Evolutionary Extraction of Association Rules: A Preliminary Study on their Effectiveness

Nicolò Flugy Papè[1], Jesús Alcalá-Fdez[2], Andrea Bonarini[1], and Francisco Herrera[2]

[1] Dipartimento di Elettronica e Informazione, Politecnico di Milano, 20133 Milano, Italy
nicolo.flugy@mail.polimi.it, bonarini@elet.polimi.it
[2] Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain
jalcala@decsai.ugr.es, herrera@decsai.ugr.es

Abstract. Data Mining is most commonly used in attempts to induce association rules from transaction data. Most previous studies focused on binary-valued transactions, however the data in real-world applications usually consists of quantitative values. In the last few years, many researchers have proposed Evolutionary Algorithms for mining interesting association rules from quantitative data. In this paper, we present a preliminary study on the evolutionary extraction of quantitative association rules. Experimental results on a real-world dataset show the effectiveness of this approach.

Keywords: Association Rules, Data Mining, Evolutionary Algorithms, Genetic Algorithms.

1 Introduction

Data Mining (DM) is the process for the automatic discovery of high level knowledge from real-world, large and complex datasets. Association Rules Mining (ARM) is one of the several DM techniques described in the literature [1].

Association rules are used to represent and identify dependencies between items in a database [2]. These are implications of the form $X \rightarrow Y$, where X and Y are sets of items and $X \cap Y = \varnothing$. It means that if all the items in X exist in a transaction then all the items in Y are also in the transaction with a high probability, and X and Y should not have common items [3]. Many previous studies focused on databases with binary values, however the data in real-world applications usually consist of quantitative values. Designing DM algorithms, able to deal with various types of data, presents a challenge to workers in this research field.

Lately, many researchers have proposed Evolutionary Algorithms (EAs) [4] for mining association rules from quantitative data [5], [6], [7], [8]. EAs, particularly Genetic Algorithms (GAs) [9], [10], are considered as one of the most successful search techniques for complex problems and have proved to be an important technique for learning and knowledge extraction. The main motivation for applying EAs to knowledge extraction task is that they are robust and adaptive search methods that perform a global search in place of candidate solutions. Moreover, EAs let to obtain feasible solutions in a limited amount of time. Hence, they have been a growing interest in DM.

E. Corchado et al. (Eds.): HAIS 2009, LNAI 5572, pp. 646–653, 2009.

In this paper, we present a preliminary study on the evolutionary extraction of quantitative association rules. We perform an experimental study to show the behaviour of three GAs (*EARMGA*, *GAR*, and *GENAR*) along with two classical algorithms (a *Trie*-based implementation of *Apriori*, and *Eclat*) for the extraction of association rules on a real-world dataset. Moreover, several experiments have been carried out to analyse the scalability of these methods.

This paper is arranged as follows. The next section provides brief preliminaries on the genetic extraction of association rules. Section 3 describes the five analysed methods. Section 4 shows the results of the experimental study. Finally, Section 5 points out some conclusions.

2 Preliminaries: Genetic Extraction of Association Rules

Although GAs were not specifically designed for learning, but rather as global search algorithms, they offer a set of advantages for machine learning. Many methodologies for machine learning are based on the search of a good model among all possible models within this space. In this sense, they are very flexible because the same GA can be used with different representations. When considering a rule based system and focusing on learning rules, genetic learning methods follow two approaches in order to encode rules within a population of individuals [11]:

- The "Chromosome = Set of rules", also called the *Pittsburgh* approach, in which each individual represents a rules set [12]. In this case, a chromosome evolves a complete Rule Base (RB) and they compete among themselves along the evolutionary process. GABIL is a proposal that follows this approach [13].
- The "Chromosome = Rule" approach, in which each individual encodes a single rule, and the whole rule set is provided by combining several individuals in a population (rule cooperation) or via different evolutionary runs (rule competition). Within the "Chromosome = Rule" approach, there are three generic proposals:
  - *Michigan* approach, in which each individual stands for an association rule. These kinds of systems are usually called learning classifier systems [14]. They are rule-based, message-passing systems that employ reinforcement learning and a GA to learn rules that guide their performance in a given environment. The GA is used for detecting new rules that replace the bad ones via a competition among chromosomes throughout the evolutionary process.
  - *IRL (Iterative Rule Learning)* approach, in which each chromosome represents a rule. Chromosomes compete in every GA run, choosing the best rule per run. The global solution is formed by the best rules obtained when the algorithm has been executed multiple times. SIA [15] is a proposal that follows this approach.
  - *GCCL (genetic cooperative-competitive learning)* approach, in which the population, or one of its subsets, encodes the RB. Moreover, the chromosomes compete and cooperate simultaneously. COGIN [16] is an example of this approach.

In the literature we can find interesting works based on the Pittsburgh approach [17], the Michigan approach [5], and the IRL approach [6], [7] for mining association rules from quantitative data.

3 Association Rules Mining: Algorithms for the Analysis

In this section we introduce the five methods used for our experimental study. We can make out two types of algorithms:

- Classical algorithms: we analysed *Apriori* [18], [19] and *Eclat* [20].
- GAs for ARM: we analysed *EARMGA* [5], *GAR* [6], and *GENAR* [7].

We present the aforementioned methods in more details in the next subsections.

3.1 Association Rules Mining through Classical Algorithms: Apriori and Eclat

Among classical algorithms, it is worthwhile to mention *Apriori* because is the first successful algorithm used for mining association rules. Several implementations based on this method can be found in the literature, basically, with the aim of speeding up the support counting [19], [21], [22]. In this paper, we have used a fast implementation of Apriori which uses *Trie* [19]. Moreover, we have chosen to analyse *Eclat* [20] because it exploits a different strategy to search for frequent itemsets.

The main aim of *Apriori* is to explore the search space by means of the *downward closure property*. The latter states that any subset of a frequent itemset must also be frequent. As a consequence, it generates candidates for the current iteration by means of frequent itemsets collected from the previous iteration. Then, it enumerates all the subsets for each transaction and increments the support of candidates which match them. Finally, those having a user-specified minimum support are marked as frequent for the next iteration. This process is repeated until all frequent itemsets have been found. Thus, Apriori follows a *breadth-first* strategy to generate candidates.

On the other hand, *Eclat* employs a *depth-first* strategy. It generates candidates by extending prefixes of an itemset until an infrequent one is found. In that case, it simply backtracks on the next prefix and then recursively applies the above procedure. Unlike Apriori, the support counting is achieved by adopting a vertical layout. That is, for each item in the dataset, it first constructs the list of all transaction identifiers (*tid-list*) containing that item. Then, it counts the support by merely intersecting two or more tid-lists to check whether they have items in common. In that case, the support count is equal to the size of this resulting set.

Most of the classical algorithms usually identify relationships among transactions in datasets with binary values. In order to apply these methods on datasets containing real values, a pre-processing step must be accomplished.

3.2 Association Rules Mining through Evolutionary Algorithms: EARMGA, GAR, and GENAR

We have used the following GAs in the literature to perform the ARM task:

- *EARMGA* [5]. It is based on the discovery of quantitative association rules.
- *GAR* [6]. It searches for frequent itemsets by dealing with numerical domains.
- *GENAR* [7]. It mines directly association rules by handling numerical domains.

A chromosome in EARMGA encodes a generalized k-rule, where k indicates the desired length. Since we may handle association rules with more than one item in the

consequent, the first gene stores an index representing the end of the antecedent part. In order to encode uniquely a rule into a chromosome, both antecedent and consequent attributes are sorted two-segmentally in an ascending order. On the contrary, the remaining k genes encode items. Each item is represented by a pair of values, where the first value is an attribute's index ranged from 1 to the number of attributes in the dataset, while the second one is a *gapped interval*. The authors have defined a gapped interval as the union of a finite number of *base intervals* obtained once a uniform discretization process has been accomplished on all attributes. Notice that we do not need to partition the domains of categorical attributes because here lower bound and upper bound basically coincide. Nevertheless, a base interval is always represented by an integer, apart from the kind of attributes we deal with. As a consequence, a gapped interval is a set of these values. Now we give some details of the genetic operators applied on each chromosome:

- *Selection*: a chromosome is selected only if the product of its fitness value with a random number is less than a given probability of selection (*ps*).
- *Crossover*: all the selected chromosomes have the chance to reproduce offsprings at a probability of crossover (*pc*). This operation simply consists in exchanging a segment of genes from the first chromosome to the second one and vice-versa, depending on two randomly generated crossover-points.
- *Mutation*: by considering both a probability of mutation (*pm*) and the fitness value, a chromosome is altered by changing the boundary between antecedent attributes and consequent attributes within the same rule. In addition, the operator randomly chooses a gene and modifies the attribute's index along with its gapped interval. Notice that the new gapped interval is always a union of base intervals which now form a sub-domain of the new attribute.

Finally, the ARM problem has been restated by the authors of this work because the algorithm searches for k-association rules by evaluating fitness values that represent measures of interest known as *positive confidence* [5].

On the contrary, *GAR* follows different strategies. First, a chromosome is composed of a variable number of genes, between 2 and n, where n is the maximum number of attributes. However, as we find frequent itemsets with this method, it is afterwards necessary to run another procedure for generating association rules. Moreover, we do not have to discrete a priori the domain of the attributes since each gene is represented by an upper bound and a lower bound along with an identifier for the attribute. We briefly recall the genetic operators for this method as follows:

- *Selection*: an elitist strategy is used for selecting a percentage (*ps*) of chromosomes which have the best fitness values from the current population. These are the first individuals that later form the new population.
- *Crossover*: the new population is completed by reproducing offsprings until reaching a desired size. To do that, the parents are randomly chosen at a probability of *pc*. Then, we obtain two different offsprings whenever their parents contain genes having the same attribute. In that case, their intervals could simply be exchanged by considering all the possible combinations between them, but, at last, always two chromosomes should be generated. Finally, only the best of them will be added to the population.

- *Mutation*: at a probability of *pm*, it alters one gene in the way that each limit could randomly decrease or increase its current value.

The fitness function tends to reward frequent itemsets that have a high support as well as a high number of attributes. In addition, it punishes frequent itemsets which have already covered a record in the dataset and which have intervals too large.

GENAR was the first attempt by the same authors of GAR in handling continuous domains. Here, a chromosome represents an association rules that always contain intervals. Nevertheless, the length of rules is fixed to the number of attributes and only the last attribute acts as consequent. The crossover operator employs a one-point strategy to reproduce offspring chromosomes, whereas the remaining operators are similar to those defined in GAR. For this reason, we do not give further details [7]. On the contrary, its fitness function only considers the support count of rules and punishes rules which have already covered the same records in the dataset.

4 Experimental Results

To evaluate the usefulness of the genetic extraction of association rules several experiments have been carried out on a real-world dataset named *House_16* (*HH*). It concerns a study to predict the median price of the house in the region by considering both the demographic composition and the state of housing market. This dataset contains 22,784 records with 17 quantitative attributes[1].

The parameters used for running the algorithms are:

- *Apriori* and *Eclat*: *minimum Support* = 0.1 and *minimum Confidence* = 0.8.
- *EARMGA*: *maxloop* = 100, *popsize* = 100, $k = 4$, *ps* = 0.75, *pc* = 0.7, *pm* = 0.1, and $\alpha = 0.01$.
- *GAR*: *nItemset* = 100, *nGen* = 100, *popsize* = 100, *ps* = 0.25, *pc* = 0.7, *pm* = 0.1, *af* = 2.0, $\omega = 0.4$, $\psi = 0.7$, and $\mu = 0.5$.
- *GENAR*: *nRules* = 100, *nGen* = 100, *popsize* = 100, *ps* = 0.25, *pc* = 0.7, *pm* = 0.1, *af* = 2.0, and *pf* = 0.7.

Moreover, a uniform discretization in 4 intervals was accomplished on the continuous attributes in the dataset only if needed by one of these algorithms.

The results returned by all the methods are presented in Table 1, where *Itemsets* stands for the number of the discovered frequent itemsets, *Rules* for the number of the generated association rules, *Avg_Sup* and *Avg_Conf*, respectively, for the average support and the average confidence of the mined rules, *Avg_Amp* for the average length of the rules antecedents, and *%Records* for the percentage of records covered by these rules on the total records in the dataset. We remark that our results always refer to association rules which have minimum confidence greater or equal than 0.8.

[1] This database was designed on the basis of data provided by US Census Bureau [http://www.census.gov] (under Lookup Access [http://www.census.gov/cdrom/lookup]: Summary Tape File 1).

Table 1. Results for the dataset *HH*

Algorithm	Itemsets	Rules	Avg_Sup	Avg_Conf	Avg_Amp	%Records
Apriori	305229	1982211	0.22	0.96	7	**100.00**
Eclat	305229	1982211	0.22	0.96	7	**100.00**
EARMGA	-	**100**	0.24	**1.00**	**2**	**100.00**
GAR	100	167	**0.73**	0.94	**2**	**100.00**
GENAR	-	**100**	0.46	0.99	16	88.60

Analysing the results presented in Table 1, we highlight the following issues:

- The classical methods returned a large set of association rules (approximately 2 million) with the minimum support and confidence. On the contrary, the GAs let us to obtain a reduced set of high quality rules, thus denoting more interesting patterns. For instance, EARMGA mined 100 rules having the maximum average confidence and GAR mined 167 association rules with very high average support.
- EARMGA and GAR discovered association rules involving only few attributes in the antecedents, giving the advantage of a better understanding from the user's perspective. Notice that, GENAR considers rules of the maximal length due to the fact that it always involves all attributes in the dataset.
- The rules returned by the GAs achieved a good covering of the records although the number of rules is restricted by the population size. As an example, EARMGA and GAR covered 100% of the records in the dataset.

Several experiments have been also carried out to analyse the scalability of the algorithms. All the experiments were performed by using a Pentium Corel 2 Quad, 2.5GHz CPU with 4GB of memory and by running Linux. Fig. 1 and Fig. 2 show the relationship between the runtime and, respectively, the number of records and the number of attributes in the dataset for each algorithm.

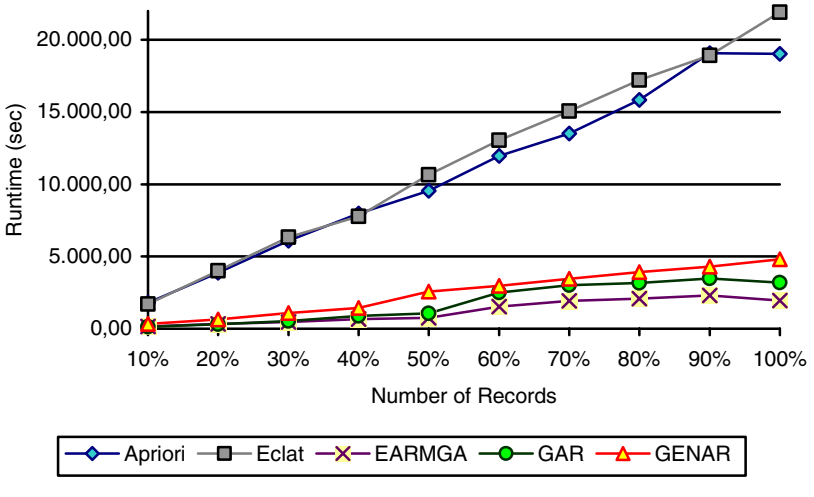

Fig. 1. Relationship between runtimes and number of records with all the attributes

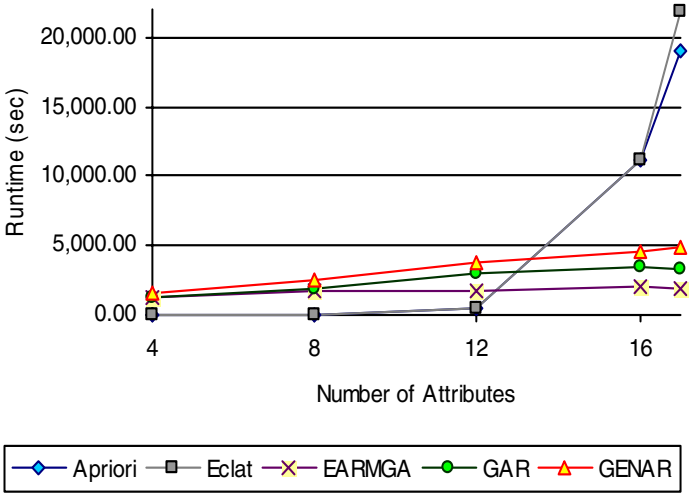

Fig. 2. Relationship between runtimes and number of attributes with 100% of the records

It can be easily seen from Fig. 1 that the runtime of the classical methods increase with respect to the runtime of the GAs since we increase the size of the problem. In Fig. 2 we notice that the classical algorithms expended a large amount of time when the number of attributes is also increased. On the contrary, from both the figures we can see how the GAs scaled quite linearly when performing the ARM task on the used dataset.

Finally, it is worthwhile to remark that the GAs expended a reasonable amount of time in mining a reduced set of high quality association rules. Nevertheless, increasing the population size of the GAs would increase the runtimes of these methods which, eventually, could be higher than those of Apriori and Eclat.

5 Concluding Remarks

In this paper, we have presented a preliminary study on the extraction of association rules by means of GAs. By evaluating the results on a real-world dataset, we point out the following conclusions about the effectiveness of these methods:

- GAs let us to obtain a reduced set of high quality association rules in a reasonable amount of time and also achieving a good covering of the dataset.
- The association rules returned by GAs consider few attributes in their antecedents, giving the advantage of a better understanding from the user's perspective.

Acknowledgments. This paper has been supported by the Spanish Ministry of Education and Science under Project TIN2008-06681-C06-01.

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)

2. Zhang, C., Zhang, S.: Association Rule Mining: Models and Algorithms. LNCS(LNAI), vol. 2307. Springer, Heidelberg (2002)
3. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: ACM SIGMOD ICMD, pp. 207–216. ACM Press, Washington (1993)
4. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing, 1st edn. Natural Computing Series. Springer, Heidelberg (2003)
5. Yan, X., Zhang, C., Zhang, S.: Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support. Expert Systems with Applications 36(2), 3066–3076 (2009)
6. Mata, J., Alvarez, J.L., Riquelme, J.C.: An Evolutionary Algorithm to Discover Numeric Association Rules. In: Proc. of ACM SAC 2002, Madrid, Spain, pp. 590–594 (2002)
7. Mata, J., Alvarez, J.L., Riquelme, J.C.: Mining Numeric Association Rules with Genetic Algorithms. In: 5th International Conference on Artificial Neural Networks and Genetic Algorithms, Prague, pp. 264–267 (2001)
8. Mata, J., Alvarez, J.L., Riquelme, J.C.: Discovering Numeric Association Rules via Evolutionary Algorithm. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS, vol. 2336, pp. 40–51. Springer, Heidelberg (2002)
9. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, New York (1998)
10. Holland, J.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, London (1975)
11. Herrera, F.: Genetic Fuzzy Systems: Taxonomy, Current Research Trends and Prospects. Evolutionary Intelligence 1, 27–46 (2008)
12. Smith, S.: A learning system based on genetic algorithms. Ph.D. thesis. University of Pittsburgh (1980)
13. De Jong, K., Spears, W., Gordon, D.: Using genetic algorithms for concept learning. Machine Learning 13(2-3), 161–188 (1993)
14. Holland, J., Reitman, J.: Cognitive systems based on adaptive algorithms. In: Waterman, D.A., Hayes-Roth, F. (eds.) Patter-directed inference systems, pp. 1148–1158. Academic Press, London (1978)
15. Venturini, G.: SIA: a supervised inductive algorithm with genetic search for learning attrib-ute based concepts. In: ECML, Vienna, Austria, pp. 280–296 (1993)
16. Greene, D.P., Smith, S.F.: Competition-based induction of decision models from examples. Machine Learning 13(2-3), 229–257 (1993)
17. Pei, M., Goodman, E., Punch, W.: Pattern Discovery from Data using Genetic Algorithm. In: Proc. of PAKDD 1997, Singapore, pp. 264–276 (1997)
18. Srikant, R., Agrawal, R.: Mining Quantitative Association Rules in Large Relational Tables. In: Proc. of ACM SIGMOD ICMD 1996, pp. 1–12. ACM Press, Montreal (1996)
19. Borgelt, C.: Efficient Implementations of Apriori and Eclat. In: Workshop on Frequent Itemset Mining Implementations. CEUR Workshop Proc. 90, Florida, USA (2003)
20. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New Algorithms for Fast Discovery of Association Rules. Technical Report 651, University of Rochester (1997)
21. Bodon, F.: A trie-based APRIORI implementation for mining frequent item sequences. In: 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations, Chicago, Illinois, USA, pp. 56–65. ACM Press, New York (2005)
22. Han, J., Pei, H., Yin, Y.: Mining Frequent Patterns without Candidate Generation. In: Proc. of ACM SIGMOD ICMD 2000, Dallas, TX. ACM Press, New York (2000)