Computational Intelligence - Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing, Wiley, West Sussex, United Kingdom, 2013

[36] P. Tan and V. Kumar. Interestingness measures for association patterns: A perspective. In Proceedings of the Workshop on Postprocessing in Machine Learning and Data Mining, KDD '00, New York, USA.

[37] S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás. JCLEC: A java framework for evolutionary computation. Soft Computing, 12(4), 2008, pp.381–392.

[38] X. Yan, C. Zhang, S. Zhang, ARMGA: Identifying interesting association rules with genetic algorithms, Applied Artificial Intelligence 19 (7), 2005, pp. 677–689.
[39] X. Yan, C. Zhang and S. Zhang. Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support. Expert Systems with Applications, 36 (2), 2009, pp. 3066-3076.
[40] Q. Zhao and S.S. Bhowmick. Association Rule Mining: A Survey. Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003116 , 2003.